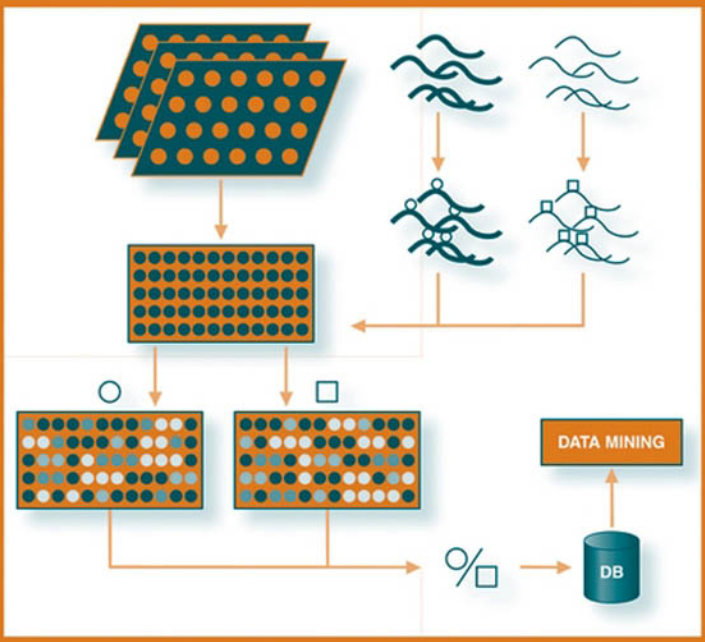
Genomics and Proteomics Engineering in Medicine and Biology



METIN AKAY

IEEE Press Series in Biomedical Engineering
Metin Akay, *Series Editor*

IEEE Engineering in Medicine and Biology Society, *Sponsor*

GENOMICS AND PROTEOMICS ENGINEERING IN MEDICINE AND BIOLOGY

IEEE Press
445 Hoes Lane
Piscataway, NJ 08854

GENOMICS AND PROTEOMICS ENGINEERING IN MEDICINE AND BIOLOGY

Edited by

Metin Akay

IEEE Engineering in Medicine and Biology Society, *Sponsor*

IEEE PRESS

WILEY-INTERSCIENCE
A JOHN WILEY & SONS, INC., PUBLICATION

Published by John Wiley & Sons, Inc. Published simultaneously in Canada.

Library of Congress Cataloging-in-Publication Data is available

ISBN-13 978-0-471-63181-1
ISBN-10 0-471-63181-7

Printed in the United States of America

10 9 8 7 6 5 4 3 2 1

To the memory of my late brother, Çetin Akay, who dedicated his short but meaningful life to the well-being and happiness of others as well as a democratic and secular Turkey.

May God bless his soul.

CONTENTS

Preface **xi**

Contributors **xiii**

1. Qualitative Knowledge Models in Functional Genomics and Proteomics **1**
Mor Peleg, Irene S. Gabashvili, and Russ B. Altman

1.1. Introduction 1
1.2. Methods and Tools 3
1.3. Modeling Approach and Results 6
1.4. Discussion 19
1.5. Conclusion 20
References 21

2. Interpreting Microarray Data and Related Applications Using Nonlinear System Identification **25**
Michael Korenberg

2.1. Introduction 25
2.2. Background 25
2.3. Parallel Cascade Identification 30
2.4. Constructing Class Predictors 34
2.5. Prediction Based on Gene Expression Profiling 35
2.6. Comparing Different Predictors Over the Same Data Set 46
2.7. Concluding Remarks 48
References 49

3. Gene Regulation Bioinformatics of Microarray Data **55**
Gert Thijs, Frank De Smet, Yves Moreau, Kathleen Marchal, and Bart De Moor

3.1. Introduction 55
3.2. Introduction to Transcriptional Regulation 57
3.3. Measuring Gene Expression Profiles 59
3.4. Preprocessing of Data 61
3.5. Clustering of Gene Expression Profiles 63

3.6. Cluster Validation 70
3.7. Searching for Common Binding Sites of Coregulated Genes 76
3.8. Inclusive: Online Integrated Analysis of Microarray Data 87
3.9. Further Integrative Steps 89
3.10. Conclusion 90
References 91

4. Robust Methods for Microarray Analysis 99
George S. Davidson, Shawn Martin, Kevin W. Boyack, Brian N. Wylie, Juanita Martinez, Anthony Aragon, Margaret Werner-Washburne, Mónica Mosquera-Caro, and Cheryl Willman

4.1. Introduction 99
4.2. Microarray Experiments and Analysis Methods 100
4.3. Unsupervised Methods 103
4.4. Supervised Methods 117
4.5. Conclusion 127
References 128

5. In Silico Radiation Oncology: A Platform for Understanding Cancer Behavior and Optimizing Radiation Therapy Treatment 131
G. Stamatakos, D. Dionysiou, and N. Uzunoglu

5.1. *Philosophiae Tumoralis Principia Algorithmica*: Algorithmic Principles of Simulating Cancer on Computer 131
5.2. Brief Literature Review 133
5.3. Paradigm of Four-Dimensional Simulation of Tumor Growth and Response to Radiation Therapy In Vivo 135
5.4. Discussion 148
5.5. Future Trends 150
References 150

6. Genomewide Motif Identification Using a Dictionary Model 157
Chiara Sabatti and Kenneth Lange

6.1. Introduction 157
6.2. Unified Model 160
6.3. Algorithms for Likelihood Evaluation 164
6.4. Parameter Estimation via Minorization–Maximization Algorithm 167
6.5. Examples 170
6.6. Discussion and Conclusion 171
References 172

7. Error Control Codes and the Genome 173
Elebeoba E. May

7.1. Error Control and Communication: A Review 173

7.2. Central Dogma as Communication System 180
7.3. Reverse Engineering the Genetic Error Control System 184
7.4. Applications of Biological Coding Theory 203
References 205

8. Complex Life Science Multidatabase Queries 209
Zina Ben Miled, Nianhua Li, Yue He, Malika Mahoui, and Omran Bukhres

8.1. Introduction 209
8.2. Architecture 212
8.3. Query Execution Plans 214
8.4. Related Work 219
8.5. Future Trends 222
References 223

9. Computational Analysis of Proteins 227
Dimitrios I. Fotiadis, Yorgos Goletsis, Christos Lampros, and Costas Papaloukas

9.1. Introduction: Definitions 227
9.2. Databases 229
9.3. Sequence Motifs and Domains 232
9.4. Sequence Alignment 235
9.5. Modeling 241
9.6. Classification and Prediction 242
9.7. Natural Language Processing 248
9.8. Future Trends 252
References 252

10. Computational Analysis of Interactions Between Tumor and Tumor Suppressor Proteins 257
E. Pirogova, M. Akay, and I. Cosic

10.1. Introduction 257
10.2. Methodology: Resonant Recognition Model 261
10.3. Results and Discussions 265
10.4. Conclusion 284
References 285

Index 289

About the Editor 299

PREFACE

The biological sciences have become more quantitative and information-driven since emerging computational and mathematical tools facilitate collection and analysis of vast amounts of biological data. Complexity analysis of biological systems provides biological knowledge for the organization, management, and mining of biological data by using advanced computational tools. The biological data are inherently complex, nonuniform, and collected at multiple temporal and spatial scales. The investigations of complex biological systems and processes require an extensive collaboration among biologists, mathematicians, computer scientists, and engineers to improve our understanding of complex biological process from gene to system. Lectures in the summer school expose attendees to the latest developments in these emerging computational technologies and facilitate rapid diffusion of these mathematical and computational tools in the biological sciences. These computational tools have become powerful tools for the study of complex biological systems and signals and can be used for characterizing variability and uncertainty of biological signals across scales of space and time since the biological signals are direct indicators of the biological state of the corresponding cells or organs in the body.

The integration and application of mathematics, engineering, physics and computer science have been recently used to better understand the complex biological systems by examining the structure and dynamics of cell and organ functions. This emerging field called "Genomics and Proteomics Engineering" has gained tremendous interest among molecular and cellular researchers since it provides a continuous spectrum of knowledge. However, this emerging technology has not been adequately presented to biological and bioengineering researchers. For this reason, an increasing demand can be found for interdisciplinary interactions among biologists, engineers, mathematicians, computer scientists and medical researchers in these emerging technologies to provide the impetus to understand and develop reliable quantitative answers to the major integrative biological and biomedical challenges.

The main objective of this edited book is to provide information for biological science and biomedical engineering students and researchers in genomics and proteomics sciences and systems biology. Although an understanding of genes and proteins are important, the focus is on understanding a system's structure and dynamics of several gene regulatory networks and their biochemical interactions.

System-level understanding of biology is derived using mathematical and engineering methods to understand complex biological processes. It exposes readers with biology background to the latest developments in proteomics and genomics engineering. It also addresses the needs of both students and postdoctoral fellows in computer science and mathematics who are interested in doing research in biology and bioengineering since the book provides exceptional insights into the fundamental challenges in biology.

I am grateful to Jeanne Audino of the IEEE Press and Lisa Van Horn of Wiley for their help during the editing of this book. Working in concert with them and the contributors really helped me with content development and to manage the peer-review process.

Finally, many thanks to my wife, Dr. Yasemin M. Akay, and our son, Altug R. Akay, for their support, encouragement, and patience. They have been my driving source. I also thank Jeremy Romain for his help in rearranging the chapters and getting the permission forms from the contributors.

METIN AKAY

Tempe, Arizona
September 2006

CONTRIBUTORS

Metin Akay, Harrington Department of Bioengineering, Fulton School of Engineering, Arizona State University, Tempe, Arizona

Russ B. Altman, Stanford Medical Informatics, Stanford University, Stanford, California

Anthony Aragon, Department of Biology, University of New Mexico, Albuquerque, New Mexico

Zia Ben Miled Electrical and Computer Engineering, Purdue School of Engineering and Technology, IUPUI, Indianapolis, Indiana

Kevin W. Boyack, Computation, Computers, Information and Mathematics, Sandia National Laboratories, Albuquerque, New Mexico

Omran Bukhres Department of Computer and Information Science, IUPUI, Indianapolis, Indiana

I. Cosic, School of Electrical and Computer Engineering, RMIT University, Melbourne, Australia

George S. Davidson, Computation, Computers, Information and Mathematics, Sandia National Laboratories, Albuquerque, New Mexico

Bart De Moor, Department of Electrical Engineering (ESAT-SCD), Katholieke Universiteit Leuven, Leuven, Belgium

Frank De Smet, Department of Electrical Engineering (ESAT-SCD), Katholieke Universiteit Leuven, Leuven, Belgium

D. Dionysiou, In Silico Oncology Group, Laboratory of Microwaves and Fiber Optics, Institute of Communication and Computer Systems, Department of Electrical and Computer Engineering, National Technical University of Athens, Zografos, Greece

Dimitrios I. Fotiadis, Unit of Medical Technology and Intelligent Information Systems, Department of Computer Science, University of Ioannina, Ionnina, Greece

Irene S. Gabashvili, Hewlett Packard Labs, Palo Alto, California

Yorgos Goletsis, Unit of Medical Technology and Intelligent Information Systems, Department of Computer Science, University of Ioannina, Ionnina, Greece

Yue He Electrical and Computer Engineering, Purdue School of Engineering and Technology, IUPUI, Indianapolis, Indiana

Michael Korenberg, Department of Electrical and Computer Engineering, Queen's University, Kingston, Ontario, Canada

Christos Lampros, Unit of Medical Technology and Intelligent Information Systems, Department of Computer Science, University of Ioannina, Ionnina, Greece

Kenneth Lange, Biomathematics, Human Genetics, and Statistics Department, University of California at Los Angeles, Los Angeles, California 90095

Nianhua Li Electrical and Computer Engineering, Purdue School of Engineering and Technology, IUPUI, Indianapolis, Indiana

Malika Mahoui School of Informatics, IUPUI, Indianapolis, Indiana

Kathleen Marchal, Department of Electrical Engineering (ESAT-SCD), Katholieke Universiteit Leuven, Leuven, Belgium

Shawn Martin, Computation, Computers, Information and Mathematics, Sandia National Laboratories, Albuquerque, New Mexico

Juanita Martinez, Department of Biology, University of New Mexico, Albuquerque, New Mexico

Elebeoba E. May, Computational Biology Department, Sandia National Laboratories, Albuquerque, New Mexico

Yves Moreau, Department of Electrical Engineering (ESAT-SCD), Katholieke Universiteit Leuven, Leuven, Belgium

Monica Mosquera-Caro, Cancer Research and Treatment Center, Department of Pathology, University of New Mexico, Albuquerque, New Mexico

Costas Papaloukas, Unit of Medical Technology and Intelligent Information Systems, Department of Computer Science, University of Ioannina, Ionnina, Greece

Mor Peleg, Department of Management Information Systems, University of Haifa, Israel

E. Pirogova, School of Electrical and Computer Engineering, RMIT University, Melbourne, Australia

Chiara Sabatti, Human Genetics and Statistics Department, University of California at Los Angeles, Los Angeles, California

G. Stamatakos, In Silico Oncology Group, Laboratory of Microwaves and Fiber Optics, Institute of Communication and Computer Systems, Department of Electrical and Computer Engineering, National Technical University of Athens, Zografos, Greece

Gert Thijs, Department of Electrical Engineering (ESAT-SCD), Katholieke Universiteit Leuven, Leuven, Belgium

N. Uzunoglu, In Silico Oncology Group, Laboratory of Microwaves and Fiber Optics, Institute of Communication and Computer Systems, Department of Electrical and Computer Engineering, National Technical University of Athens, Zografos, Greece

Margaret Werner-Washburne, Department of Biology, University of New Mexico, Albuquerque, New Mexico

Cheryl Willman, Cancer Research and Treatment Center, Department of Pathology, University of New Mexico, Albuquerque, New Mexico

Brian N. Wylie, Computation, Computers, Information and Mathematics, Sandia National Laboratories, Albuquerque, New Mexico

CHAPTER 1

Qualitative Knowledge Models in Functional Genomics and Proteomics

MOR PELEG, IRENE S. GABASHVILI, and RUSS B. ALTMAN

1.1. INTRODUCTION

Predicting pathological phenotypes based on genetic mutations remains a fundamental and unsolved issue. When a gene is mutated, the molecular functionality of the gene product may be affected and many cellular processes may go awry. Basic molecular functions occur in networks of interactions and events that produce subsequent cellular and physiological functions. Most knowledge of these interactions is represented diffusely in the published literature, Excel lists, and specialized relational databases and so it is difficult to assess our state of understanding at any moment. Thus it would be very useful to systematically store knowledge in data structures that allow the knowledge to be evaluated and examined in detail by scientists as well as computer algorithms. Our goal is to develop technology for representing qualitative, noisy, and sparse biological results in support of the eventual goal of fully accurate quantitative models.

In a recent paper, we described an ontology that we developed for modeling biological processes [1]. *Ontologies* provide consistent definitions and interpretations of concepts in a domain of interest (e.g., biology) and enable software applications to share and reuse the knowledge consistently [2]. Ontologies can be used to perform logical inference over the set of concepts to provide for generalization and explanation facilities [3]. Our biological process ontology combines and extends two existing components: a workflow model and a biomedical ontology, both described in the methods and tools section. Our resulting framework possesses the following properties: (1) it allows qualitative modeling of structural and functional aspects of a biological system, (2) it includes biological and medical concept models to allow for querying biomedical information using biomedical abstractions, (3) it allows

Genomics and Proteomics Engineering in Medicine and Biology. Edited by Metin Akay

hierarchical models to manage the complexity of the representation, (4) it has a sound logical basis for automatic verification, and (5) it has an intuitive, graphical representation.

Our application domain is disease related to transfer ribonucleic acid (tRNA). Transfer RNA constitutes a good test bed because there exists rich literature on tRNA molecular structure as well as the diseases that result from abnormal structures in mitochondria (many of which affect neural processes). The main role of tRNA molecules is to be part of the machinery for the translation of the genetic message, encoded in messenger RNA (mRNA), into a protein. This process employs over 20 different tRNA molecules, each specific for one amino acid and for a particular triplet of nucleotides in mRNA (codon) [4]. Several steps take place before a tRNA molecule can participate in translation. After a gene coding for tRNA is transcribed, the RNA product is folded and processed to become a tRNA molecule. The tRNA molecules are covalently linked (acylated) with an amino acid to form amino-acylated tRNA (aa-tRNA). The aa-tRNA molecules can then bind with translation factors to form complexes that may participate in the translation process. There are three kinds of complexes that participate in translation: (i) an *initiation complex* is formed by exhibiting tRNA mimicry release factors that bind to the stop codon in the mRNA template or by a misfunctioning tRNA complexed with guanidine triphosphate (GTP) and elongation factor causing abnormal termination, and (iii) a *ternary complex* is formed by binding elongating aa-tRNAs (tRNAs that are acylated to amino acids other than formylmethionine) with GTP and the elongation factor EF-tu. During the translation process, tRNA molecules recognize the mRNA codons one by one, as the mRNA molecule moves through the cellular machine for protein synthesis: the ribosome. In 1964, Watson introduced the classical two-site model, which was the accepted model until 1984 [5]. In this model, the ribosome has two regions for tRNA binding, so-called aminoacyl (A) site and peptidyl (P) site. According to this model, initiation starts from the P site, but during the normal cycle of elongation, each tRNA enters the ribosome from the A site and proceeds to the P site before exiting into the cell's cytoplasm. Currently, it is hypothesized that the ribosome has at least three regions for tRNA binding: the A and P sites and an exit site (E site) through which the tRNA exits the ribosome into the cell's cytoplasm [6]. Protein synthesis is terminated when a stop codon is reached at the ribosomal A site and recognized by a specific termination complex, probably involving factors mimicking tRNA. Premature termination (e.g., due to a mutation in tRNA) can also be observed [7].

When aa-tRNA molecules bind to the A site, they normally recognize and bind to matching mRNA codons—a process known as reading. The tRNA mutations can cause abnormal reading that leads to mutated protein products of translation. Types of abnormal reading include (1) *misreading*, where tRNA with nonmatching amino acid binds to the ribosome's A site; (2) *frame shifting*, where tRNA that causes frame shifting (e.g., binds to four nucleotides of the mRNA at the A site) participates in elongation; and (3) *halting*, where tRNA that cause premature termination (e.g., tRNA that is not acetylated with an amino acid) binds to the A site.

These three types of errors, along with the inability to bind to the A site or destruction by cellular enzymes due to misfolding, can create complex changes in protein profiles of cells. This can affect all molecular partners of produced proteins in the chain of events connecting genotype to phenotype and produce a variety of phenotypes. Mutations in human tRNA molecules have been implicated in a wide range of disorders, including myopathies, encephalopathies, cardiopathies, diabetes, growth retardation, and aging [8]. Development of models that consolidate and integrate our understanding of the molecular foundations for these diseases, based on available structural, biochemical, and physiological knowledge, is therefore urgently needed.

In a recent paper [9], we discussed an application of our biological process ontology to genomics and proteomics. This chapter extends the section on general computer science theories, including Petri Nets, ontologies, and information systems modeling methodologies, as well as extends the section on biological sources of information and discusses the compatibility of our outputs with popular databases and modeling environments.

The chapter is organized as follows. Section 1.2 describes the components we used to develop the framework and the knowledge sources for our model. Section 1.3 discusses our modeling approach and demonstrates our knowledge model and the way in which information can be viewed and queried using the process of translation as examples. We conclude with a discussion and conclusion.

1.2. METHODS AND TOOLS

1.2.1. Component Ontologies

Our framework combines and extends two existing components: The workflow model and biomedical ontology. The *workflow model* [10] consists of a process model and an organizational (participants/role) model. The process model can represent ordering of processes (e.g., protein translation) and the structural components that participate in them (e.g., protein). Processes may be of low granularity (high-level processes) or of high granularity (low-level processes). High-level processes are nested to control the complexity of the presentation for human inspection. The *participants/role model* represents the relationships among participants (e.g., an EF-tu is a member of the elongation factors collection in prokaryotes) and the roles that participants play in the modeled processes (e.g., EF-tu has enzymic function: GTPase). We used the workflow model as a biological process model by mapping workflow activities to biological processes, organizational units to biomolecular complexes, humans (individuals) to their biopolymers and networks of events, and roles to biological processes and functions.

A significant advantage of the workflow model is that it can map to *Petri Nets* [11], a mathematical model that represents concurrent systems, which allows verification of formal properties as well as qualitative simulation [12]. A *Petri Net* is represented by a directed, bipartite graph in which nodes are either places or

transitions, where places represent conditions (e.g., parasite in the bloodstream) and transitions represent activities (e.g., invasion of host erythrocytes). Tokens that are placed on places define the state of the Petri Net (marking). A token that resides in a place signifies that the condition that the place represents is true. A Petri Net can be executed in the following way. When all the places with arcs to a transition have a token, the transition is enabled, and may fire, by removing a token from each input place and adding a token to each place pointed to by the transition. *High-level Petri Nets*, used in this work, include extensions that allow modeling of time, data, and hierarchies.

For the biomedical ontology, we combine the Transparent Access to Multiple Biological Information Sources (TAMBIS) [13] with the Unified Medical Language System (UMLS) [14]. TAMBIS is an ontology for describing data to be obtained from bioinformatics sources. It describes biological entities at the molecular level. UMLS describes clinical and medical entities. It is a publicly available federation of biomedical controlled terminologies and includes a semantic network with 134 semantic types that provides a consistent categorization of thousands of biomedical concepts. The 2002AA edition of the UMLS Metathesaurus includes 776,940 concepts and 2.1 million concept names in over 60 different biomedical source vocabularies. We augmented these two core terminological models [1] to represent mutations and their effects on biomolecular structures, biochemical functions, cellular processes, and clinical phenotypes. The extensions include classes for representing (1) mutations and alleles and their relationship to sequence components, (2) a nucleic acid three-dimensional structure linked to secondary and primary structural blocks, and (3) a set of composition operators, based on the nomenclature of composition relationships, due to Odell [15].

Odell introduced a nomenclature of six kinds of composition. We are using three of these composition relationships in our model. The relationship between a biomolecular complex (e.g., ternary complex) and its parts (e.g., GTP, EF-tu, aa-tRNA) is a *component–integral object composition*. This relationship defines a *configuration* of parts within a whole. A *configuration* requires the parts to bear a particular functional or structural relationship to one another as well as to the object they constitute. The relationship between an individual molecule (e.g., tRNA) and its domains (e.g., D domain, T domain) is a *place–area composition*. This relationship defines a configuration of parts, where parts are the same kind of thing as the whole and the parts cannot be separated from the whole. *Member–bunch composition* groups together molecules into collections when the collection members share similar functionality (e.g., elongation factors) or cellular location (e.g., membrane proteins). We have not found the other three composition relationships due to Odell to be relevant for our model.

We implemented our framework using the Protégé-2000 knowledge-modeling tool [16]. We used Protégé's axiom language (PAL) to define queries in a subset of first-order predicate logic written in the Knowledge Interchange Format syntax. The queries present, in tabular format, relationships among processes and structural components as well as the relationship between a defective process or clinical phenotype and the mutation that is causing it.

1.2.2. Translation into Petri Nets

We manually translated the tRNA workflow model into corresponding Petri Nets, according to mapping defined by others [12]. The Petri Net models that we used were high-level Petri Nets that allow the representation of hierarchy and data. Hierarchies enable expanding a transition in a given Petri Net to an entire Petri Net, as is done in expanding workflow high-level processes into a net of lower level processes. We upgraded the derived Petri Nets to Colored Petri Nets (CPNs) by:

1. Defining color sets for tRNA molecules (mutated and normal), mRNA molecules, and nucleotides that comprise the mRNA sequence and initiating the Petri Nets with an initial marking of colored tokens
2. Adding guards on transitions that relate to different types of tRNA molecules (e.g., fMet-tRNA vs. elongating tRNA molecules)
3. Defining mRNA sequences that serve as the template for translation

We used the *Woflan* Petri Net verification tool [17] to verify that the Petri Nets are bounded (i.e., no accumulation of an infinite amount of tokens) and live (i.e., deadlocks do not exist). To accommodate limitations in the Woflan tool, which does not support colored Petri Nets, we manually made several minor changes to the Petri Nets before verifying them. We simulated the Petri Nets to study the dynamic aspects of the translation process using the Design CPN tool [18], which has since been replaced by *CPN Tools*.

1.2.3. Sources of Biological Data

We gathered information from databases and published literature in order to develop the tRNA example considered in this work. We identified data sources with information pertaining to tRNA sequence, structure, modifications, mutations, and disease associations. The databases that we used were:

- Compilation of mammalian mitochondrial tRNA genes [19], aimed at defining typical as well as consensus primary and secondary structural features of mammalian mitochondrial tRNAs (http://mamit-trna.u-strasbg.fr/)
- Compilation of tRNA sequences and sequences of tRNA genes [20] (http://www.uni-bayreuth.de/departments/biochemie/sprinzl/trna/)
- The Comparative RNA website (http://www.rna.icmb.utexas.edu/), which provides a modeling environment for sequence and secondary-structure comparisons [21]
- Structural Classifications of RNA (SCOR, http://scor.lbl.gov/scor.html) [22]
- The RNA Modification Database (http://medlib.med.utah.edu/RNAmods), which provides literature and data on nucleotide modifications in RNA [23]
- A database on tRNA genes and molecules in mitochondria and photosynthetic eukaryotes (http://www.ba.itb.cnr.it/PLMItRNA/) [8]

- Online Mendelian Inheritance in Man (OMIM) (http://www.ncbi.nlm.nih.gov/omim/), which catalogs human genes and genetic disorders [24]
- BioCyc (http://metacyc.org/), a collection of genome and metabolic pathway databases which describes pathways, reactions, and enzymes of a variety of organisms [25]
- Entrez, the life sciences search engine, which provides views for a variety of genomes, complete chromosomes, contiged sequence maps, and integrated genetic and physical maps (http://www.ncbi.nlm.nih.gov/gquery/gquery.fcgi?itool = toolbar) [26]
- MITOMAP, A human mitochondrial genome database [27] (http://www.mitomap.org/)
- The UniProt/Swiss-Prot Protein Knowledgebase, which gives access to wealthy annotations and publicly available resources of protein information (http://us.expasy.org/sprot/sprot-top.html)

In addition, we used microarrays [28] and mass spectral data [29], providing information on proteins involved in tRNA processing or affected by tRNA mutations.

1.3. MODELING APPROACH AND RESULTS

Our model represents data using process diagrams and participant/role diagrams. Appendix A on our website (http://mis.hevra.haifa.ac.il/~morpeleg/NewProcessModel/Malaria_PN_Example_Files.html) presents the number of processes, participants, roles, and links that we used in our model. The most granular thing that we represented was at the level of a single nucleotide (e.g., GTP). The biggest molecule that we represented was the ribosome. We chose our levels of granularity in a way that considers the translation process under the assumption of a perfect ribosome; we only considered errors in translation that are due to tRNA. This assumption also influenced our design of the translation process model. This design follows individual tRNA molecules throughout the translation process and therefore represents the translocation of tRNA molecules from the P to the E site and from the A to the P site as distinct processes that occur in parallel. The level of detail in which we represented the model led us to consider questions such as (1) "Can tRNA bind the A site before previously bound tRNA molecule is released from the E site?" and (2) "Can fMet tRNA form a ternary complex?"

1.3.1. Representing Mutations

Variation in gene products (protein or RNA) can result from mutations in the nucleotide sequence of a gene, leading to altered (1) translation, (2) splicing, (3) posttranscriptional end processing, or (4) interactions with other cellular components coparticipating in biological processes. In addition, variation can result from a

normal sequence that is translated improperly by abnormal tRNA molecules. Thus, we must be able to represent variation not only in DNA sequences (genome) but also in RNA and protein. Therefore, in our ontology, every sequence component (of a nucleic acid or protein) may be associated with multiple alleles. Each allele may have mutations that are either pathogenic (associated with abnormal functions) or neutral. A mutation is classified as a substitution, insertion, or deletion [30].

1.3.2. Representing Nucleic Acid Structure

The TAMBIS terminology did not focus on three-dimensional structure. We extended the TAMBIS ontology by specifying tertiary-structure components of nucleic acids. A nucleic acid tertiary-structure component is composed of interacting segments of nucleic acid secondary-structure components. We added three types of nucleic acid secondary-structure components: nucleic acid helix, nucleic acid loop, and nucleic acid unpaired strand. Figure 1.1 shows the tertiary-structure components of tRNA (acceptor domain, D domain, T domain, variable loop, and anticodon domain). Also shown is the nucleic acid tertiary-structure component frame that corresponds to the tRNA acceptor domain. The division of tRNA into structural domains, the numbering of nucleotides of the generic tRNA molecule, and the sequence-to-structure correspondence was done according to conventional rules [20].

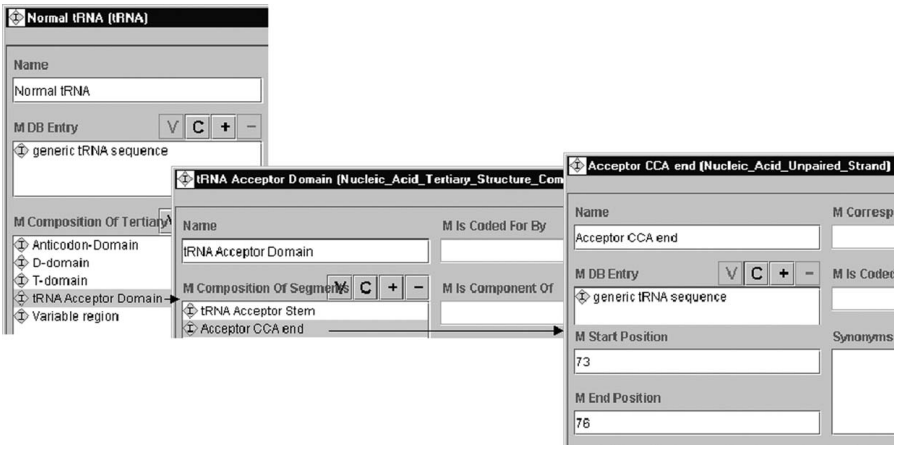


FIGURE 1.1. Tertiary-structure components. Normal tRNA is composed of five nucleic acid tertiary-structure components. One of these components (tRNA acceptor domain) is shown in the middle frame. Each nucleic acid tertiary-structure component is composed of segments of nucleic acid secondary-structure components. The nucleic acid unpaired strand of the tRNA acceptor domain, which is a kind of nucleic acid secondary-structure component, is shown on the right.

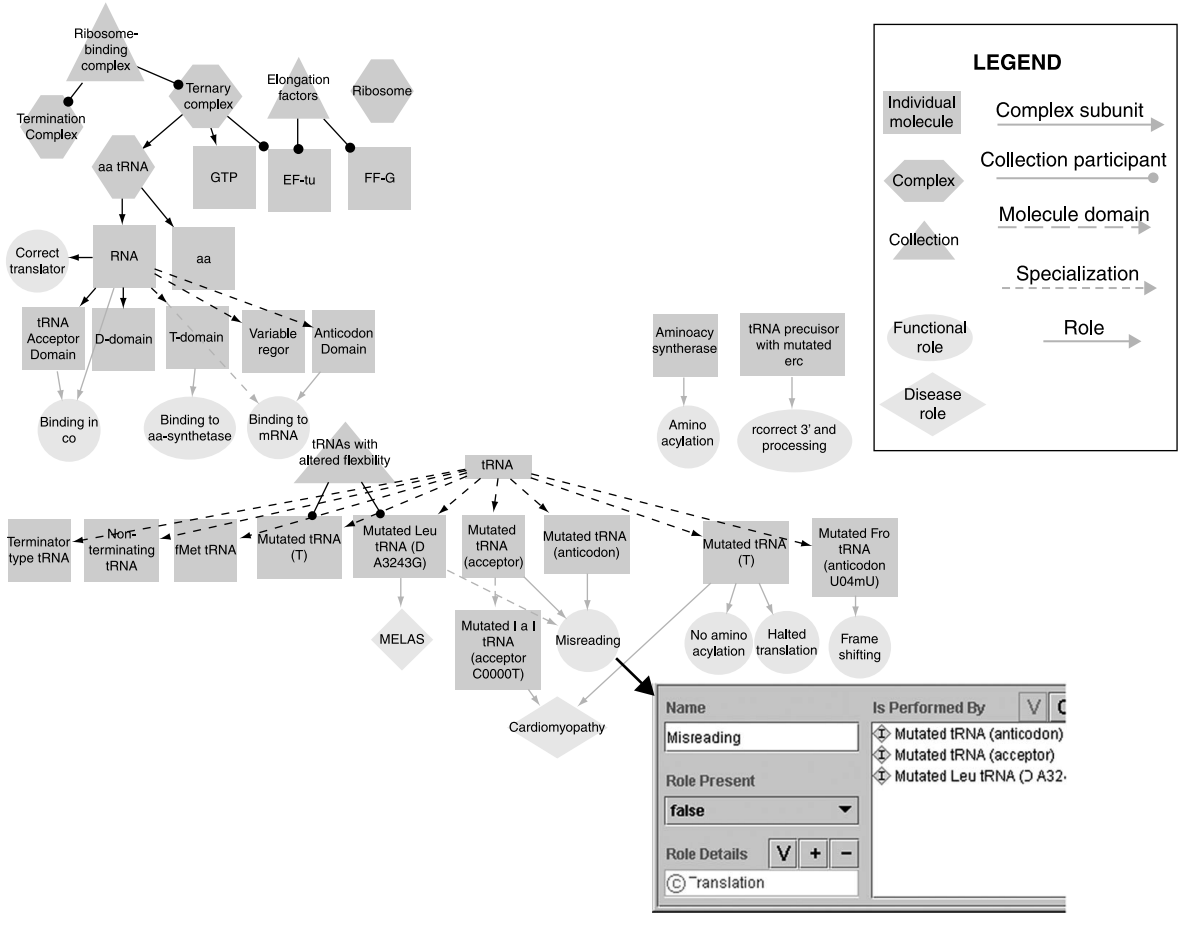

Ribosome-binding complex
Ternary complex
Elongation factors
Ribosome
Termination Complex
aa tRNA
GTP
EF-tu
FF-G
Correct translator
RNA
aa
tRNA Acceptor Domain
D-domain
T-domain
Variable regor
Anticodon Domain
Binding in co
Binding to aa-synthetase
Binding to mRNA
tRNAs with altered flexibility
tRNA
Aminoacy synthetase
tRNA precuisor with mutated erc
Amino acylation
rcorrect 3' and processing
LEGEND
Individual molecule
Complex subunit
Collection participant
Complex
Molecule domain
Collection
Specialization
Functional role
Role
Disease role
Terminator type tRNA
Non-terminating tRNA
fMet tRNA
Mutated tRNA (T)
Mutated Leu tRNA (D A3243G)
Mutated tRNA (acceptor)
Mutated tRNA (anticodon)
Mutated tRNA (T)
Mutated Fro tRNA (anticodon U04mU)
MELAS
Mutated I a l tRNA (acceptor C0000T)
Misreading
No amino acylation
Halted translation
Frame shifting
Cardiomyopathy
Name
Misreading
Role Present
false
Role Details
Translation
Is Performed By
Mutated tRNA (anticodon)
Mutated tRNA (acceptor)
Mutated Leu tRNA (D A32

FIGURE 1.2.

1.3.3. Representing Molecular Complexes

Biological function can be associated with different levels of molecular structure. In some cases a function can be associated with a domain (of a protein or nucleic acid). In other cases, a function is associated with individual molecules or with molecular complexes. Sometimes, a function is not specifically mapped to a molecular structure but is attributed to collections of molecules that are located in a particular cellular compartment. In addition, biologists define collections of molecules that share a common function (e.g., termination factors). The participant/role representation of our framework represents molecular structures that participate in processes as well as composition and generalization relationships among participants (molecules).

In our tRNA example, we are using three kinds of these composition relationships: (1) component–integral object composition, (2) member–bunch composition, and (3) place–area composition. Figure 1.2 shows examples of these relationships. Generalization (is-a) relationships are used to relate subclasses of participants to their superclasses. For example, terminator tRNA, nonterminating tRNA, and fMet tRNA are subclasses of the tRNA class.

1.3.4. Representing Abnormal Functions and Processes

In addition to representing relationships among process participants, our framework can represent the roles that participants have in a modeled system. We distinguish two types of roles: molecular-level functional roles (e.g., a role in translation) and roles in clinical disorders (e.g., the cause of cardiomyopathy). Each role is specified using a function/process code taken from the TAMBIS ontology. To represent dysfunctional molecular-level roles, we use an attribute, called role_present, which signifies whether the role is present or absent or this information is unknown. For example, Figure 1.2 shows that three mutations of tRNA that exhibit the role of misreading. The figure also shows tRNA mutations that have roles in the cardiomyopathy disorder. Cardiomyopathy is one of the concepts from the clinical ontology, discussed later in this section.

FIGURE 1.2. Part of participant/role diagram showing molecules involved in translation and roles they fulfill. Individual molecules are shown as rectangles (e.g., tRNA). They are linked to domains (e.g., D domain) using dashed connectors. Biomolecular complexes are shown as hexagons (e.g., ternary complex) and linked to their component molecules using arrowhead connectors. Collections of molecules that share similar function or cellular location are shown as triangles (e.g., elongation factors) and are linked to the participants that belong to them using connectors with round heads. Generalization relationships are shown as dotted lines (e.g., fMet-tRNA is-a tRNA). Functional roles are shown as ellipses that are linked to the participants that exhibit those roles. Clinical disorders that are associated with mutated participants are shown as diamonds (e.g., cardiomyopathy) and are linked to the participants that exhibit roles in these disorders. The insert shows the details of the misreading role. It is specified as a translation role (TAMBIS class) that is not present (role_present = false). Also shown are some of the participants that perform the misreading role.

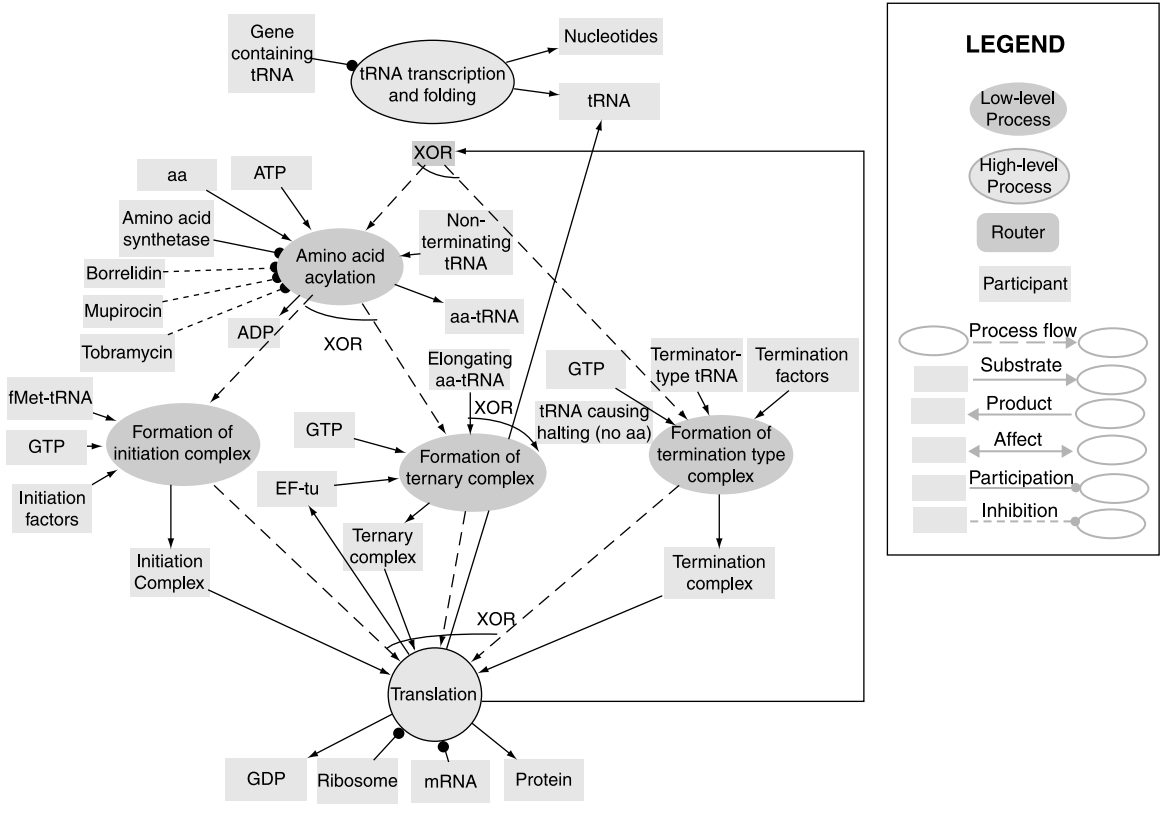

Gene containing tRNA
tRNA transcription and folding
Nucleotides
tRNA
XOR
aa
ATP
Amino acid synthetase
Amino acid acylation
Non-terminating tRNA
Borrelidin
Mupirocin
Tobramycin
ADP
XOR
aa-tRNA
Elongating aa-tRNA
XOR
GTP
Terminator-type tRNA
Termination factors
tRNA causing halting (no aa)
fMet-tRNA
GTP
Initiation factors
Formation of initiation complex
GTP
EF-tu
Formation of ternary complex
Formation of termination type complex
Ternary complex
Initiation Complex
Termination complex
XOR
Translation
GDP
Ribosome
mRNA
Protein
LEGEND
Low-level Process
High-level Process
Router
Participant
Process flow
Substrate
Product
Affect
Participation
Inhibition

FIGURE 1.3.

Processes are represented using the process model component of our framework. We augmented the workflow model with elements taken from the object process methodology (OPM) [31] to create a graphical representation of the relationships between a process and the static components that participate in it, as shown in Figure 1.3. We used different connectors to connect a process to its input sources, output sources, and participants that do not serve as substrates or products (e.g., catalysts such as amino acid synthetase). We added a fourth type of connector that links a process to a chemical that inhibits the process (e.g., borrelidin). Figures 1.3 through 1.6 present details of the translation process and the processes leading to it. The figures show the normal process as well as processes that result in abnormal translation. We have considered only tRNA-related failures of translation. Detailed explanation of each process diagram is given in the legends. Figures 1.4 and 1.5 present the details of the translation process, depicted in Figure 1.3. Figure 1.4 presents the translation process according to the classical two-site model [5]. Figure 1.5 presents a recent model of the translation process [32]. The details of the process of tRNA binding to the A site, of Figure 1.5, are shown in Figure 1.6.

The processes normal reading, misreading, frame shifting, and halting, shown in Figure 1.6, all have a process code of binding, since in all of them tRNA binds to ribosome that has occupied E and P sites.

The types of arrows that connect molecules to a process define their role as substrates, products, inhibitors, activators, or molecules that participate without changing their overall state in the framework (e.g., enzyme). The logical relationships among participants are specified in a formal expression language. For example, double-clicking on the misreading process, shown in Figure 1.6, shows its participants, which are specified as

```
(Shine–Delgarno in E XOR tRNA0 in E) AND tRNA1 in P AND
(tRNA2 that can bind to incorrect codon in ternary
complex XOR tRNA that has altered flexibility in
ternary complex) AND tRNA2 in A AND EF-tu AND GDP
```

FIGURE 1.3. Process diagram showing processes leading to translation. Ellipses represent activities. Ellipses with bold contours represent high-level processes, whereas ellipses without bold contours represent low-level processes (that are not further expanded). The dark rounded rectangles represent routing activities for representing logical relationships among component activities of a process diagram. The router (checkpoint) labeled XOR represents a XOR split that signifies that the two processes that it connects to are mutually exclusive. A XOR join connects the three processes shown in the middle of the diagram to the translation process. Dotted arrows that link two activities to each other represent order relationships. Participants are shown as light rectangles. Arrows that point from a participant toward a process specify that the participant is a substrate. Arrows that point in the opposite direction specify products. Connectors that connect participants (e.g., amino acid synthetase) to processes and have a circle head represent participation that does not change the state of the participant. Inhibitors (e.g., tobramycin) are linked to processes via a dashed connector. The details of the translation process are shown in Figures 1.4 and 1.5.

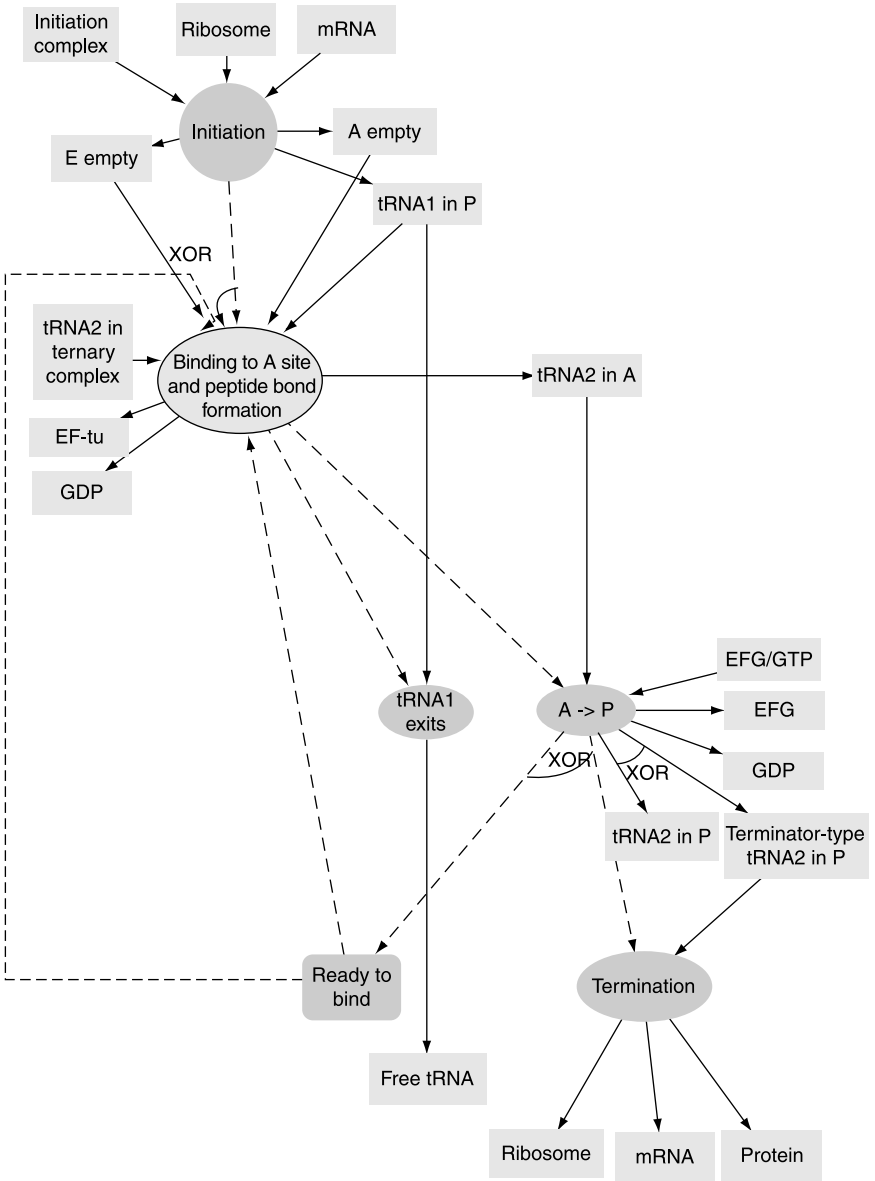


FIGURE 1.4. Process diagram showing details of translation process of Figure 1.4 according to classical two-site model [5]. The symbols are as explained in the legend of Figure 1.3. After initiation, there is an aa-tRNA in the P site (tRNA1 in P). During the process labeled "binding to A site and peptide bond formation" a second aa-tRNA in the ternary complex binds to the A site. Two processes occur simultaneously at the next stage: movement of the second tRNA that bound to the A site to the P site and, at the same time, exit from the ribosome of the first tRNA that bound to the P site. If the second tRNA, bound to the P site, is of terminator type, termination occurs. Otherwise, the ribosome is ready to bind; the second tRNA to bind tRNA is now labeled as "tRNA1 in P" and another cycle of elongation can begin.

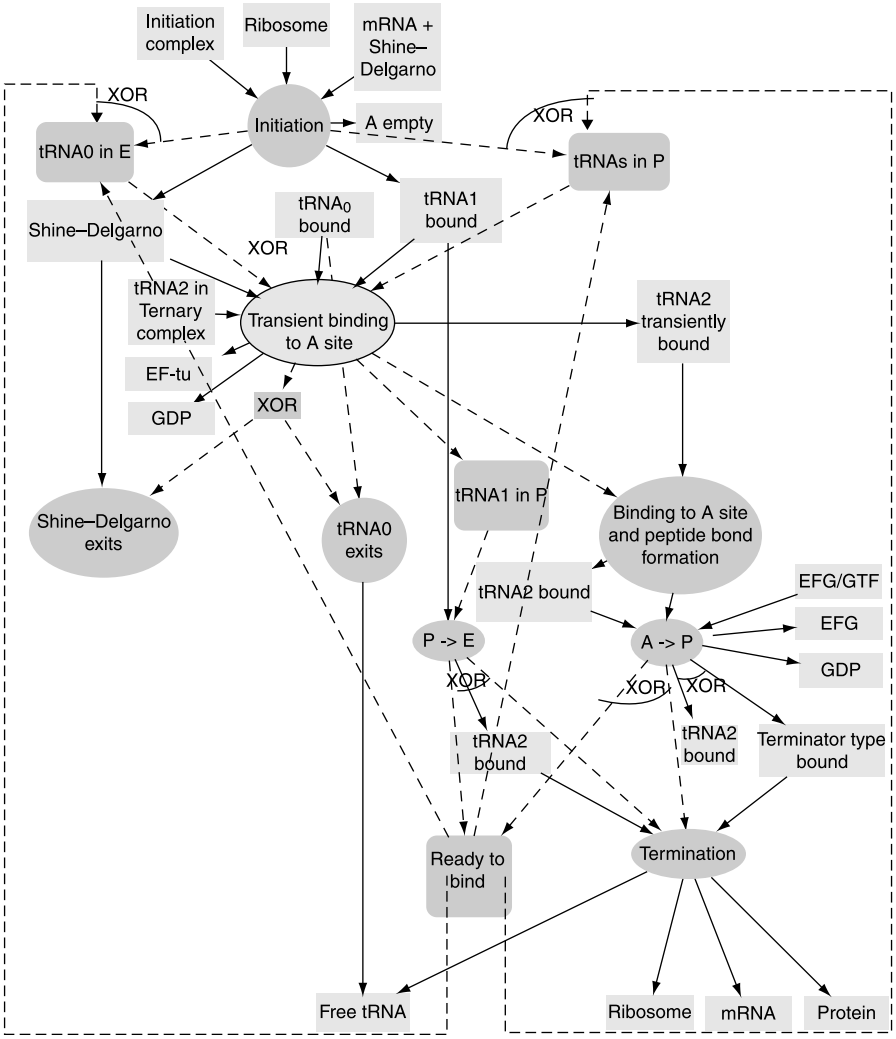


FIGURE 1.5. Process diagram showing details of the translation process of Figure 1.4 according to model of Connell and Nierahus [32]. The details of the process labeled "binding of tRNA to A site" are shown in Figure 1.6. After initiation, shine dalgarno is placed at the E site, and the first tRNA (tRNA1) is placed at the P site. Next, tRNA2 transiently binds to the A site. This step is followed by three activities which are done concurrently: (1) exit from the E site of either Shine–Delgarno or tRNA0 bound to the E site (at later stages of the elongation process), (2) binding to the A site followed by peptide bond formation, and (3) a routing activity (marked by an unlabeled round-corner square). The routing activity is needed for correspondence with the CPN that simulates the translation process, which needs to distinguish among the tRNA molecules that are bound to each of the three sites. At the next stage, tRNA2 at the A site shifts to the P site and at the same time, tRNA1 at the P site shifts to the E site. If tRNA2 bound to the P site is of terminator type, termination occurs. Otherwise, the ribosome is ready to bind; the second tRNA to bind is now labeled as "bound tRNA1," and the first tRNA to bind is labeled as "bound tRNA0," and another cycle of elongation can begin.

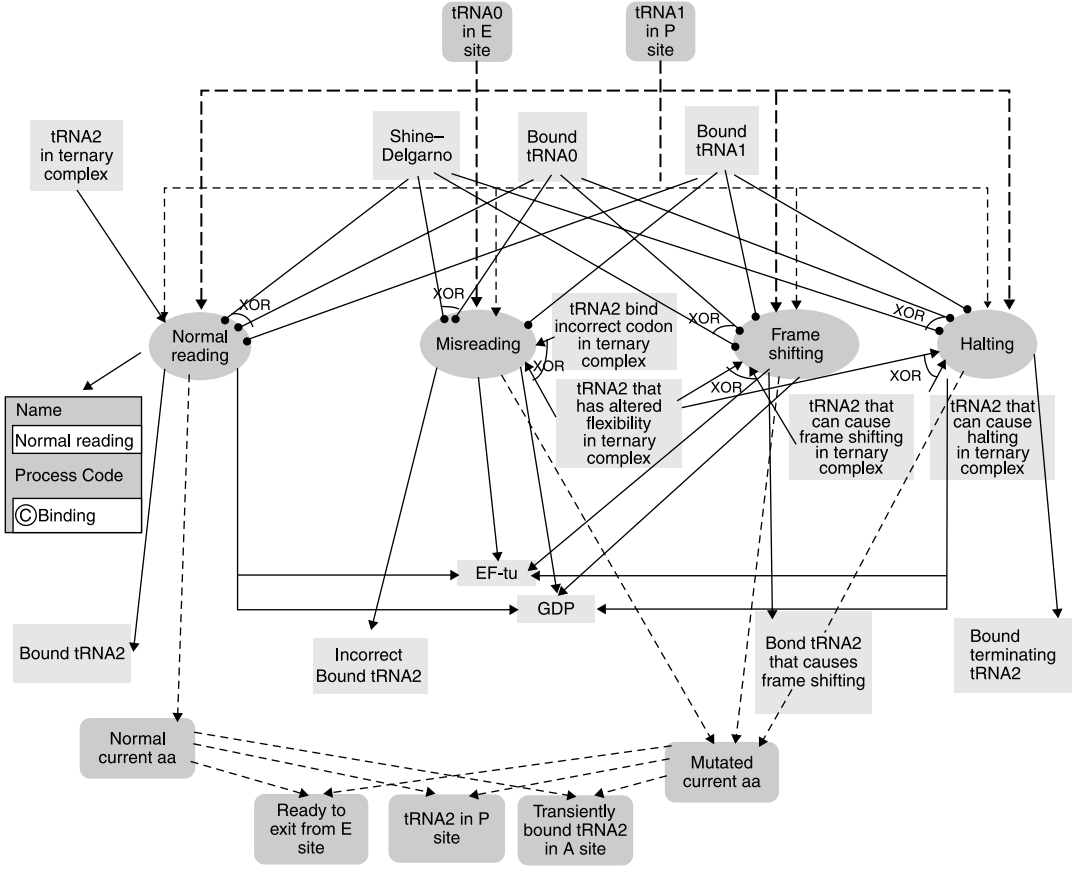


FIGURE 1.6. Process diagram showing normal and abnormal reading processes. Starting from a ribosome with tRNA in both E and P sites, four alternative processes can lead to ribosome with the A site occupied with tRNA: normal reading, misreading, frame shifting, and halting. Symbols are as explained in the legend of Figure 1.3. The insert shows the process code (binding) of the reading process, taken from the TAMBIS ontology.

1.3.5. Representing High-Level Clinical Phenotypes

Our clinical ontology relies on the UMLS but does not include all of the concepts of the Metathesaurus. Instead, we are building our clinical ontology by importing concepts as we need them. We add clinical concepts to the clinical ontology by creating them as subclasses of the semantic types defined by the semantic network. Each concept has a concept name and a concept code that come from the Metathesaurus as well as synonyms. Figure 1.7 shows part of the clinical ontology. Figure 1.3 shows that mutated leucine tRNA (in the tRNA acceptor domain) and mutated tRNA (in the T domain) have roles in some forms of cardiomyopathy. Many tRNA-related diseases are also linked to mutations in protein components of mitochondrial respiratory chains. Proteomic studies in [28] provide a larger list of protein candidates. Twenty identified proteins are shown to either overproduce (9) or be underrepresented (11) when the mitochondrial genome has the A8344G mutation (in $tRNA^{Lys}$) associated with myoclonic epilepsy and ragged red fibers (MERFF) condition.

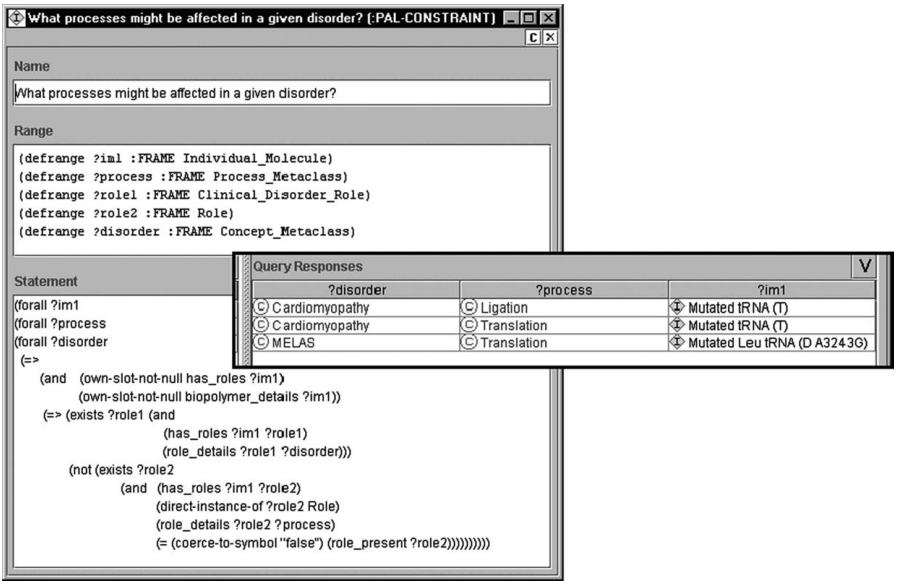


FIGURE 1.7. Query that shows individual molecules involved in both disorders and dysfunctional processes. The results of this query may indicate which processes are involved in a given disorder. The query is shown on the left. The results are shown on the right. The frames ?im1 ?process, ?role1, ?role2, and ?disorder represent individual molecules, processes, roles, and disorders, respectively. The query is written as a constraint. Instances that violate the constraint are returned. The predicate (*own-slot-not-null A B*) returns true if slot A of frame B is not null. The constraint looks for all individual molecules, which (1) have roles that are disorders and (2) have roles that are dysfunctional processes or functions.

1.3.6. Representing Levels of Evidence for Modeled Facts

Different facts that are represented in our framework are supported by varying degrees of evidence. It is important to allow users to know what support different facts have, especially in cases of conflicting information. We therefore added a categorization of evidence according to the type of experimentation by which facts were established. The categorization includes broad categories, such as "in vivo," "in vitro," "in situ," "in culture," "inferred from other species," and "speculative." Facts, such as the existence of a biomolecule or its involvement in a process are tagged with the evidence categories.

1.3.7. Querying the Model

Using PAL we composed first-order logic queries that represent in tabular form relationships among processes and structural components. Table 1.1 shows a

TABLE 1.1 Types of Biological Queries and Motivating Biological Examples

Query Type	Example	Derived Answer from Model
1. Alleles		
1.1 Alleles that have roles in dysfunctional processes and/or disorders	Alleles that have roles in both dysfunctional processes and disorders	Mutated tRNA (T) causes cardiomyopathy and has roles in amino acylation + halted translation Mutated Leu tRNA (D) causes mitochondrial myopathy encephalopathy lactocidosis stroke (MELAS) and has a role in misreading
2. Roles		
2.1 Individual molecules or biocomplexes that have the same role Scoped to cellular location, same substrates and products, same biological process (participation), or to same (or different) inhibitor	Individual molecules that have the same set of roles	Mutated tRNA (anticodon) and mutated tRNA (acceptor) both have only the role of misreading
	Individual molecules that have a role in a dysfunctional process	Incorrect translation: mutated tRNA (anticodon), mutated tRNA (T), mutated Pro tRNA (anticodon U34mU), mutated Leu tRNA (D A3243G), mutated tRNA (acceptor) Incorrect ligation: mutated tRNA (T) Incorrect processing: tRNA precursor with mutated 3′ end
	Individual molecules that have a role in a disorder	Cardiomyopathy: mutated tRNA (T), mutated Leu tRNA (acceptor) MELAS: mutated Leu tRNA (D)

(*continued*)

TABLE 1.1 *Continued*

Query Type	Example	Derived Answer from Model
3. Reaction (functional model)		
3.1 All atomic activities that share the same substrates (products, inhibitors)	What atomic activities have the same substrates and products?	None in the modeled system
4. Biological process		
4.1 All activities of a certain kind of biological process, according to the TAMBIS classification hierarchy (scoped to cellular location substrates)	All activities that are a kind of binding and involve binding of tRNA	Formation of ternary complex, formation of initiation complex, formation of termination complex, binding to A site, normal reading, misreading, halting, frame shifting
4.2 All activities that are inhibited by inhibitor x	Activities inhibited by tobramycin and mupirocin	Amino acid acylation
4.3 What processes might be affected in a given disorder?	What processes might be affected in a given disorder?	Amino acid acylation and translation (reading) are affected in cardiomyopathy Translation (reading) is affected in cardiomyopathy
5. Reachability		
5.1 If an activity is inhibited what other activities can take place? Is it a deadlock?	Inhibiting "normal reading" (no supply of normal tRNA): what activities may take place?	Directly in XOR: misreading, frame shifting, halting
5.2 If an activity is inhibited, can we still get to a specified state?	If we inhibit "formation of ternary complex," can we reach a state where the activity "termination" is enabled?	Yes. For example, the firing sequence $t_1t_2t_4t_1t_2t_5t_6t_7t_8t_{10}t_{11}$ $t_{12}t_{13}$ of Figure 1.8
5.3 Does an inhibitor inhibit an entire high-level process?	Does tobramycin inhibit the translation process?	Yes. It inhibits the process "formation of initiation complex" which is essential to take place before translation
5.4 Establish a marking, find reachability	Elongating tRNA is a substrate. What pathways will be taken?	Amino acid acylation, followed by formation of ternary complex, followed by translation
6. Temporal/dynamic aspects		
6.1 What other processes occur in parallel to process X?	What processes occur in parallel to "binding to A site" (Fig. 1.6)	"Shine–Delgarno exits" XOR "tRNA1 exits"

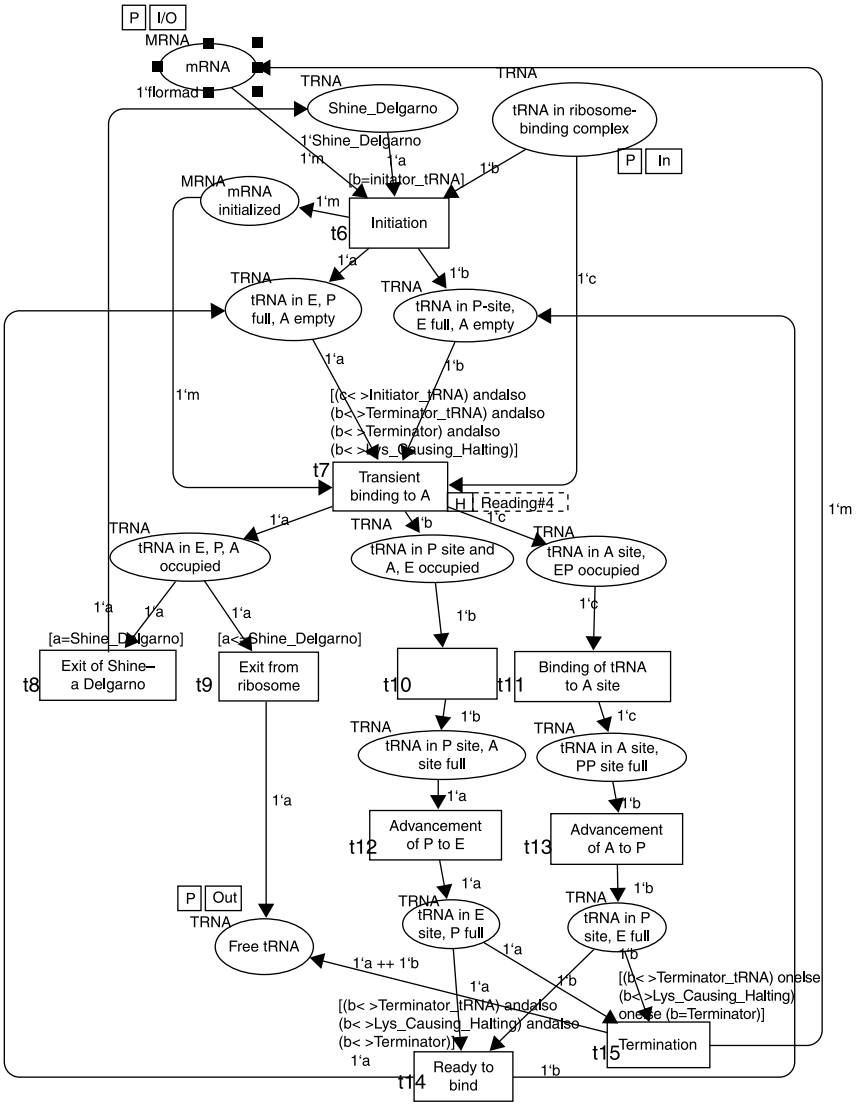


FIGURE 1.8. Colored Petri Net that corresponds to Figure 1.5 showing current three-site model of translation. Squares represent transitions, corresponding to workflow processes. Ellipses represent places, corresponding to conditions that are true after a workflow process has terminated. Text to the top left of places indicates their allowed token type, which can be tRNA or mRNA. The values of tokens of tRNA type used in this figure are Shine_Delgarno, Initiator_tRNA, Terminator_tRNA, Terminator, and Lys_Causing_Halting. Other token types that we use in our model (not shown) represent other mutations of tRNA molecules. The values of tokens of mRNA type are always "normal." Text below places specifies initial placement of tokens in those places. Text above transitions indicates guarding conditions, which refer to token types. Text on connectors indicates token variables that flow on those connectors. The variables used are a, b, and c for tRNA tokens and *m* for mRNA tokens. Transitions are also labeled t6, . . . , t15, in correspondence with query 5.2 of Table 1.1.

summary of all the query types that we composed. They are grouped into six categories that concern (1) alleles, (2) functional roles and roles in disorder phenotypes, (3) reactions and their participants, (4) biological processes, (5) ability to reach a certain state of a modeled system, and (6) temporal/dynamic aspects of a modeled system. Queries that were especially interesting to us were (1) finding mutations that cause molecular-level processes and functions to be dysfunctional, (2) finding mutations that cause clinical disorders, and (3) finding processes that might be affected in a given disorder. Figure 1.7 shows the query and query results for the third query.

1.3.8. Simulating the Model

As shown in Figures 1.4 and 1.5, we created two different models of the translation process: a historical model and a current model. When we translated the workflow models into the corresponding Petri Nets, we were able to test predictions of these two models by showing that under certain concentrations of reactants the different models resulted in different dynamic behavior which produced different translation products. For example, when the mRNA contained a sequence of Asn–Leu–Asn (or in general, aa_1–aa_2–aa_1) and the system was initialized with a low concentration of Asn-tRNA, then protein translation proceeded in the classical two-site model but was halted in the current three-site model, which required Asn-tRNA and Leu-tRNA to be bound to the ribosome while a second Asn-tRNA bound the A site. The Petri Net that corresponds to the workflow model of Figure 1.5 is shown in Figure 1.8. The tRNA mutations were represented as colored tokens, belonging to the tRNA color set (see Fig. 1.8), and mRNA molecules were represented as tokens belonging to the mRNA color set.

The Petri nets derived from our workflow model can also be used for educational purposes. They can demonstrate (1) concurrent execution of low-level processes within the translation process (e.g., tRNA molecules that were incorporated into synthesized proteins can be amino acylated and used again in the translation process), (2) introduction of mutations into synthesized proteins, and (3) the affect of certain dysfunctional components on pools of reactants (e.g., nonmutated tRNAs).

1.4. DISCUSSION

Deducing molecular mechanisms of disease based on molecular models is a very difficult problem. Even more complicated is the task of correlating genotypic variation to clinical phenotypes. A review by Florentz and Sissler [33] shows that, despite the accumulation of information about the positions of a large number of mutations within mitochondrial tRNAs, it is not possible to identify simple basic patterns for use in predicting the pathogenicity of new mutations. The multifaceted nature of effects produced by tRNA mutations is apparent from recent proteomics studies [29] and is emphasized in current reviews [34, 35]. The authors conclude that it is critical to examine not only the affected tRNA but also its interactions, or relationships, with other compartmental components. These arguments

emphasize the importance of a knowledge model able to integrate practical information at multiple levels of detail and from multiple experimental sources.

The knowledge framework presented here links genetic sequence, structure, and local behavior to high-level biological processes (such as disease). The model provides a mechanism for integrating data from multiple sources. In our tRNA example, we integrated information from structural biology, genetics and genomics, molecular biology, proteomics, and clinical science. The information can be presented graphically as process diagrams or participant/role diagrams. The frames that represent participants, roles, processes, and relationships among them contain citations to the original data sources.

Our model has several advantages, in addition to its ability to integrate data from different sources. First, we can define queries that create views of the model in a tabular format. The queries extract useful relationships among structures, sequences, roles, processes, and clinical phenotypes. Second, our model can be mapped in a straightforward manner to Petri Nets. We developed software that automatically translates our biological process model into Petri Net formalisms and formats used by various Petri Net tools [36]. We have used available tools to qualitatively simulate a modeled system and to verify its boundedness and liveness and to answer a set of biological questions that we defined [36]. *Boundedness* assumes that there is no infinite accumulation of tokens in any system state. In our example, this corresponds to concentration of tRNA and mRNA molecules in a cell. *Liveness* ensures that all Petri Net transitions (which correspond to workflow activities) can be traversed (enabled).

A disadvantage of our model is its need for manual data entry. Natural language processing techniques are not able to automatically parse scientific papers into the semantic structure of our ontology. The effort required to enter data into our model is considerable. The entry of a substantial set of data about all relevant cellular reactions and processes would require a major distributed effort by investigators trained in knowledge representation and biology.

1.5. CONCLUSION

One of the ultimate goals of proteomics and genomics engineering is to develop a model of the real cell, of its program responsible for different behaviors in various intra- and extracellular environments. Our long-term goal is to develop a robust knowledge framework that is detailed enough to represent the phenotypic effects of genomic mutations. The results presented here are a first step in which we demonstrate that the knowledge model developed in another context (malaria invasion biology) is capable of capturing a qualitative model of tRNA function. We have presented a graphical knowledge model for linking genetic sequence polymorphisms to their structural, functional, and dynamic/behavioral consequences, including disease phenotypes. We have shown that the resulting qualitative model can be queried (1) to represent the compositional properties of the molecular ensembles, (2) to represent the ways in which abnormal processes can result from

structural variants, and (3) to represent the molecular details associated with high-level physiological and clinical phenomena. By translating the workflow representation into Petri Nets we were able to verify boundedness and liveness. Using simulation tools, we showed that the Petri Nets derived from the historic and current views of the translation process yield different dynamic behavior.

ACKNOWLEDGMENTS

The work was funded by the Burroughs-Wellcome Fund and by National Institutes of Health grants LM-05652 and LM-06422.

REFERENCES

1. M. Peleg, I. Yeh, and R. B. Altman, "Modeling biological processes using Workflow and Petri Net models," *Bioinformatics*, 18: 825–837, 2002.
2. T. R. Gruber, "Toward principles for the design of ontologies used for knowledge sharing," *Int. J. Human-Computer Stud.*, 43: 907–928, 1995.
3. S. Schulze-Kremer, "Ontologies for molecular biology," paper presented at the Proceedings of the Third Pacific Symposium on Biocomputing, Hawaii, 1998.
4. M. Ibba, C. Stathopoulos, and D. Soll, "Protein synthesis: Twenty three amino acids and counting," *Curr Biol.*, 11: R563–565, 2001.
5. K. H. Nierahus, "New aspects of the ribosomal elongation cycle," *Mol. Cell Biochem.*, 61: 63–81, 1984.
6. D. N. Wilson, G. Blaha, S. R. Conell, P. V. Ivanov, H. Jenke, U. Stelzl, Y. Teraoka, and K. H. Nierahus, "Protein synthesis at atomic resolution: Mechanistics of translation in the light of highly resolved structures for the ribosome," *Curr. Protein Peptide Sci.*, 3: 1–53, 2002.
7. P. J. Farabaugh and G. R. Bjork, "How translational accuracy influences reading frame maintenance," *EMBO J.*, 18: 1427–1434, 1999.
8. V. Volpetti, R. Gallerani, C. D. Benedetto, S. Liuni, F. Licciulli, and L. R. Ceci, "PLMItRNA, a database on the heterogenous genetic origin of mitochondrial tRNa genes and tRNAs in photosynthetic eukaryotes," *Nucleic Acids Res.*, 31: 436–438, 2003.
9. M. Peleg, I. S. Gabashvili, and R. B. Altman, "Qualitative models of molecular function: Linking genetic polymorphisms of tRNA to their functional sequelae," *Proc. IEEE*, 90: 1875–1886, 2002.
10. L. Fisher, *Workflow Handbook*, published in association with the Workflow Management Coalition, Future Strategies, Lighthouse Point, FL, 2001.
11. J. L. Peterson, *Petri Net Theory and the Modeling of Systems*, Prentice-Hall, Englewood Cliffs, NJ, 1981.
12. W. M. P. v. d. Aalst, "The application of Petri Nets to workflow management," *J. Circuits, Syst. Computers*, 8: 21–66, 1998.
13. P. G. Baker, C. A. Goble, S. Bechhofer, N. W. Paton, R. Stevens, and A. Brass, "An ontology for bioinformatics applications," *Bioinformatics*, 15: 510–520, 1999.
14. C. Lindberg, "The Unified Medical Language System (UMLS) of the National Library of Medicine," *J. Am. Med. Rec. Assoc.*, 61: 40–42, 1990.

15. J. Odell, "Six different kinds of composition," *J. Object-Oriented Prog.*, 7: 10–15, 1994.
16. S. W. Tu and M. A. Musen, "Modeling data and knowledge in the EON guideline architecture," paper presented at Medinfo, London, 2001.
17. H. M. W. Verbeek, T. Basten, and W. M. P. v. d. Aalst, "Diagnosing workflow processes using Woflan," *Computer J.*, 44: 246–279, 2001.
18. D. CPN group at the University of Aarhus, "Design/CPN—Computer Tool for Coloured Petri Nets," http://www.daimi.au.dk/designCPN/, 2002.
19. M. Helm, H. Brule, D. Friede, R. Giege, D. Putz, and C. Florentz, "Search for characteristic structural features of mammalian mitochondrial tRNAs," *RNA*, 6: 1356–1379, 2000.
20. M. Sprinzl and K. S.Vassilenko, "Compilation of tRNA sequences and sequences of tRNA genes," *Nucleic Acids Res.*, 33: D135–D138, 2005.
21. J. Cannone, S. Subramanian, M. N. Schnare, J. R. Collett, L. M. D'Souza, Y. Du, B. Feng, N. Lin, L. V. Madabusi, K. M. Muller, N. Pande, Z. Shang, N. Yu, and R. R. Gutell, "The Comparative RNA Web (CRW) site: An online database of comparative sequence and structure information for ribosomal, intron, and other RNAs," *BMC Bioinformatics*, 3: 2, 2002.
22. P. S. Klosterman, M. Tamura, S. R. Holbrook, and S. E. Brenner, "SCOR: A structural classification of RNA database," *Nucleic Acids Res.*, 30: 392–394, 2002.
23. P. A. Limbach, P. F. Crain, and J. A. McCloskey, "Summary: The modified nucleosides of RNA," *Nucleic Acids Res.*, 22: 2183–2196, 1994.
24. "Online Mendelian Inheritance in Man, OMIM (TM)," McKusick-Nathans Institute for Genetic Medicine, Johns Hopkins University (Baltimore, MD) and National Center for Biotechnology Information, National Library of Medicine (Bethesda, MD), http://www.ncbi.nlm.nih.gov/omim/, 2000.
25. P. D. Karp, C. A. Ouzounis, C. Moore-Kochlacs, L. Goldovsky, P. Kaipa, D. Ahrén, S. Tsoka, N. Darzentas, V. Kunin, and N. López-Bigas, "Expansion of the BioCyc collection of pathway/genome databases to 160 genomes," *Nucleic Acids Res.*, 33: 6083–6089, 2005.
26. D. L. Wheeler, T. Barrett, D. A. Benson., S. H. Bryant, K. Canese, D. M. Church, M. DiCuccio, R. Edgar, S. Federhen, W. Helmberg, D. L. Kenton, O. Khovayko, D. J. Lipman, T. L. Madden, D. R. Maglott, J. Ostell, J. U. Pontius, K. D. Pruitt, G. D. Schuler, L. M. Schrim, E. Sequeira, S. T. Sherry, K. Sirotkin, G. Starchenko, T. O. Suzek, R. Tatusov, T. A. Tatusova, L. Wagner, and E. Yaschenko, "Database resources of the National Center for Biotechnology Information," *Nucleic Acids Res.*, 33: D39–D45, 2005.
27. M. C. Brandon, M. T., Lott, K. C. Nguyen, S. Spolim, S. B. Navathe, P. Baldi, and D. C. Wallace, "MITOMAP: A human mitochondrial genome database—2004 update," *Nucleic Acids Res.*, 33: D611–613, 2005.
28. W. T. Peng, M. D. Robinson, S. Mnaimneh, N. J. Krogan, G. Cagney, Q. Morris, A. P. Davierwala, J. Grigull, X. Yang, W. Zhang, N. Mitsakakis, O. W. Ryan, N. Datta, V. Jojic, C. Pal, V. Canadien, D. Richards, B. Beattie, L. F. Wu, S. J. Altschuler, S. Roweis, B. J. Frey, A. Emili, J. F. Greenblatt, and T. R. Hughes, "A panoramic view of yeast noncoding RNA processing," *Cell*, 113: 919–933, 2003.
29. P. Tryoen-Toth, S. Richert, B. Sohm, M. Mine, C. Marsac, A. V. Dorsselaer, E. Leize, and C. Florentz, "Proteomic consequences of a human mitochondrial tRNA mutation beyond the frame of mitochondrial translation," *J. Biol. Chem.*, 278: 24314–24323, 2003.

30. V. Giudicelli and M. P. Lefranc, "Ontology for immunogenetics: The IMGT-ONTOLOGY," *Bioinformatics*, 15: 1047–1054, 1099.
31. D. Dori, "Object-process analysis: Maintaining the balance between system structure and behavior," *J. Logic Computation*, 5: 227–249, 1995.
32. S. Connell and K. Nierahus, "Translational termination not yet at its end," *Chembiochem*, 1: 250–253, 2000.
33. C. Florentz and M. Sissler, "Disease-related versus polymorphic mutations in human mitochondrial tRNAs: Where is the difference?," *EMBO Reps.*, 2: 481–486, 2001.
34. H. T. Jacobs, "Disorders of mitochondrial protein synthesis," *Hum. Mol. Genet.*, 12: R293–R301, 2003.
35. L. M. Wittenhagen and S. O. Kelley, "Impact of disease-related mitochondrial mutations on tRNA structure and function," *Trends Biochem. Sci.*, 28: 605–611, 2003.
36. M. Peleg, D. Rubin, and R. B. Altman, "Using Petri Net tools to study properties and dynamics of biological systems," *J. Am. Med. Inform. Assoc.*, 12(2): 181–199, 2005.

CHAPTER 2

Interpreting Microarray Data and Related Applications Using Nonlinear System Identification

MICHAEL KORENBERG

2.1. INTRODUCTION

We begin by considering some methods of building a model for approximating the behavior of a nonlinear system, given only the system inputs and outputs gathered experimentally. Such methods are sometimes referred to as "blackbox" approaches to nonlinear system identification, because they build a mimetic model of the input–output relation without assuming detailed knowledge of the underlying mechanisms by which the system actually converts inputs into outputs. Then we show that such approaches are well suited to building effective classifiers of certain biological data, such as for determining the structure/function family of a protein from its amino acid sequence, to detecting coding regions on deoxyribonucleic acid (DNA), and to interpreting microarray data. We concentrate on the latter application and in particular on predicting treatment response and clinical outcome and metastatic status of primary tumors from gene expression profiles. It is shown that one advantage of applying such nonlinear system identification approaches is to reduce the amount of training data required to build effective classifiers. Next, we briefly consider a means of comparing the performance of rival predictors over the same test set, so as to highlight differences between the predictors. We conclude with some remarks about the use of fast orthogonal search (FOS) in system identification and training of neural networks.

2.2. BACKGROUND

The field of nonlinear system identification is vast; here we confine ourselves to methods that yield mimetic models of a particular nonlinear system's behavior,

Genomics and Proteomics Engineering in Medicine and Biology. Edited by Metin Akay

given only access to the system inputs and outputs gathered experimentally. The methods are sometimes called "nonparametric" because they do not assume a detailed model structure such as that the inputs and outputs are related through a set of differential equations, where only certain parameter values need to be ascertained. Indeed, virtually no a priori knowledge of the system's structure is assumed, rather the system is regarded as an impenetrable blackbox. We consider only such methods because we will view the interpretation of gene expression profiles as essentially a case where the expression levels give rise to input signals, while the classes of importance, such as *metastatic* and *nonmetastatic* or *failed outcome* and *survivor*, create the desired output signals. The desired class predictor results from identifying a nonlinear system that is defined only by these input and output signals.

Throughout this chapter, it will be assumed that the given nonlinear system is time invariant, namely that a translation of the system inputs in time results in a corresponding translation of the system outputs. Such an assumption causes no difficulty in applying the approach to class prediction. One celebrated blackbox approach assumes that the input–output relation can be well approximated by a functional expansion such as the Volterra [1, 2] or the Wiener series [3]. For example, for the case of a single input $x(t)$ and single output $y(t)$, the approximation has form

$$y(t) = z(t) + e(t) \tag{2.1}$$

where

$$z(t) = \sum_{j=0}^{J} \int_0^T \cdots \int_0^T h_j(\tau_1, \ldots, \tau_j) x(t - \tau_1) \cdots x(t - \tau_j)\, d\tau_1 \cdots d\tau_j \tag{2.2}$$

and $e(t)$ is the model error. The right side of Eq. (2.2) is a Jth-order Volterra series with memory length T and the weighting function h_j is called the jth-order Volterra kernel. The zero-order kernel h_0 is a constant. System identification here reduces to estimation of all of the significant Volterra kernels, which involves solution of a set of simultaneous integral equations and is usually a nontrivial task. A fairly narrow class of systems, known as analytic [1, 2], can be exactly represented by the Volterra series of Eq. (2.2), where both J and T could be infinite. However, a much wider class can be uniformly approximated by such a series, with both J and T finite, according to Fréchet's theorem [4], which has been extended by Boyd and Chua [5]. The essential requirements are that the nonlinear system must have either finite [4] or fading [5] memory, and its output must be a continuous mapping of its input, in that "small" changes in the system input result in small changes in the system output. Then, over a uniformly bounded, equicontinuous set of input signals, the system can be uniformly approximated, to any specified degree of accuracy, by a Volterra series of sufficient but finite order J.

Wiener [3] essentially used the Gram–Schmidt process to rearrange the Volterra series into a sum of terms that were mutually orthogonal for a white Gaussian input x

with a specified power density level. The mutually orthogonal terms in Wiener's functional expansion were called G-functionals (G for Gaussian). The advantage of creating orthogonal functionals was to simplify kernel estimation and remove the requirement of solving simultaneous integral equations. Indeed, Wiener kernels in this expansion were determinable using the cross-correlation formula of Lee and Schetzen [6]. The Wiener kernels are not, in general, the same as the Volterra kernels, but when the complete set of either is known, the other set can be readily computed. If a system can be represented exactly by a second-order Volterra series [i.e., $e(t) \equiv 0$ in Eq. (2.1) and $J = 2$ in Eq. (2.2)], then the first- and second-order Volterra kernels equal the Wiener kernels of corresponding order. Once the kernels have been estimated, Eq. (2.2) can be used to calculate the Volterra series output for an arbitrary input x and thus "predict" the actual output y of the given nonlinear system.

In discrete time, for single input $x(i)$ and single output $y(i)$, the approximation has form

$$y(i) = z(i) + e(i) \tag{2.3}$$

where

$$z(i) = \sum_{d=0}^{D} \sum_{j_1=0}^{R} \cdots \sum_{j_d=0}^{R} h_d(j_1, \ldots, j_d) x(i - j_1) \cdots x(i - j_d) \tag{2.4}$$

and $e(i)$ is the model error. The Dth-order Volterra series on the right side of Eq. (2.4) has memory length $R + 1$ because $z(i)$ depends not only on $x(i)$ but also on earlier values $x(i), \ldots, x(i - R)$, that is, at input lags $0, \ldots, R$.

Indeed, this discrete-time Volterra series is simply a Dth-degree multidimensional polynomial in $x(i), \ldots, x(i - R)$, and the kernels $h_d(j_1, \ldots, j_d)$, $d = 0, \ldots, D$, are directly related to the coefficients of this polynomial. The zero-order kernel h_0 is the constant term of the polynomial. Any discrete-time system of finite [7] or fading memory whose output is a continuous mapping of its input (in the sense described above) can be uniformly approximated, over a uniformly bounded set of input signals, by the Volterra series on the right side of Eq. (2.4). Of course, D and R must be sufficiently large (but finite) to achieve the desired degree of accuracy. Applying the Gram–Schmidt process to the terms on the right side of Eq. (2.4) for a white Gaussian input x of specified variance can create a discrete form of the Wiener series. The kernels in this Wiener series are directly related to the Volterra kernels, and once the complete set of either is known, the other set can be readily calculated.

Several methods are available for estimating the Wiener or the Volterra kernels. If the input x is white Gaussian, then the Wiener kernels can be estimated using cross correlation either directly in the time domain by the Lee–Schetzen method [6] or efficiently in the frequency domain via the method of French and Butz [8]. The Lee–Schetzen approach was actually presented in continuous time [6] but is now most commonly applied in discrete time. Amorocho and Brandstetter [9], and later

Watanabe and Stark [10] and Marmarelis [11], expanded the kernels using Laguerre functions, then least-squares estimated the coefficients in the resulting expansion, and finally reconstructed the kernels using the estimated coefficients. Ogura [12] noted that use of Laguerre functions to approximate kernels having initial delay was less accurate than employing "associated" Laguerre functions and developed a fast algorithm for calculating the outputs of biorthogonal Laguerre filters.

Alternatively, the fast orthogonal algorithm (FOA) [13, 14] can be employed to estimate either the Wiener or the Volterra kernels, as can parallel cascade identification [14, 15]. Very widespread use has been made of the Lee–Schetzen [6] technique, with Sandberg and Stark [16] and Stark [17] being some of the first to exploit its power in modeling the pupillary control system. Marmarelis and Naka [18] and Sakai and Naka [19, 20] made imaginative applications of the Lee–Schetzen [6] technique to study information processing in the vertebrate retina. Barahona and Poon [21] used the FOA to detect deterministic nonlinear dynamics in short experimental time series. Orcioni et al. [22] studied the Lee–Schetzen [6] and fast orthogonal [13, 14] algorithms and gave practical suggestions concerning optimal use of these methods for estimating kernels up to third order. Zhang et al. [23] proposed a method of combining Akaike's final prediction error criterion [24, 25] with the FOA [13, 14] to determine the memory length simultaneously with the kernels. Westwick et al. [26] developed bounds for the variance of kernel estimates, computable from single data records, for the FOA [13, 14] and for kernel estimation via use of Laguerre functions [9–12].

When different kernel estimation procedures are compared, an issue that is sometimes overlooked is whether the test system's kernels are smooth or, instead, jagged and irregular. Smooth kernels can usually be well approximated using a small number of suitably chosen basis functions, such as the Laguerre set; jagged kernels typically cannot. If the simulated test system has smooth kernels, this favors basis expansion methods for estimating kernels, because they will require estimation of far fewer parameters (the coefficients in a brief basis function expansion) than the set of all distinct kernel values estimated by the FOA. In those circumstances, basis expansion methods will be shown in their best light, but a balanced presentation should point out that the situation is quite different when the test kernels have jagged or irregular shapes. Indeed, one may overlook valuable information inherent in the shape of a system's kernels by assuming a priori that the kernels are smooth. Moreover, in some applications, for example, Barahona and Poon's [21] use of functional expansions to detect deterministic dynamics in short time series, restrictive assumptions about the kernels' shapes must be avoided. If it cannot be assumed that the kernels are smooth, then the basis function approach will generally require an elaborate expansion with many coefficients of basis functions to be estimated. Hence the FOA is advantageous because it exploits the lagged structure of the input products on the right side of Eq. (2.4) to dramatically reduce computation and memory storage requirements compared with a straightforward implementation of basis expansion techniques.

The FOA uses the observation that estimating least-squares kernels up to Dth order in Eq. (2.4) requires calculating the input autocorrelation only up to order

$2D-1$. For example, suppose that the system input $x(i)$ and output $y(i)$ are available for $i=0,\ldots,I$ and that we seek the best approximation of the output, in the least-squares sense, by a second-order Volterra series [$D=2$ in Eq. (2.4)]. Then this requires calculating the input mean and autocorrelations up to third order, namely

$$\phi_{xxxx}(i_1,i_2,i_3)=\frac{1}{I-R+1}\sum_{i=R}^{I}x(i)x(i-i_1)x(i-i_2)x(i-i_3)$$

for

$$i_1=0,\ldots,R \qquad i_2=0,\ldots,i_1 \qquad i_3=0,\ldots,i_2$$

The lower order autocorrelations are defined analogously. For $I>R^3$, the most time-consuming part of the FOA is the calculation of the third-order autocorrelation. The computational requirement to do this has been overestimated in various publications (e.g. [27]), sometimes three times too large, so it is worthwhile to consider an efficient scheme. For example, the input mean and third- and lower order autocorrelations can be obtained as follows:

```
For i=R to I
Q=x(i)
Avg=Avg+Q
  For i1=0 to R
  QQ=Q·x(i-i1)
  φxx(i1)=φxx(i1)+QQ
     For i2=0 to i1
     QQQ=QQ·x(i-i2)
     φxxx(i1,i2)=φxxx(i1,i2)+QQQ
          For i3=0 to i2
          QQQQ=QQQ·x(i-i3)
          φxxxx(i1,i2,i3)=φxxxx(i1,i2,i3)+QQQQ
          Next i3
     Next i2
  Next i1
Next i
```

After this, each of Avg, ϕ_{xx}, ϕ_{xxx}, ϕ_{xxxx} is divided by $I-R+1$. The above pseudocode requires about $IR^3/3!$ multiplications to compute the mean and all autocorrelations up to third order [14], and for $I>R^3$, that is the majority of multiplications the FOA requires to calculate kernels up to second order.

For the most part, Volterra [1, 2] and Wiener [3] series are of practical use for approximating "weakly nonlinear" systems, with an order of nonlinearity less than or equal to, say, 3. In instances where it is possible to precisely apply special inputs, for example complete binary *m*-sequences, then high-order kernels can be rapidly and accurately estimated by Sutter's innovative approach [28, 29]. Alternatively, some nonlinear systems can be well approximated by a cascade of a dynamic

linear, a static nonlinear, and a dynamic linear element [30–33], sometimes referred to as an *LNL* cascade. A method that leads to an identification of each of these elements, within arbitrary scaling constants and a phase shift, from a single application of a white Gaussian noise input was presented in 1973 [32, 33] and has had several applications in neurophysiology [34–36]. One interesting property of the *LNL* cascade is that its Wiener kernels are proportional to its Volterra kernels of corresponding order [32, 33]. Moreover, for a white Gaussian input, the shape of the first linear element's impulse response is given by the first nonnegligible slice of a second-order cross correlation parallel to one axis [15, 37]. Thus the first nonnegligible term in the sequence $\phi_{xxy}(j, 0)$, $\phi_{xxy}(j, 1), \ldots$ (here ϕ_{xxy} is the second-order cross correlation between a white Gaussian input $x(i)$ and the cascade output $y(i)$ after removing its mean) reveals the first dynamic linear element up to a scaling constant. Subsequently, methods related to the approach of [32, 33] have been published by a number of authors [15, 37–39]. However, the use of such inputs does not apply to the case of interpreting gene expression profiles. For the latter application, it is useful to resort to parallel cascade identification [14, 15], which is effective in approximating systems with high-order nonlinearities and does not require special properties of the input.

2.3. PARALLEL CASCADE IDENTIFICATION

Parallel cascade identification (PCI) seeks to approximate a given discrete-time dynamic nonlinear system by building a parallel array of alternating dynamic linear (L) and static nonlinear (N) elements using only the system's input–output data gathered experimentally [14, 15]. By "dynamic" is meant that the element has memory of length $R + 1$, as explained above, where $R > 0$. An example of a parallel *LN* cascade model is shown in Figure 2.1 and will be used below for class prediction. This parallel *LN* model is related to a parallel *LNL* model introduced by Palm [7] to approximate a discrete-time nonlinear system, of finite memory and anticipation, whose output was a continuous mapping of its input, in the sense explained above. While Palm allowed his linear elements to have anticipation as well as memory [7], only nonanticipatory elements will be discussed here. In certain applications (e.g., to locate the boundaries of coding regions of DNA), anticipation will be beneficial. In Palm's model, the static nonlinear elements were logarithmic and exponential functions rather than the polynomials used here. Palm did not suggest any method for identifying his parallel *LNL* model, but his article motivated much additional research in this area. When each N in Figure 2.1 is a polynomial, the model has also been called a polynomial artificial neural network [27], but we will continue the tradition of referring to it as a parallel cascade model.

Subsequent to Palm's [7] work, a method was proposed for approximating, to an arbitrary degree of accuracy, any discrete-time dynamic nonlinear system having a Wiener series representation by building a parallel cascade model (Fig. 2.1) given only the system input and output [14, 15]. The method begins by approximating the nonlinear system by first a cascade of a dynamic linear element followed by a

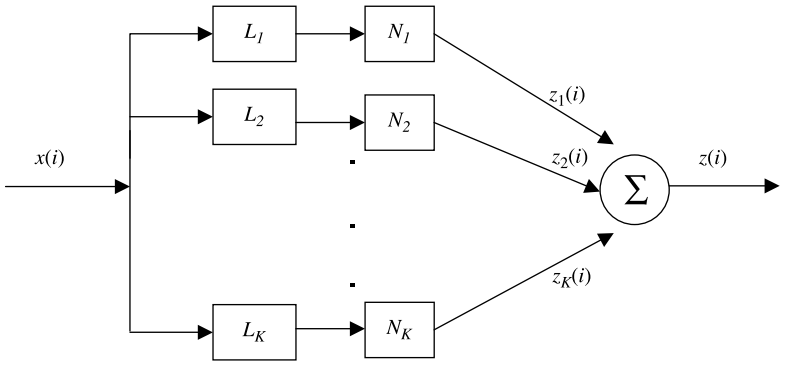


FIGURE 2.1. Parallel cascade model used for class prediction. In each path, L is a dynamic linear element and N is a polynomial static nonlinearity. (From [63].)

polynomial static nonlinearity. The residual, namely the difference between system and cascade outputs, is treated as the output of a new nonlinear system driven by the same input, and a second cascade is found to approximate the new system. The new residual is computed, a third cascade path can be found to improve the approximation, and so on. Each time a cascade is added, the polynomial static nonlinearity can be least-squares fit to the current residual. Under broad conditions, the given nonlinear system can be approximated arbitrarily accurately, in the mean-square sense, by a sum of a sufficient number of the cascades. However, each of the cascade paths may be found individually, which keeps the computational requirement low and the algorithm fast.

We will describe in detail the identification of a single-input, single-output nonlinear system, although a multivariate form of PCI is also available [14]. Assume that the nonlinear system output $y(i)$ depends on input values $x(i), \ldots, x(i-R)$, that is, has memory length $R+1$, and that its maximum degree of nonlinearity is D. Moreover, this input and output are only available over a finite record: $x(i)$, $y(i)$, $i = 0, \ldots, I$. Suppose that $z_k(i)$ is the output of the kth cascade and $y_k(i)$ is the residual left after k cascades have been added to the model. Then $y_0(i) = y(i)$, and more generally, for $k \geq 1$,

$$y_k(i) = y_{k-1}(i) - z_k(i) \tag{2.5}$$

Consider finding the kth cascade, which will begin with a dynamic linear element that can be specified by its impulse response $h_k(j)$, and there are many ways that this can be chosen. One alternative is to set it equal to the first-order cross correlation of the input with the latest residual, $y_{k-1}(i)$, or to a slice of a higher order cross correlation with impulses added at diagonal values. Thus, for $j = 0, \ldots, R$, set

$$h_k(j) = \phi_{xy_{k-1}}(j) \tag{2.6}$$

if the first-order cross correlation $\phi_{xy_{k-1}}$ is used, or

$$h_k(j) = \phi_{xy_{k-1}}(j, A) \pm C\delta(j - A) \tag{2.7}$$

if the second-order cross correlation $\phi_{xxy_{k-1}}$ is instead chosen, or

$$h_k(j) = \phi_{xxxy_{k-1}}(j, A_1, A_2) \pm C_1\delta(j - A_1) \pm C_2\delta(j - A_2) \tag{2.8}$$

if the third-order cross correlation $\phi_{xxxy_{k-1}}$ is employed [14, 15]. Analogous choices involving slices of higher order cross correlations can be made (up to the assumed order of nonlinearity D). Which alternative is selected to define $h_k(j)$ can be decided at random, provided that there is a nonzero probability that each may be chosen. If Eq. (2.7) is used to define $h_k(j)$, then the value of A (determining the slice) can be chosen at random from $0, \ldots, R$ and the sign of the δ-term can also be chosen randomly. The coefficient C in Eq. (2.7) is chosen to tend to zero as the mean square of $y_{k-1}(i)$ tends to zero. When PCI is used for kernel estimation, it is useful to further constrain the magnitude of C to not exceed the maximum absolute value of the slice $\phi_{xxy_{k-1}}(j, A), j = 0, \ldots, R$. Analogous comments apply when Eq. (2.8) is used to define $h_k(j)$. Instead of randomly choosing $h_k(j)$ in the manner just set out, a deterministic progression through each of the various alternatives can be employed. Alternatively, the same "random" sequence can be used every time the algorithm is run. Many other strategies [14] can be used to define $h_k(j)$, and the method is not limited to use of slices of cross-correlation functions.

Once the dynamic linear element beginning the k th cascade has been determined, calculate its output,

$$u_k(i) = \sum_{j=0}^{R} h_k(j)x(i - j) \tag{2.9}$$

which forms the input to the polynomial static nonlinearity. The latter's coefficients can be found by least-squares fitting its output,

$$z_k(i) = \sum_{d=0}^{D} a_{kd}u_k^d(i) \tag{2.10}$$

to the latest residual $y_{k-1}(i)$. To increase the accuracy of estimating the coefficients, the impulse response function $h_k(j)$ can first be scaled so that the linear element's output $u_k(i)$ has unity mean square [40]. The new residual is then calculated from Eq. (2.5), and the process of adding cascades can continue analogously. Since the coefficients a_{kd} are least-squares estimated, it follows that

$$\overline{y_k^2(i)} = \overline{y_{k-1}^2(i)} - \overline{z_k^2(i)} \tag{2.11}$$

where the overbar denotes the average over $i = R, \ldots, I$. Thus, by Eq. (2.11), adding the kth cascade reduces the mean square of the residual by an amount equal to the

mean square of that cascade's output. This alone does not imply that the mean square of the residual can be driven to zero by adding further cascades. However, due to the way the impulse responses for the dynamic linear elements are defined as cascades are added, the parallel cascade output does converge to the nonlinear system output in the mean-square sense [14]. Moreover, as noted above, other effective methods of finding the impulse responses exist. If there are K cascades in total, then the PCI model output is

$$z(i) = \sum_{k=1}^{K} z_k(i) \tag{2.12}$$

To reduce the possibility of adding ineffectual cascades that are merely fitting noise, before accepting a candidate for the k th path, one may require [14] that

$$\overline{z_k^2(i)} > T\overline{\frac{y_{k-1}^2(i)}{I_1}} \tag{2.13}$$

where I_1 is the number of output points used in the identification and T is a threshold. Here, the output $y(i)$ was used over the interval $i = R, \ldots, I$, so $I_1 = I - R + 1$. In the applications below to class prediction, I_1 has a slightly different value to accommodate transition regions in the training input. If the residual $y_{k-1}(i)$ were independent zero-mean Gaussian noise, then, when $T = 4$ and I_1 is sufficiently large, the inequality (2.13) would *not* be satisfied with probability of about 0.95. Clearly, increasing T in the above inequality increases the reduction in mean-square error (MSE) required of a candidate cascade for admission into the model. If the candidate fails to satisfy this inequality, then a new candidate cascade can be constructed and tested for inclusion as the k th path. This involves making a new choice for $h_k(j)$ using the strategy described above and then best fitting the polynomial static nonlinearity that follows. The process of adding cascades may be stopped when a specified number have been added or tested, or when the MSE has been made sufficiently small, or when no remaining candidate can cause a significant reduction in MSE [14].

While the above background material has focused on nonparametric identification methods, there exist general-purpose search techniques, such as FOS [13, 41, 42], for building difference equation or other models of dynamic nonlinear systems with virtually no a priori knowledge of system structure. Fast orthogonal search is related to an approach by Desrochers [43] for obtaining nonlinear models of static systems. However, the latter method has computational complexity and storage requirement dependent upon the square of the number of candidate terms that are searched, while in FOS the dependence is reduced to a linear relationship. In addition FOS and/or iterative forms [44–47] of FOS have been used for high-resolution spectral analysis [42, 45, 47, 48], direction finding [44, 45], constructing generalized single-layer networks [46], and design of two-dimensional filters [49], among many applications. Wu et al. [50] have compared FOS with canonical variate analysis for biological applications.

2.4. CONSTRUCTING CLASS PREDICTORS

The blackbox identification considered above seeks to build a model that can approximate the behavior of a given dynamic nonlinear system from knowledge only of the system input and output. Such a capability lends itself well to constructing effective class predictors in many practical cases. For example, consider the problem of predicting the structure/function family of a novel protein given only its primary amino acid sequence. The amino acid sequences can be regarded as the inputs and their corresponding families as the outputs [40, 51]. First, some means is used to map the amino acid sequence into a corresponding numerical sequence and similarly to numerically designate the families to be distinguished.

For example, the Rose [52] scale assigns each amino acid a hydrophobicity value, converting each protein sequence into a hydrophobicity profile. To distinguish between, say, the globin and calcium-binding families, the profiles of one or more exemplars from each family were spliced together to form a training input [40]. The corresponding training output was defined to have ideal value -1 over globin segments of the training input and 1 over calcium-binding segments. A parallel cascade model was found to approximate this input–output relation. Suppose that $R+1$ is the memory length of the model, so that its output depends on input delays $0, \ldots, R$. Then, to allow the model to "settle," those values of the training output corresponding to the first R points of each segment joined to produce the training input were excluded from the identification. A novel sequence was classified by feeding its corresponding profile through the identified model and then computing the average of the resulting output starting with the $(R+1)$th point. If the average was less than zero, the sequence was classified as globin and otherwise as calcium binding [40]. While effective classifiers were built encoding amino acids by the Rose scale, the resulting training inputs had some drawbacks for nonlinear system identification. First, the Rose scale is not one to one, since some amino acids, such as leucine, methionine, and tryptophan, are assigned the same hydrophobicity value, so there is a loss of information in going from the amino acid sequence to the resulting hydrophobicity profile. Second, the assigned values cover a narrow range while weighting some amino acids more heavily than others.

Indeed, as reported subsequently [51], use of certain "simultaneously axially and radially aligned hydrophobicities (SARAH) scales" to uniquely encode the amino acids via 5-tuples increased PCI classification accuracy. In the SARAH1 scale, the code for each amino acid has three entries that are 0 and two that are both 1 or both -1. In the SARAH2 scale, each code has two 0 entries and three that are all 1 or all -1. For both scales, the amino acids were ranked according to the Rose scale (breaking ties), and then the codes were assigned in descending order of their binary number values. Either SARAH scale leads to either a numerical sequence five times longer than the amino acid sequence or to a set of five signals each of the same length as the amino acid sequence [51] for use with a five-input parallel cascade classifier [14]. To classify a novel protein that was encoded by a SARAH scale, an MSE ratio test [51] yielded higher accuracy than using the sign of the mean output as discussed above. In the ratio test, the MSE of the output signal

from the ideal value for a class is normalized by the MSE of the training exemplars for that class, and the sequence is assigned to the class with the smallest of the ratios. When tested on over 16,000 protein sequences in two-way classifications, parallel cascade classifiers using SARAH1-encoded sequences outperformed state-of-the-art hidden Markov models [53] trained with the same three protein exemplars (one sequence each from globin, calcium-binding, and kinase classes) [51].

David et al. [54] reviewed advances in threading approaches to protein fold recognition, including unconventional threaders such as proximity correlation matrices [55] and PCI. In addition to the work described above, similar applications of PCI have been made to distinguish between coding (exon) and noncoding (intron) human DNA sequences [56] and to recognize sites on proteins that bind to adenosine triphosphate (ATP) and guanosine triphosphate (GTP) [57].

In the next section, the use of PCI to interpret gene expression profiles is considered in detail.

2.5. PREDICTION BASED ON GENE EXPRESSION PROFILING

The parallel cascade model can be regarded as a special kind of artificial neural network (ANN) where the interaction between pathways is reduced to summing of their outputs and where the pathways contain nonlinearities in the form of polynomial activation functions. The ANNs have been successfully used to classify cancers based on gene expression, with Khan et al. [58] demonstrating flawless discrimination among four categories of small, round blue-cell tumors. The present section describes how PCI predictors of treatment response, clinical outcome, and metastatic status have been built based on gene expression profiles.

A gene expression profile $\boldsymbol{p}_j$ can be thought of as a column vector containing the expression levels $e_{i,j}$, $i = 1, \ldots, I$, of I genes. We suppose that we have J of these profiles for training, so that $j = 1, \ldots, J$. Each of the profiles $\boldsymbol{p}_j$ was created from a sample (e.g., from a tumor) belonging to some class. The samples may be taken from patients diagnosed with various classes of leukemia, for example, acute lymphoblastic leukemia (ALL) or acute myeloid leukemia (AML), as in a classic paper by Golub et al. [59]. Given a training set of profiles belonging to known classes, (e.g., ALL and AML), the problem is to create a predictor that will assign a new profile to its correct class. Brazma and Vilo [60] refer to the table of I gene rows and J sample columns as a gene expression matrix.

This section focuses on three classification problems based on gene expression profiling, predicting:

1. Treatment response of a group of AML patients using data from Golub et al. [59]
2. Clinical outcome of a group of medulloblastoma patients using data from Pomeroy et al. [61]
3. Metastatic status of primary medulloblastoma tumors using data from MacDonald et al. [62]

One difference from protein family prediction, or recognition of coding regions, is that, unlike sequences of amino acids or DNA sequences, the gene expression values in a profile are not ordered sequentially. However, it is still possible to build effective gene-expression-based predictors via nonlinear system identification, and indeed there are even some advantages, such as the freedom to vary the ordering of the gene expression levels in constructing the training input [63]. Once the ordering scheme is chosen, it is maintained both to construct each segment of the training input and to prepare the individual input signals corresponding to test profiles. Thus, while the expression values in the original profiles are not ordered sequentially, the training input segments and the test input signals do have an imposed order. Parallel cascade identification simply looks for a pattern in the data. The approach depends on training exemplars from different classes producing different patterns and the PCI model having appropriate memory length to capture the pattern for each class [63].

2.5.1. Predicting AML Treatment Response

For 15 adult AML patients treated with anthracycline-cytarabine, there were eight failed treatments, where complete remission had not resulted, and seven successful treatments, where complete remission of at least 36 months had been achieved. At the time of leukemia diagnosis, samples were obtained that were used for gene expression profiling via an Affymetrix oligonucleotide microarray. Each profile contained expression levels of 6817 human genes, but because of duplicates and additional probes in the microarray, a total of 7129 expression levels were present in the profile. Golub et al. [59] were able to construct predictors that could distinguish very accurately between various acute leukemia classes. However, they found no strong gene expression signature correlated with clinical outcome and stated that their outcome predictors were "not highly accurate in cross-validation" [59]. Similarly, for the same data, Schuster et al. [64] found that none of five different clustering methods (Kohonen clustering, fuzzy Kohonen network, growing cell structures, *K*-means clustering, and fuzzy *K*-means clustering) clustered patients having similar treatment response.

Recently, PCI was used successfully to predict treatment response [63] from the same microarray data as above. See also the review by Kirkpatrick [65]. To build an appropriate parallel cascade model, the first step was to create a training input. A set of genes were selected using the same number and selection method found to be effective in building PCI models to distinguish ALL from AML profiles. Therefore, the first profile corresponding to a failed (F) treatment and the first profile corresponding to a successful (S) treatment were compared and the 200 "most important" genes were located. For each of these genes, the absolute value of the difference between the corresponding raw scores on the first F and S profiles ranked in the top 200 of the 7129 genes. The raw expression values from this F profile for these 200 genes were juxtaposed to form the F segment to be used for training, and the S segment was similarly prepared from the first S profile. The two information-rich segments were then spliced together to form a 400-point training input $x(i)$ (Fig. 2.2*a*). The 200 expression values for each segment were appended

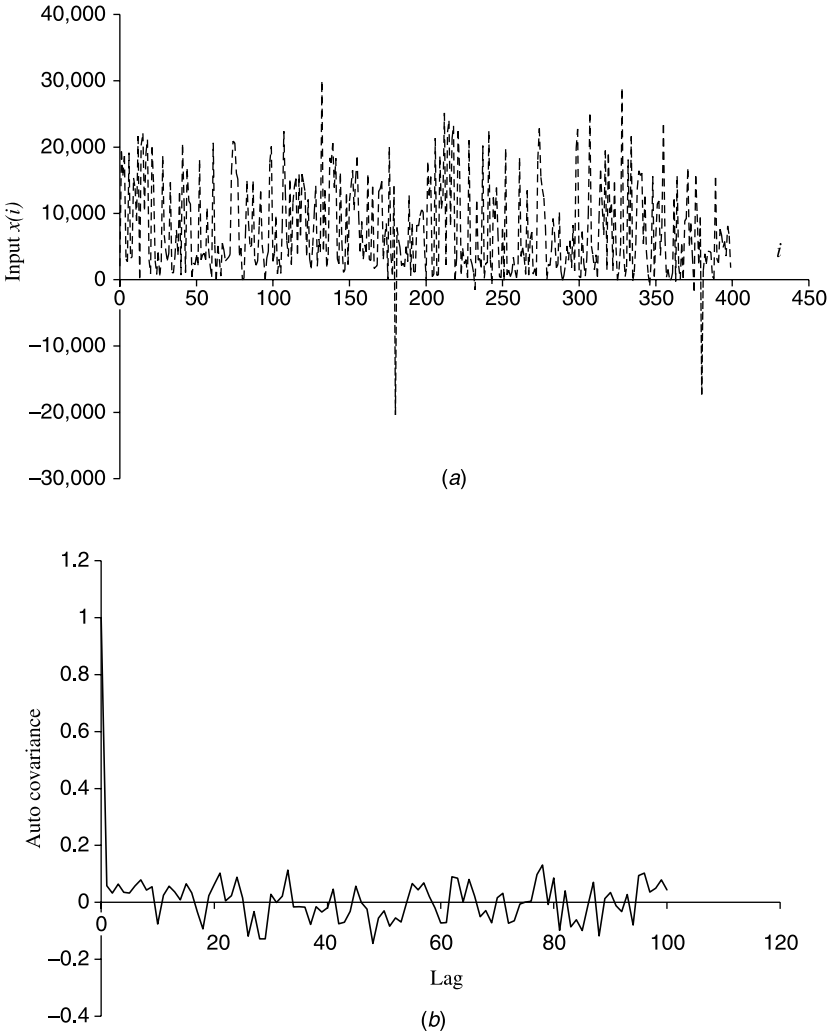
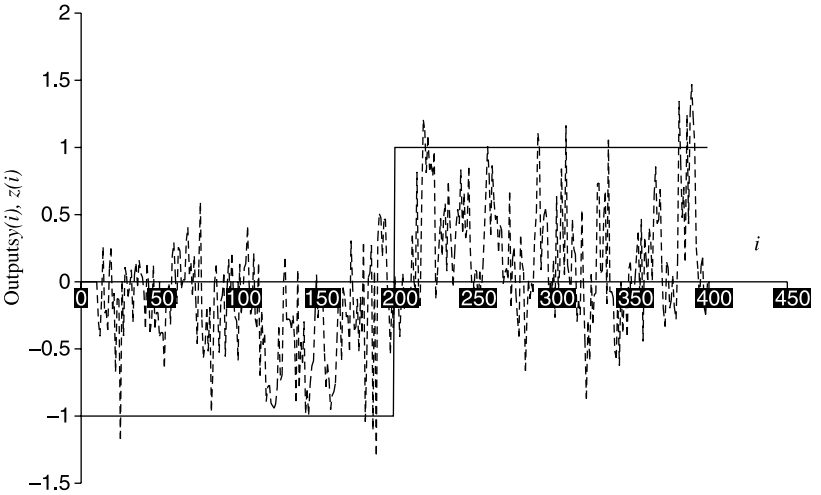


FIGURE 2.2. (*a*) Training input $x(i)$ formed by splicing together raw expression levels of genes from first "failed treatment" profile 28 and first "successful treatment" profile 34. The genes used were the 200 having greatest difference in expression levels between the two profiles. (*b*) The order used to append the expression levels of the 200 genes caused the autocovariance of the training input to be nearly a delta function, indicating that the training input was approximately white.

in the same relative order that they had in the original profile, and this is true for all the examples described here. However, it has been found that other ordering schemes may be beneficial, for example those that cause the autocovariance of the training input to be almost a delta (i.e., discrete impulse) function [63].



(*c*)

FIGURE 2.2. (*c*) Training output $y(i)$ (solid line) defined as -1 over failed treatment portion of training input and 1 over successful treatment portion. The training input and output were used to identify a parallel cascade model of the form in Figure 2.1. The dashed line represents calculated output $z(i)$ when the identified model is stimulated by training input $x(i)$. Note that $z(i)$ is predominately negative over the failed treatment portion and positive over the successful treatment portion of the training input. (From [63].)

Indeed, as shown in Figure 2.2*b*, the order used resulted in an almost white input, which is typically advantageous for nonlinear system identification techniques, including PCI. The corresponding training output $y(i)$ (Fig. 2.2*c*, solid line) was defined to be -1 over the F segment and 1 over the S segment.

For this training input and output, a PCI model was identified using the method described above [14, 15]. This model could certainly distinguish between F and S profiles, at least for the training exemplars. Indeed, as shown in Figure 2.2*c*, when the training input $x(i)$ was applied to the identified model, the resulting output $z(i)$ (dashed line) was predominately negative over the F segment and positive over the S segment of the input.

To identify this model, certain parameters chiefly related to architecture had to be determined. As noted earlier, these are (1) the memory length $(R+1)$ of the dynamic linear element, (2) the degree D of the polynomial static nonlinearity, (3) the maximum number of cascades allowed into the model, and (4) the threshold T concerning required reduction in MSE for accepting a candidate cascade. How these parameters were selected is explained next.

The first F and S profiles used to construct the training input were reserved for this purpose, which left 13 profiles for testing. Each time, 1 of the 13 profiles was held out for testing, while the other 12 profiles were employed to determine the above parameter values to be used in that test. This was done by testing the accuracy

over the 12 profiles of various PCI models identified from the same training input and output with different values for the above four parameters. The most accurate model was chosen to classify the test profile. Thus the test profile was not used to determine the parameter values for the model that would classify it. The procedure was repeated until all 13 profiles had been classified. It emerged that the same parameter settings (memory length 12, polynomial degree 7, seven cascades in total, threshold $T = 11$) were chosen each time, so that the same PCI model was in fact selected for each classification of the held-out profile [63].

To classify a test profile, a corresponding input signal was prepared by concatenating the raw expression values of the 200 selected genes in the same order used above. This input was fed to the identified PCI model to obtain a corresponding output signal. The sign of the mean output was used as the decision criterion for classification, as explained above [63]. The PCI model correctly classified five of the seven test F profiles and five of the six test S profiles. Moreover, the model's *individual* output values for the test profiles clearly correlated with the F-versus-S class distinction ($P < 0.0155$, one tail). Finally, the way that the model's *mean* output values ordered the test profiles also showed that it distinguished between F and S profiles. Indeed, the ranking of the test profiles by their corresponding mean outputs in Table 2.1 demonstrates that F profiles tend to precede S profiles, and this difference is significant on the Mann-Whitney test ($P < 0.0367$, one tail). One-tailed tests were used because, due to the way the training output had been defined, output values corresponding to F profiles would be expected to be smaller than those corresponding to S profiles.

Why does the nonlinear system identification approach work with so few training data? It is because the system output value depends only upon the present and a finite

TABLE 2.1 Parallel Cascade Ranking of Test Expression Profiles

Rank	Mean Output	Actual Outcome	Profile No.
1	−1.17	F	31
2	−0.863	F	32
3	−0.757	F	33
4	−0.408	S	37
5	−0.298	F	50
6	−0.0046	F	30
7	0.0273	S	53
8	0.078	S	38
9	0.110	F	51
10	0.148	F	29
11	0.194	S	52
12	0.267	S	36
13	16.82	S	35

Source: Ref. [63].

Note: F = failed treatment, S = successful treatment. The complete set of profiles is found in [59], and "Profile No." follows the same numbering scheme.

number of delayed input (and possibly output) values, covering a shorter length than the length of the individual segments joined to form the training input. This requirement is always met by a model having finite memory less than the segment lengths but applies more generally to finite-dimensional systems. These systems include difference equation models, which have fading rather than finite memory. However, the output at a particular "instant" depends only upon delayed values of the output and present and delayed values of the input covering a finite interval. For example, the difference equation might have the form

$$y(i) = F\,[\,y(i-1), \ldots, y(i-I_A), x(i), \ldots, x(i-I_B)]$$

So long as the maximum of the output delay I_A and the input delay I_B is considerably less than the number of points in each input segment, we derive numerous training examples from each segment joined to form the input.

To illustrate, the parallel cascade model was assumed above to have a memory length of 12 points, whereas the F and S segments of the training input each comprised 200 points [63]. Having a memory length of 12 means that we assume it is possible for the parallel cascade model to decide whether a segment portion is F or S based on the expression values of 12 genes. Thus the first F training example for the parallel cascade model is provided by the first 12 points of the F segment, the second F training example is formed by the 2nd to the 13th points, and so on. Hence each 200-point segment actually provides 189 training examples, so that a total of 378 training examples, and not just 2, are provided by the single F and S input segments.

2.5.2. Predicting Medulloblastoma Clinical Outcome

Predicting clinical outcome from gene expression profiles obtained at diagnosis could dramatically alter and individualize cancer therapy. Bredel et al. [66] have comprehensively reviewed the use of gene expression profiling in human brain tumors and pointed out that drug resistance here is likely to involve a complex network of regulatory dynamics. Recently, Pomeroy et al. [61] showed that a variety of classification techniques, including weighted voting (WV), *k*-nearest neighbors (*k*-NN), support vector machines (SVMs), and IBM SPLASH could be used to predict clinical outcome of a group of 60 medulloblastoma patients from their gene expression profiles. While these methods made relatively few errors in leave-one-out testing, they were biased in favor of recognizing survivors compared to those with failed outcomes. For example *k*-NN made the fewest total errors (13) but correctly identified only 10 of 21 (47.6%) with failed outcome, in contrast to 37 of 39 (94.9%) survivors, averaging 71% if the two outcome subgroups are weighted equally. A single-gene TRKC (neurotrophin-3 receptor) predictor showed reverse bias, recognizing 81% with failed outcomes and 59% of survivors. By combining predictors via majority voting, Pomeroy et al. [61] reduced total errors to 12, but the resulting accuracy still favored the survivor subgroup (89.7%) over the failed subgroup (61.9%). Such inaccuracy in recognizing one of the outcomes poses a problem for clinical use.

In [67], PCI was applied to this data set using the raw values given after rescaling by Pomeroy et al. [61], where all expression profiles were of tumor samples obtained at diagnosis. First, the same method of selecting genes, the same number of genes, and the same architectural parameter values employed in the AML study [63] were used to identify a PCI model. Thus, the first profile for a failed (F) outcome and the first for a survivor (S) outcome were compared, and the 200 top-ranked genes, that is, with greatest difference in raw expression levels between the two profiles, were selected. The selected genes' raw expression levels from the first F profile were appended in the relative order they had in the profile to form an F segment, and an S segment was similarly prepared from the first S profile. The F and S segments were spliced together to form a 400-point training input, and the corresponding output was again defined as -1 over F and 1 over S segments of the training input. Then the identical parameter values (memory length 12, polynomial degree 7, seven cascades in total, threshold $T = 11$) were used as in the AML study [63] to identify a PCI model from the training input and output. Hence the remaining 58 profiles in the data set were not used to obtain the model but were instead reserved for testing it [67].

In particular, the PCI model was used as a filter that converted input signals corresponding to the test profiles into output signals that were much easier to classify than the original profiles [67]. Thus a 200-point input signal was prepared from each test profile by appending the raw expression values of the previously selected genes in the same order used above. Each input signal was fed through the PCI model to obtain an output signal corresponding to the test profile. Since memory length was 12, the first 11 points of each output signal were ignored to allow the model to settle, and only the last 189 points of the output were used to classify it.

The first issue to resolve was whether replacing the input signals by the model outputs benefited classification accuracy. Pomeroy et al. [61] had used a leave-one-out protocol to measure the accuracy of the methods tested, and the same procedure was adopted in [67]. Thus, each of the 58 test profiles was classified by calculating the Euclidean distance of its output signal from each of the other 57 output signals and choosing the class of the closest. Of the 58 profiles, 12 of 20 F (60%) and 31 of 38 S (81.6%) were correctly classified, a 71% average. However, using the same test procedure with the input signals, rather than their corresponding model outputs, resulted in only 7 of 20 F (35%) and 25 of 38 S (65.8%) correctly classified, averaging 50% and showing that the PCI model was essential.

At this point, the accuracy obtained classifying the PCI model output signals [67] appears similar to that obtained by Pomeroy et al. [61] using k-NN to classify the profiles, but there is a crucial difference in how the accuracy was measured. The k-NN accuracy reported was for leave-one-out creation and testing of 60 eight-gene predictive models with $k = 5$, which were found to be optimal after trying models with 1 to 200 genes and different values of k. Because of the leave-one-out creation of the models, no single set of genes was shown to form an effective predictor for all the profiles. However, the PCI model and the genes to use were found from only the first F and S profiles, which were excluded from the test set. The four parameter values needed (for memory length, polynomial degree, etc.) came from the AML study, as did the number of genes to use and the method of

selecting genes. Thus, unlike the earlier study [61], there was no searching for a better set of parameter values or number of genes to use. The same set of 200 genes and the same PCI model were employed to create the output signals for the remaining 58 test profiles, although a leave-one-out protocol was used in classifying these output signals.

For fairer comparison, various numbers of genes and different parameter values were tried in creating PCI models, still using only the first F and S profiles to select the top-ranked genes and construct the training input. Once the model output signals were produced for the remaining 58 profiles, a leave-one-out protocol was again used to classify these output signals. It was found that better accuracy was obtained by selecting 22 genes, a memory length of 4, polynomial degree of 5, a threshold T of 6, two cascades in the final model, and using the largest correlation coefficient, rather than the smallest Euclidean distance, as the decision criterion [67]. Then 14 of 20 (70%) F and 32 of 38 (84.2%) S test profiles were classified correctly, a 77% average. Using the input signals rather than the model outputs dropped accuracy to 50% on F and 76.3% on S profiles, averaging 63%, again showing the benefit of the PCI model.

An encouraging development was that PCI formed a strong component in combination predictors. Indeed, combining PCI with various predictors used by Pomeroy et al. could reduce total errors to 9 (70% on test F, 92.1% on test S). Another more symmetric predictor (80% on test F, 78.9% on test S) resulted from a PCI, metastatic staging, TRKC combination [67]. These results still require verification on independent sets since certain parameter values for the PCI model and number of genes to be used had not been prespecified, unlike the 200-gene medulloblastoma outcome prediction case discussed first.

Indeed, the importance of separating training and test sets is illustrated by a recent study by van't Veer et al. [68] concerned with using gene expression profiles to predict clinical outcome of breast cancer. The study appears to show a large advantage to a gene-expression-based predictor compared with a number of clinical predictors, such as based on tumor grade, estrogen receptor status, progesterone receptor status, tumor size, patient age, and angioinvasion. Moreover, there was some validation of the microarray predictor on an additional independent set, with only 2 incorrect out of 19 classifications. However, Tibshirani and Efron [69] stress that comparing predictors over the same data set used to derive the microarray predictor (called "reuse") strongly biases the results in favor of the latter predictor. They used a "prevalidation analysis" in which the predictor of the class of a particular profile was not derived using knowledge of that profile's class. Their resulting odds ratio (odds of developing distant metastases within 5 years with a "poor prognosis" signature to the odds of distant metastases without the signature) for the microarray predictor was much smaller than earlier computed by van't Veer et al. also using prevalidation. In addition, using cross validation to prevent reuse, Tibshirani and Efron [69] could not find that including the microarray predictor with six clinical predictors in a logistic regression model conferred any prediction advantage compared to a logistic model without the microarray predictor.

2.5.3. Predicting Medulloblastoma Metastasis

An intriguing study by MacDonald et al. [62] has helped to elucidate the genetic pathways underlying metastasis of medulloblastoma. Their work identified some genes critical to this process and also led them to suggest novel therapeutic interventions, including the use of specific inhibitors of platelet-derived growth factor receptor α as possible new treatments of medulloblastoma. In addition, they adapted the weighted voting scheme of Golub et al. [59] to obtain promising results predicting medulloblastoma metastatic status of primary tumors. Their data set comprised profiles for 14 nonmetastatic (M0) and 9 metastatic (M+) tumor samples.

The set was so small that completely separating training and test sets left very few exemplars to build a PCI predictor. The PCI training set consisted only of the first three each of the M+ and M0 profiles, while the remaining profiles were entirely reserved for testing [70]. The first M+ and M0 profiles were used both to select genes and to construct a training input, as described above for medulloblastoma outcome prediction [67]. With the same parameter values (memory length 4, polynomial degree 5, threshold 6, two cascades in model) and number of genes (22) from that study, a PCI model was identified [70]. Then the model was used to obtain reference output signals corresponding to the remaining two training profiles from each class. Using correlation with the reference model outputs to predict class [67] (see above) yielded these test results: 5 of the 6 novel M+ and 8 of the 11 novel M0 profiles were correctly classified (Matthews' [71] correlation coefficient $\phi = 0.54$, Fisher's exact test $P < 0.043$, one tail, $P < 0.05$ two tail) [70].

However, some luck is needed when there is such a paucity of training data: The few known profiles must be representative of their classes or nothing will work, and typically so few exemplars will not be enough to cover the variety of profile types in each class. Indeed, all the above PCI models were identified with a training input constructed from one exemplar expression profile from each class. These are extreme cases, and ideally more exemplars will be available both to train the PCI model and to construct reference output signals for classifying novel profiles. To illustrate this, the first four exemplars from each of the M0 and M+ classes were used to select the top-ranked 22 genes (Table 2.2). One of these genes, *SPARC/osteonectin*, is also in the set of 22 previously selected to predict clinical outcome of medulloblastoma [67].

The above four exemplars from each of the M0 and M+ classes were also employed to construct a training input (Fig. 2.3*a*). Hence these exemplars gave rise to eight 22-point segments, resulting in a 176-point training input. The corresponding training output (solid line, Fig. 2.3*b*) was defined as -1 over each M+ segment and as 1 over each M0 segment of the training input. The same parameter values as before (memory length 4, polynomial degree 5, threshold 6, two cascades in the model) were used for the PCI model. The dashed line in Figure 2.3*b* shows the output when the identified model is stimulated by the training input. Next, the remaining profiles were used to prepare reference output signals from the model. The resulting predictor could classify correctly all three metastatic cell lines and

TABLE 2.2 Twenty-Two Genes Used to Predict Medulloblastoma Metastasis

Position in Profile (1-2059)	Description
90	M33764cds Human ornithine decarboxylase gene, complete cds
115	M11717mRNA Human heat shock protein (hsp 70) gene, complete cds
219	D13748 HUM4AI Human mRNA for eukaryotic initiation factor 4AI
467	D78577expanded D78576S2 Human DNA for 14-3-3 protein eta chain; exon2 and complete cds
744	M55409 Human pancreatic tumor-related protein mRNA, 3′ end
763	D11139exons# 1-4 HUMTIMP Human gene for tissue inhibitor of metalloproteinases; partial sequence
1078	X58965 H.sapiens RNA for nm23-H2 gene
1083	X73066cds Homo sapiens NM23-H1 mRNA
1138	M55914 HUMCMYCQ Human c-myc binding protein (MBP-1) mRNA; complete cds
1168	L19182 HUMMAC25X Human MAC25 mRNA; complete cds
1194	D17517 HUMSKY Human sky mRNA for Sky; complete cds
1291	HG4322-HT4592 Tubulin, Beta
1423	V00567cds HSMGLO Human messenger RNA fragment for the beta-2 microglobulin
1570	M94250expanded Human retinoic acid inducible factor (MK) gene exons 1-5, complete cds
1664	J03040 Human SPARC/osteonectin mRNA, complete cds
1669	J04164 HUM927A Human interferon-inducible protein 9-27 mRNA; complete cds
1684	J02783mRNA HUMTHBP Human thyroid hormone binding protein (p55) mRNA; complete cds
1762	D00017 HUMLIC Homo sapiens mRNA for lipocortin II; complete cds
1822	U21689cds Human glutathione S-transferase-P1c gene; complete cds
1863	M93311cds Human metallothionein-III gene, complete cds
1871	M29386mRNA HUMPRLA Human prolactin mRNA; 3′ end
1949	HG1980-HT2023 Tubulin, Beta 2

Source: Ref. [70].

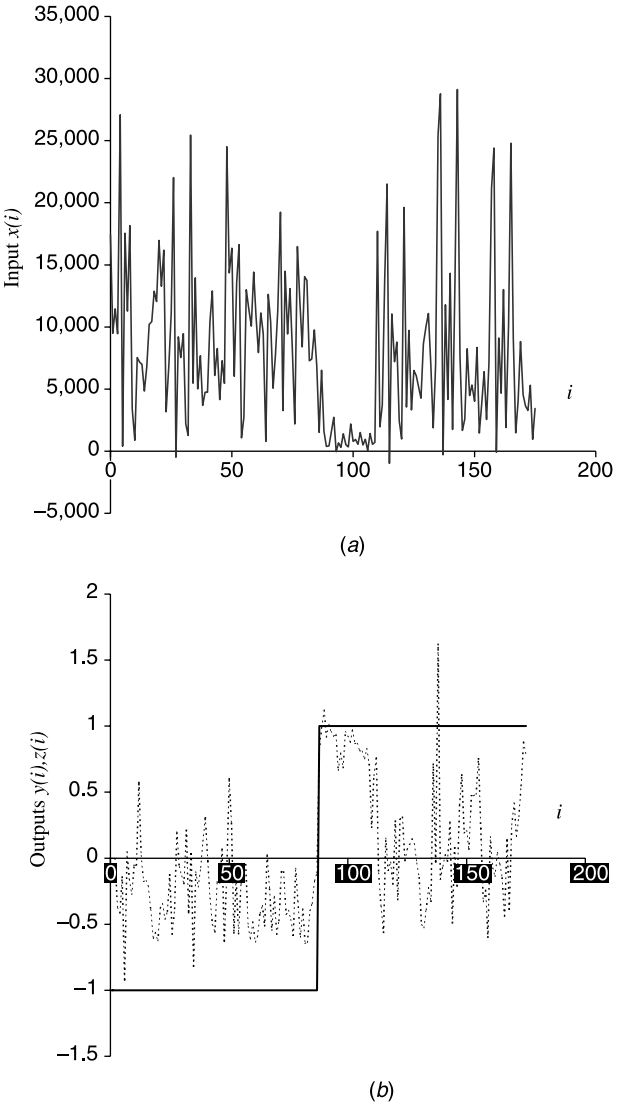


FIGURE 2.3. (*a*) Training input $x(i)$ formed by splicing together raw expression levels of genes from first four metastatic (M+) profiles and first four nonmetastatic (M0) profiles. The genes used (Table 2.2) were the 22 having greatest difference in expression levels between the M+ and M0 training profiles. (*b*) Training output $y(i)$ (solid line) defined as -1 over M+ portions of training input and 1 over M0 portions. The training input and output were used to identify a parallel cascade model of the form in Figure 2.1. The dashed line represents calculated output $z(i)$ when the identified model is stimulated by training input $x(i)$. Note that $z(i)$ is predominately negative over the M+ portions and positive over the M0 portions of the training input. The identified model's ability to separate metastatic and nonmetastatic profiles is exploited by replacing the profiles with corresponding model output signals that are easier to classify and predict metastasis. (From [70].)

four of five nonmetastatic tumors in an independent set that had also been used in [62]. Although the latter predictor may well be more reliable than the first predictor built using only the first 3 M0 and 3 M+ profiles from the original set, the independent set is not sufficiently large to show this. In fact, the first predictor achieved almost the same level of performance over the independent set, correctly classifying two of the three metastatic cell lines and four of the five nonmetastatic tumors.

2.6. COMPARING DIFFERENT PREDICTORS OVER THE SAME DATA SET

The immediately above remarks allude to a familiar problem with small data sets: One predictor might appear to show superior performance compared to another, but it is not clear that the improvement is significant. As Slonim [72] points out, because most data sets lack sufficient samples, generally only a few errors separate the winning and losing predictors.

Here we suggest a method of analyzing classifier performance that helps to make differences between competing predictors more apparent. The method was motivated by Pomeroy et al.'s presentation of the *k*-NN classification accuracy over subsets of patients with localized disease, low TRKC expression, and similar treatment regimens [61]. It was noted that, for example, not all patients with localized disease survived, yet *k*-NN prediction of outcome remained significant over this subset, indicating improvement over prediction based on metastatic staging [61].

As noted above, several methods have been used to predict clinical outcome in medulloblastoma, so we will focus now on this problem. Given two predictors, A and B, assume that neither correctly classifies all survivors or all failed outcomes. Then we suggest the following method of comparing the two predictors. Begin by splitting the data set into the subset predicted by A to be survivors and the subset predicted by A to fail treatment. Clearly A has no ability to distinguish survivors from failed outcomes within each subset. We then test whether B predictions over each subset positively correlate with actual outcome, obtaining the corresponding Fisher's exact test *P*-values. Assuming a positive correlation over each subset and treating the B performance over the two subsets as independent events, we can calculate the level of significance for the overall B prediction over two subsets where A has no predictive ability. Then the analysis is repeated, but reversing the roles of A and B. Often, this can reveal a major difference in performance by the two predictors.

In particular, we illustrate this approach by comparing *k*-NN and PCI performance over the 58 profiles used to test PCI. Of these profiles, PCI correctly classified 14 of 20 F and 32 of 38 S; for *k*-NN, 9 of 20 F and 36 of 38 S were correctly classified. (In addition, *k*-NN was correct on the first F and S profiles used to construct the PCI training input; these two profiles were not part of the PCI test set.) Table 2.3A shows PCI performance over the subset of profiles all classified as F and the subset all classified as S by *k*-NN. Over the first subset, PCI was flawless in distinguishing the nine F from the two S profiles (Matthews' correlation coefficient $\phi = 1$, $P < 0.0182$, one or two tails). Over the second subset, there was again a

TABLE 2.3 Comparing PCI and *k*-NN Performance by Splitting the Data Set Two Different Ways

A.	Subset Classified as F by *k*-NN		Subset Classified as S by *k*-NN		
	PCI Classifies as F	PCI Classifies as S		PCI Classifies as F	PCI Classifies as S
Actual S	0	2	Actual S	6	30
Actual F	9	0	Actual F	5	6
B.	**Subset Classified as F by PCI**		**Subset Classified as S by PCI**		
	k-NN Classifies as F	*k*-NN Classifies as S		*k*-NN Classifies as F	*k*-NN Classifies as S
Actual S	0	6	Actual S	2	30
Actual F	9	5	Actual F	0	6

positive correlation (coefficient $\phi = 0.29$, $P < 0.063$ one tail, $P < 0.098$ two tails) of PCI predictions with actual outcomes. Treating the PCI performance over the two subsets as independent events, we calculate the probability of achieving by chance the observed accuracy or better over both of these subsets is less than $0.02 \times 0.1 = 0.002$.

In contrast, Table 2.3B shows k-NN performance over the subset of profiles all classified as F and the subset all classified as S by PCI. Note that this portion of the table can be set down by inspection of the upper portion: The two outer number columns stay the same while the two inner columns switch. Over the first of the latter subsets, there was a positive correlation ($\phi = 0.59$, $P < 0.012$ one tail, $P < 0.0141$ two tails) of k-NN predictions with actual outcomes. However, over the second subset, the correlation of k-NN predictions with actual outcomes was *negative* ($\phi = -0.1$, not significant).

We conclude that when the test data set is split into the two subsets within which k-NN has no ability to distinguish F from S profiles, PCI outcome prediction still correlates positively with actual outcome within both subsets, and its overall performance is significant at better than $P < 0.002$. However, when the data set is split into the two subsets within which PCI has no predictive ability, k-NN outcome prediction does not correlate positively with actual outcome within both subsets but only correlates positively within the smaller subset. This represents a clear difference between k-NN and positively performance.

2.7. CONCLUDING REMARKS

While this chapter has focused on uses of PCI, we conclude with a few remarks about FOS. As noted above, the latter is a general-purpose method of searching through a candidate set of basis functions to build a concise model of a system, where computation time scales linearly with number of candidate functions. Introduced in 1987 [41], FOS has been applied in system identification [42, 73], in time-series analysis [42, 74], and within an iterative version, to build generalized single-layer ANNs, where it determined model size as well as its significant terms [46]. Applications of FOS have included Raman spectral estimation [75] and detection of abnormalities in prosthetic heart valves [76]. It is interesting that FOS is actually more efficient than a similar algorithm published later [77] that has been extensively used in the neural network field. This point is discussed in [46].

This chapter began with a consideration of blackbox methods of building models that approximate the input output behavior of a given nonlinear system. It was then shown that these approaches are well suited to constructing effective classifiers in the proteomic and genomic areas. One encouraging aspect of this work was illustrated in predicting medulloblastoma clinical outcome; namely, PCI classifiers combine well with other predictors to achieve accuracy beyond that of any of the individual methods. Indeed, developing ways for predictors to cooperate is likely to be a fruitful line of enquiry in genomics and proteomics research.

REFERENCES

1. V. Volterra, *Leçons sur les Fonctions de Lignes*, Gauthier-Villars, Paris, 1913.
2. V. Volterra, *Theory of Functionals and of Integral and Integro-Differential Equations*, Dover, New York, 1959.
3. N. Wiener, *Nonlinear Problems in Random Theory*, MIT Press, Cambridge, MA, 1958.
4. M. Fréchet, "Sur les fonctionnelles continues," *Ann. Sci. l'Ecole Normal Supérieure*, 27: 193–219, 1910.
5. S. Boyd and L. O. Chua, "Fading memory and the problem of approximating non-linear operators with Volterra series," *IEEE Trans. Circ. Sys.*, 32: 1150–1160, 1985.
6. Y. W. Lee and M. Schetzen, "Measurement of the Wiener kernels of a non-linear system by crosscorrelation," *Int. J. Contr.*, 2: 237–254, 1965.
7. G. Palm, "On representation and approximation of nonlinear systems. Part II: Discrete time," *Biol. Cybernet.*, 34: 49–52, 1979.
8. A. S. French and E. G. Butz, "Measuring the Wiener kernels of a non-linear system using the fast Fourier transform algorithm," *Int. J. Control*, 17: 529–539, 1973.
9. J. Amorocho and A. Brandstetter, "Determination of nonlinear functional response functions in rainfall runoff processes," *Water Resources Res.*, 7: 1087–1101, 1971.
10. A. Watanabe and L. Stark, "Kernel method for nonlinear analysis: Identification of a biological control system," *Math. Biosci.*, 27: 99–108, 1975.
11. V. Z. Marmarelis, "Identification of nonlinear biological systems using Laguerre expansions of kernels," *Ann. Biomed. Eng.*, 21: 573–589, 1993.
12. H. Ogura, "Estimation of Wiener kernels of a nonlinear system and fast algorithm using digital Laguerre filters," in K.-I. Naka and Y.-I. Ando (Eds.), *Proceedings of The Fifteenth NIBB Conference on Information Processing in Neuron Network: White Noise Analysis*, National Institute for Basic Biology, Okazaki, Japan, 1986, pp. 14–62.
13. M. J. Korenberg, "Identifying nonlinear difference equation and functional expansion representations: The fast orthogonal algorithm," *Ann. Biomed. Eng.*, 16: 123–142, 1988.
14. M. J. Korenberg, "Parallel cascade identification and kernel estimation for nonlinear systems," *Ann. Biomed. Eng*., 19: 429–455, 1991.
15. M. J. Korenberg, "Statistical identification of parallel cascades of linear and nonlinear systems," *IFAC Symp. Ident. Sys. Param. Est.*, 1: 580–585, 1982.
16. A. Sandberg and L. Stark, "Wiener G-function analysis as an approach to nonlinear characteristics of human pupil light reflex," *Brain Res.*, 11: 194–211, 1968.
17. L. W. Stark, "The pupillary control system: Its nonlinear adaptive and stochastic engineering design characteristics," *Automatica*, 5: 655–676, 1969.
18. P. Z. Marmarelis and K.-I. Naka, "White-noise analysis of a neuron chain: An application of the Wiener theory," *Science*, 175: 1276–1278, 1972.
19. H. M. Sakai and K.-I. Naka, "Signal transmission in the catfish retina. IV. Transmission to ganglion cells," *J. Neurophysiol.*, 58: 1307–1328, 1987.
20. H. M. Sakai and K.-I. Naka, "Signal transmission in the catfish retina. V. Sensitivity and circuit," *J. Neurophysiol.*, 58: 1329–1350, 1987.
21. M. Barahona and C.-S. Poon, "Detection of nonlinear dynamics in short, noisy time series," *Nature*, 381: 215–217, 1996.

22. S. Orcioni, M. Pirani, C. Turchetti, and M. Conti, "Practical notes on two Volterra filter identification direct methods," *Proc. IEEE Int. Symp. Circuits Sys.*, 3: 587–590, 2002.
23. Q. Zhang, B. Suki, D. T. Westwick, and K. R. Lutchen, "Factors affecting Volterra kernel estimation: Emphasis on lung tissue viscoelasticity," *Ann. Biomed. Eng.*, 26: 103–116, 1998.
24. H. Akaike, "Fitting autoregressive models for prediction," *Ann. Inst. Stat. Math.*, 21: 243–347, 1969.
25. H. Akaike, "A new look at the statistical model identification," *IEEE Trans. Automat. Control*, AC-19: 716–723, 1974.
26. D. T. Westwick, B. Suki, and K. R. Lutchen, "Sensitivity analysis of kernel estimates: Implications in nonlinear physiological system identification," *Ann. Biomed. Eng.*, 26: 488–501, 1998.
27. D. T. Westwick and R. E. Kearney, "Nonparametric identification of nonlinear biomedical systems, Part I: Theory," *CRC Crit. Rev. Biomed. Eng.*, 26: 153–226, 1998.
28. E. E. Sutter, "A practical non-stochastic approach to nonlinear time-domain analysis," in V. Z. Marmarelis (Ed.), *Advanced Methods of Physiological System Modeling*, Vol. 1, Biomedical Simulations Resource, University of Southern California, Los Angeles, CA, 1987, pp. 303–315.
29. E. E. Sutter, "A deterministic approach to nonlinear systems analysis," in R. B. Pinter and B. Nabet (Eds.), *Nonlinear Vision: Determination of Neural Receptive Fields, Function, and Networks*, CRC Press, Boca Raton, FL, 1992, pp. 171–220.
30. H. Spekreijse, "Rectification in the goldfish retina: Analysis by sinusoidal and auxiliary stimulation," *Vision Res.*, 9: 1461–1472, 1969.
31. H. Spekreijse and H. Oosting, "Linearizing: A method for analyzing and synthesizing nonlinear systems," *Kybernetik*, 7: 22–31, 1970.
32. M. J. Korenberg, "Cross-correlation analysis of neural cascades," *Proc. 10th Ann. Rocky Mountain Bioeng. Symp.*, 1: 47–52, 1973.
33. M. J. Korenberg, "Identification of biological cascades of linear and static nonlinear systems," *Proc. 16th Midwest Symp. Circuit Theory*, 18.2: 1–9, 1973.
34. A. S. French and M. J. Korenberg, "A nonlinear cascade model for action potential encoding in an insect sensory neuron," *Biophys. J.*, 55: 655–661, 1989.
35. A. S. French and M. J. Korenberg, "Dissection of a nonlinear cascade model for sensory encoding," *Ann. Biomed. Eng.*, 19: 473–484, 1991.
36. R. C. Emerson, M. J. Korenberg, and M. C. Citron, "Identification of complex-cell intensive nonlinearities in a cascade model of cat visual cortex," *Biol. Cybernet.*, 66: 291–300, 1992.
37. M. J. Korenberg, "Identifying noisy cascades of linear and static nonlinear systems," *IFAC Symp. Ident. Sys. Param. Est.*, 1: 421–426, 1985.
38. S. A. Billings and S. Y. Fakhouri, "Identification of systems containing linear dynamic and static nonlinear elements," *Automatica*, 18: 15–26, 1982.
39. H. H. Sun and J. H. Shi, "New algorithm for Korenberg-Billings model of nonlinear system identification," in V. Z. Marmarelis (Ed.), *Advanced Methods of Physiological System Modeling*, Vol. 2, Plenum, New York, 1989, pp. 179–200.
40. M. J. Korenberg, J. E. Solomon, and M. E. Regelson, "Parallel cascade identification as a means for automatically classifying protein sequences into structure/function groups," *Biol. Cybernet.*, 82: 15–21, 2000.

41. M. J. Korenberg, "Fast orthogonal identification of nonlinear difference equation and functional expansion models," *Proc. Midwest Symp. Circuit Sys.*, 1: 270–276, 1987.
42. M. J. Korenberg, "A robust orthogonal algorithm for system identification and time-series analysis," *Biol. Cybernet.*, 60: 267–276, 1989.
43. A. A. Desrochers, "On an improved model reduction technique for nonlinear systems," *Automatica*, 17(2): 407–409, 1981.
44. K. M. Adeney and M. J. Korenberg, "Fast orthogonal search for direction finding," *Electron. Lett.*, 28(25): 2268–2269, 1992.
45. K. M. Adeney and M. J. Korenberg, "Fast orthogonal search for array processing and spectrum estimation," *IEE Proc. Vision Image Signal Process.*, 141(1): 13–18, 1994.
46. K. M. Adeney and M. J. Korenberg, "Iterative fast orthogonal search algorithm for MDL-based training of generalized single-layer networks," *Neural Networks*, 13: 787–799, 2000.
47. D. R. McGaughey, M. J. Korenberg, K. M. Adeney, S. D. Collins, and G. J. M. Aitken, "Using the fast orthogonal search with first term reselection to find subharmonic terms in spectral analysis," *Ann. Biomed. Eng.*, 31: 741–751, 2003.
48. K. H. Chon, "Accurate identification of periodic oscillations buried in white or colored noise using fast orthogonal search," *IEEE Trans. Biomed. Eng.*, 48(10): 622–629, 2001.
49. M. O. Sunay and M. M. Fahmy, "An orthogonal approach to the spatial-domain design of 2-D recursive and nonrecursive nonlinear filters," *IEEE Trans. Circuits Sys.*, 41: 669–677, 1994.
50. Y.-T. Wu, M. Sun, D. Krieger, and R. J. Sclabassi, "Comparison of orthogonal search and canonical variate analysis for the identification of neurobiological systems," *Ann. Biomed. Eng.*, 27(5): 592–606, 1999.
51. M. J. Korenberg, R. David, I. W. Hunter, and J. E. Solomon, "Automatic classification of protein sequences into structure/function groups via parallel cascade identification: A feasibility study," *Ann. Biomed. Eng.*, 28: 803–811, 2000.
52. G. D. Rose, A. R. Geselowitz, G. J. Lesser, R. H. Lee, and M. H. Aehfus. "Hydrophobicity of amino acid residues in globular proteins," *Science*, 229: 834–838, 1985.
53. R. Hughey, K. Karplus, and A. Krogh, "Sequence alignment and modeling software system," http://www.cse.ucsc.edu/research/compbio/sam.html, 1999.
54. R. David, M. J. Korenberg, and I. W. Hunter, "3D-1D threading methods for protein fold recognition," *Pharmacogenomics*, 1(4): 445–455, 2000.
55. I. V. Grigoriev and S.-H. Kim, "Detection of protein fold similarity based on correlation of amino acid properties," *Proc. Natl. Acad. Sci. USA*, 96: 14318–14323, 1999.
56. M. J. Korenberg, E. D. Lipson, J. R. Green, and J. E. Solomon, "Parallel cascade recognition of exon and intron DNA sequences," *Ann. Biomed. Eng.*, 30: 129–140, 2002.
57. J. R. Green, M. J. Korenberg, R. David, and I. W. Hunter, "Recognition of adenosine triphosphate binding sites using parallel cascade system identification," *Ann. Biomed. Eng.*, 31: 462–470, 2003.
58. J. Khan, J. S. Wei, M. Ringnér, L. H. Saal, M. Ladanyi, F. Westermann, F. Berthold, M. Schwab, C. R. Antonescu, C. Peterson, and P. S. Meltzer, "Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks," *Nature Med.*, 7: 673–679, 2001.

59. T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, M. A. Caligiuri, C. D. Bloomfield, and E. S. Lander, "Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring," *Science*, 286: 531–537, 1999. Datasets: http://wwwgenome.wi.mit.edu/MPR/data_set_ALL_AML.html.

60. A. Brazma and J. Vilo, "Gene expression data analysis," *FEBS Lett.*, 480: 17–24, 2000.

61. S. L. Pomeroy, P. Tamayo, M. Gaasenbeek, L. M. Sturla, M. Angelo, M. E. McLaughlin, J. Y. H. Kim, L. C. Goumneroval, P. M. Black, C. Lau, J. C. Allen, D. Zagzag, J. M. Olson, T. Curran, C. Wetmore, J. A. Biegel, T. Poggio, S. Mukherjee, R. Rifkin, A. Califano, G. Stolovitzky, D. N. Louis, J. P. Mesirov, E. S. Lander, and T. R. Golub, "Prediction of central nervous system embryonal tumour outcome based on gene expression," *Nature*, 415: 436–442, 2002. Supplementary information and datasets: http://www.genome.wi.mit.edu/MPR/CNS.

62. T. J. MacDonald, K. M. Brown, B. LaFleur, K. Peterson, C. Lawlor, Y. Chen, R. J. Packer, P. Cogen, D. A. Stephan, "Expression profiling of medulloblastoma: PDGFRA and the RAS/MAPK pathway as therapeutic targets for metastatic disease," *Nature Genet.*, 29: 143–152, 2001. Datasets: http://microarray.cnmcresearch.org.

63. M. J. Korenberg, "Prediction of treatment response using gene expression profiles," *J. Proteome Res.*, 1: 55–61, 2002.

64. A. Schuster, W. Dubitzky, F. J. Azuaje, M. Granzow, D. Berrar, and R. Eils, "Tumor identification by gene expression profiles: A comparison of five different clustering methods," Critical Assessment of Microarray Data Analysis *CAMDA'00*, http://bioinformatics.duke.edu/camda/CAMDA00/Abstracts/Schuster.asp, 2000.

65. P. Kirkpatrick, "Look into the future," *Nature Rev. Drug Discovery*, 1(5): 334, 2002.

66. M. Bredel, C. Bredel, and B. I. Sikic, "Genomics-based hypothesis generation: A novel approach to unravelling drug resistance in brain tumours?" *Lancet Oncol.*, 5: 89–100, 2004.

67. M. J. Korenberg, "Gene expression monitoring accurately predicts medulloblastoma positive and negative clinical outcomes," *FEBS Lett.*, 533: 110–114, 2003.

68. L. J. van't Veer, H. Dai, M. J. van de Vijver, Y. D. He, A. A. M. Hart, M. Mao, H. L. Peterse, K. van der Kooy, M. J. Marton, A. T. Witteveen, G. J. Schreiber, R. M. Kerkhoven, C. Roberts, P. S. Linsley, R. Bernards, and S. H. Friend, "Gene expression profiling predicts clinical outcome of breast cancer," *Nature*, 415: 530–536, 2002.

69. R. J. Tibshirani and B. Efron, "Pre-validation and inference in microarrays," *Statist. Applicat. Genet. Mol. Biol.*, 1(1), article 1, 2002.

70. M. J. Korenberg, "On predicting medulloblastoma metastasis by gene expression profiling," *J. Proteome Res.*, 3: 91–96, 2004.

71. B. W. Matthews, "Comparison of the predicted and observed secondary structure of T4 phage lysozyme," *Biochim. Biophys. Acta*, 405: 442–451, 1975.

72. D. K. Slonim, "From patterns to pathways: gene expression data analysis comes of age," *Nature Genet.*, 32(Suppl.): 502–508, 2002.

73. K. H. Chon, M. J. Korenberg, and N. H. Holstein-Rathlou, "Application of fast orthogonal search to linear and nonlinear stochastic systems," *Ann. Biomed. Eng.*, 25: 793–801, 1997.

74. M. J. Korenberg and L. D. Paarmann, "Application of fast orthogonal search: Time-series analysis and resolution of signals in noise," *Ann. Biomed. Eng.*, 17: 219–231, 1989.

75. M. J. Korenberg, C. J. Brenan, and I. W. Hunter, "Raman spectral estimation via fast orthogonal search," *Analyst*, 122: 879–882, 1997.

76. S. H. Kim, H. J. Lee, J. M. Huh, and B. C. Chang, "Spectral analysis of heart valve sound for detection of prosthetic heart valve diseases," *Yonsei Med. J.*, 39: 302–308, 1998.

77. S. Chen and J. Wigger, "Fast orthogonal least squares algorithm for efficient subset model selection," *IEEE Trans. Signal Process.*, 43(7): 1713–1715, 1995.

CHAPTER 3

Gene Regulation Bioinformatics of Microarray Data

GERT THIJS, FRANK DE SMET, YVES MOREAU, KATHLEEN MARCHAL, and BART DE MOOR

3.1. INTRODUCTION

Unraveling the mechanisms that regulate gene activity in an organism is a major goal of molecular biology. In the past few years, microarray technology has emerged as an effective technique to measure the level of expression of thousands of genes in a single experiment. Because of their capacity to monitor many genes, microarrays are becoming the workhorse of molecular biologists studying gene regulation. However, these experiments generate data in such amount and of such a complexity that their analysis requires powerful computational and statistical techniques. As a result, unraveling gene regulation from microarray experiments is currently one of the major challenges of bioinformatics.

Starting from microarray data, a first major computational task is to cluster genes into biologically meaningful groups according to their pattern of expression [1]. Such groups of related genes are much more tractable for study by biologists than the full data itself. As in many other applications in biology, the *guilt-by-association* principle is adopted to extract usable knowledge from the full data. Classical clustering techniques such as hierarchical clustering [2] or *K*-means [3] have been applied to microarray data. Yet the specificity of microarray data (such as the high level of noise or the link to extensive biological information) have created the need for clustering methods specifically tailored to this type of data [4]. Here both the first generation of clustering methods applied to microarray data as well as second-generation algorithms, which are more specific to microarray data, are reviewed. In particular, we address a number of shortcomings of classical clustering algorithms with a new method called adaptive quality-based clustering (AQBC) [5] in which we look for tight reliable clusters.

Genomics and Proteomics Engineering in Medicine and Biology. Edited by Metin Akay

In a second step, the question asked is what makes genes belong to the same cluster. A main cause of coexpression of genes is that these genes share the same regulation mechanism at the sequence level. Specifically, some control regions, *promoter regions*, in the neighborhood of the genes will contain specific short sequence patterns, called *binding sites*, which are recognized by activating or repressing proteins, called *transcription factors*. In such a situation, we say that the genes are transcriptionally regulated. Switching our attention from expression data to sequence data, we consider algorithms that discover such binding sites in sets of DNA sequences from coexpressed genes. We analyze the upstream region of those genes to detect patterns, also called *motifs*, that are statistically overrepresented when compared to some random model of the sequence. The detection of overrepresented patterns in DNA or aminoacid sequences is called *motif finding*.

Two classes of methods are available for motif finding: word-counting methods and probabilistic sequence models. Word-counting methods are string-matching methods based on counting the number of occurrences of each DNA word (called *oligonucleotide*) and comparing this number with the expected number of occurrences based on some statistical model. Probabilistic sequence models build a likelihood function for the sequences based on the motif occurrences and a model of the background sequence. Probabilistic optimization methods, such as expectation–maximization (EM) and Gibbs sampling, are then used to search for good configurations (motif model and positions). After briefly presenting the word-counting methods and the method based on EM, we discuss the basic principles of Gibbs sampling for motif finding more thoroughly. We also present our Gibbs sampling method, called MotifSampler, where we have introduced a number of extensions to improve Gibbs sampling for motif finding, such as the use of a more precise model of the sequence background based on higher order Markov chains. This improved model increases the robustness of the method significantly.

These two steps, clustering and motif finding, are interlocked and specifically dedicated to the discovery of regulatory motifs from microarray experiments. In particular, clustering needs to take into account that motif finding is sensitive to noise. Therefore, we need clustering methods that build conservative clusters for which coexpression can be guaranteed in an attempt to increase the proportion of coregulated genes in a cluster. This is one of the requirements that warranted the development of AQBC. Also the motif-finding algorithms are specifically tailored to the discovery of transcription factor binding motifs (while related algorithms can be developed for slightly different problems in protein sequence analysis). These tight links mandate our integrated presentation of these two topics in this chapter. Furthermore, the same links call for integrated software tools to handle this task in an efficient manner. Our INCLUSive Web tool (http://homes.esat.kuleuven.be/~dna/BioI/Software.html) supports motif finding from microarray data. Starting with the clustering of microarray data by AQBC, it then retrieves the DNA sequences relating to the genes in a cluster in a semiautomated fashion and finally performs motif finding using our MotifSampler (see Fig. 3.1). Integration

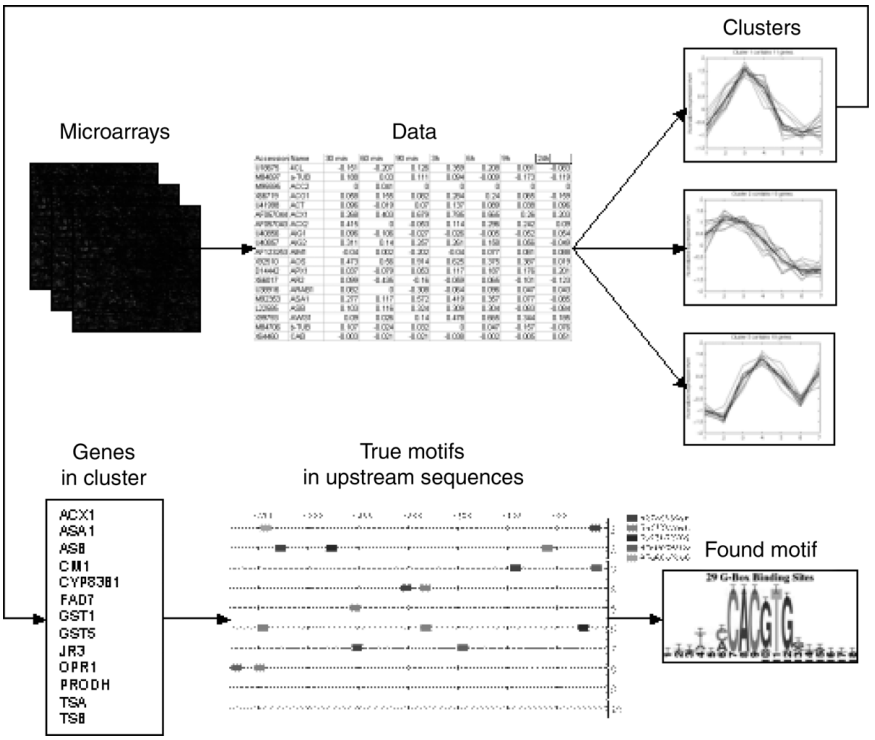


FIGURE 3.1. High-level description of data analysis for motif finding from microarray data. The analysis starts from scanned microarray images. After proper quantification and preprocessing, the data are available for clustering in the form of a data matrix. Clustering then determines clusters of potentially coregulated genes. Focusing on a cluster of genes of interest, motif finding analyzes the sequences of the control regions of the genes in the cluster. A number of true motifs are present in those sequences, but they are unknown. Motif finding analyzes those sequences for statistically overrepresented DNA patterns. Finally, candidate motifs are returned by the motif-finding algorithm and are available for further biological evaluation.

is paramount in bioinformatics as, by optimally matching the different steps of the data analysis to each other, the total analysis becomes more effective than the sum of its parts.

3.2. INTRODUCTION TO TRANSCRIPTIONAL REGULATION

To situate the problem at hand, we present in this section concisely the main concepts from biology relevant to our discussion of measuring gene expression data and motif finding in DNA sequences.

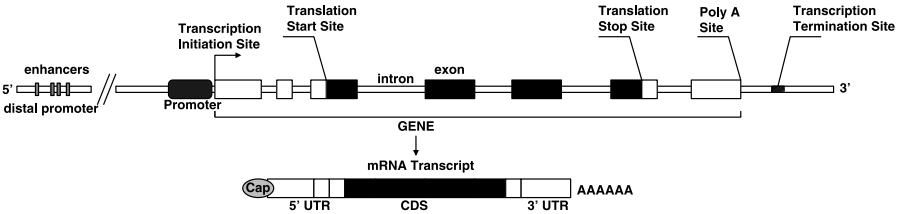


FIGURE 3.2. Structure of eukaryotic gene and single-stranded mRNA transcribed from gene.

3.2.1. Structure of Genes

Genes are segments of DNA that encode for proteins through the intermediate action of messenger RNA (mRNA). In Figure 3.2, the structure of a eukaryotic gene is displayed. A gene and the genomic region surrounding it consist of a transcribed sequence, which is converted into an mRNA transcript, and of various *untranscribed* sequences. The mRNA consists of a coding sequence that is translated into a protein and of several *untranslated* regions (UTRs). The untranscribed sequences and the UTRs play a major role in the regulation of expression. Notably, the *promoter region* in front of the transcribed sequence contains the *binding sites* for the *transcription factor* proteins that start up transcription. Moreover, the region upstream of the transcription start contains many binding sites for transcription factors that act as *enhancers* and *repressors* of gene expression (although some transcription factors can bind outside this region).

3.2.2. Transcription

Transcription means the assembly of ribonucleotides into a single strand of mRNA. The sequence of this strand of mRNA is dictated by the order of the nucleotides in the transcribed part of the gene. The transcription process is initiated by the binding of several transcription factors to regulatory sites in the DNA, usually located in the promoter region of the gene. The transcription factor proteins bind each other to form a complex that associates with an enzyme called RNA polymerase. This association enables the binding of RNA polymerase to a specific site in the promoter. In Figure 3.3, the initiation of the transcription process is shown. Together, the complex of transcription factors and the RNA polymerase unravel the DNA and separate both strands. Subsequently, the polymerase proceeds down on one strand while it builds up a strand of mRNA complementary to the DNA, until it reaches the terminator sequence. In this way, an mRNA is produced that is complementary to the transcribed part of the gene. Then, the mRNA transcript detaches from the RNA polymerase and the polymerase breaks its contact with the DNA. In a later stage, the mRNA is processed, transported out of the nucleus, and translated into a protein.

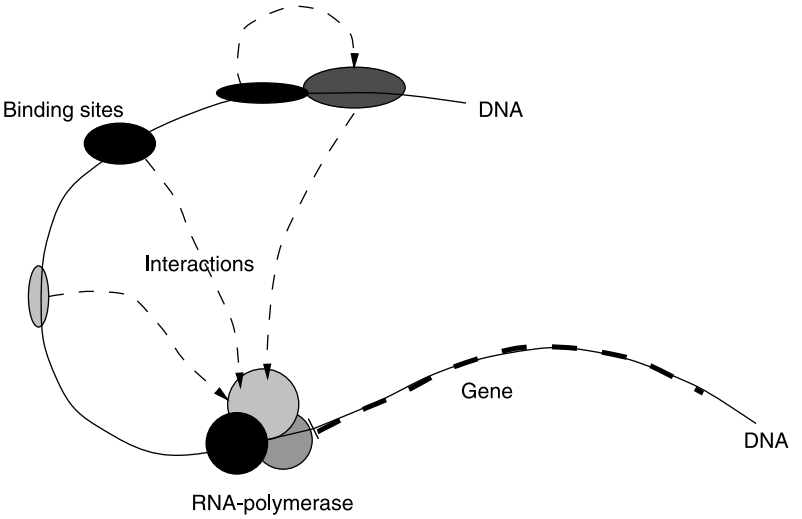


FIGURE 3.3. Schematic representation of initiation of transcription process by association of complex of transcription factors (gene regulatory proteins), RNA polymerase, and promoter region of gene.

3.2.3. Transcription Factors

Transcription factors are proteins that bind to regulatory sequences on eukaryotic chromosomes, thereby modifying the rate of transcription of a gene. Some transcription factors bind directly to specific transcription factor binding sites (TFBSs) in the DNA (promoters, enhancers, and repressors), others bind to each other. Most of them bind both to the DNA as well as to other transcription factors. It should be noted that the transcription rate can be positively or negatively affected by the action of transcription factors. When the transcription factor significantly decreases the transcription of a gene, it is called a repressor. If, on the other hand, the expression of a gene is upregulated, biologists speak of an enhancer.

3.3. MEASURING GENE EXPRESSION PROFILES

Cells produce the proteins they need to function properly by (1) *transcribing* the corresponding genes from DNA into mRNA transcripts and (2) *translating* the mRNA molecules into proteins. Microarrays obtain a snapshot of the activity of a cell by deriving a measurement from the number of copies of each type of mRNA molecule (which also gives an indirect and imperfect picture of the protein activity). The key to this measurement is the double-helix *hybridization* properties of DNA (and RNA). When a single strand of DNA is brought in

contact with a complementary DNA (cDNA) sequence, it will anneal to this complementary sequence to form double-stranded DNA. For the four DNA bases, adenine is complementary to thymine and guanine is complementary to cytosine. Because both strands have opposite orientations, the complementary sequence is produced by complementing the bases of the reference sequence starting from the end of this sequence and proceeding further upstream. Hybridization will therefore allow a DNA probe to recognize a copy of its complementary sequence obtained from a biological sample.

An array consists of a reproducible pattern of different DNA probes attached to a solid support. After RNA extraction from a biological sample, fluorescently labeled cDNA or cRNA is prepared. This fluorescent sample is then hybridized to the DNA present on the array. Thanks to the fluorescence, hybridization intensities (which are related to the number of copies of each RNA species present in the sample) can be measured by a laser scanner and converted into a quantitative readout. In this way, microarrays allow simultaneous measurement of expression levels of thousands of genes in a single hybridization assay.

Two basic array technologies are currently available: cDNA microarrays and gene chips. Complementary DNA microarrays [6] are small glass slides on which double-stranded DNA is spotted. These DNA fragments are normally several hundred base pairs in length and are often derived from reference collections of expressed sequence tags (ESTs) extracted from many sources of biological materials so as to represent the largest possible number of genes. Usually each spot represents a single gene. The cDNA microarrays use *two* samples: a reference and a test sample (e.g., normal vs. malignant tissue). A pair of cDNA samples is independently copied from the corresponding mRNA populations with the reverse transcriptase enzyme and labeled using distinct fluorescent molecules (green and red). These labeled cDNA samples are then pooled and hybridized to the array. Relative amounts of a particular gene transcript in the two samples are determined by measuring the signal intensities detected at both fluorescence wavelengths and calculating the ratios (here, only relative expression levels are obtained). A cDNA microarray is therefore a differential technique, which intrinsically normalizes for part of the experimental noise. An overview of the procedure that can be followed with cDNA microarrays is given in Figure 3.4.

GeneChip oligonucleotide arrays (Affymetrix, Inc., Santa Clara, CA) [7] are high-density arrays of oligonucleotides synthesized using a photolithograpic technology similar to microchip technology. The synthesis uses in situ light-directed chemistry to build up hundreds of thousands of different oligonucleotide probes (25-mer). Each gene is represented by 15 to 20 different oligonucleotides, serving as unique sequence-specific detectors. In addition, mismatch control oligonucleotides (identical to the perfect-match probes except for a single base-pair mismatch) are added. These control probes allow estimation of cross hybridization and significantly decrease the number of false positives. With this technology, absolute expression levels are obtained (no ratios).

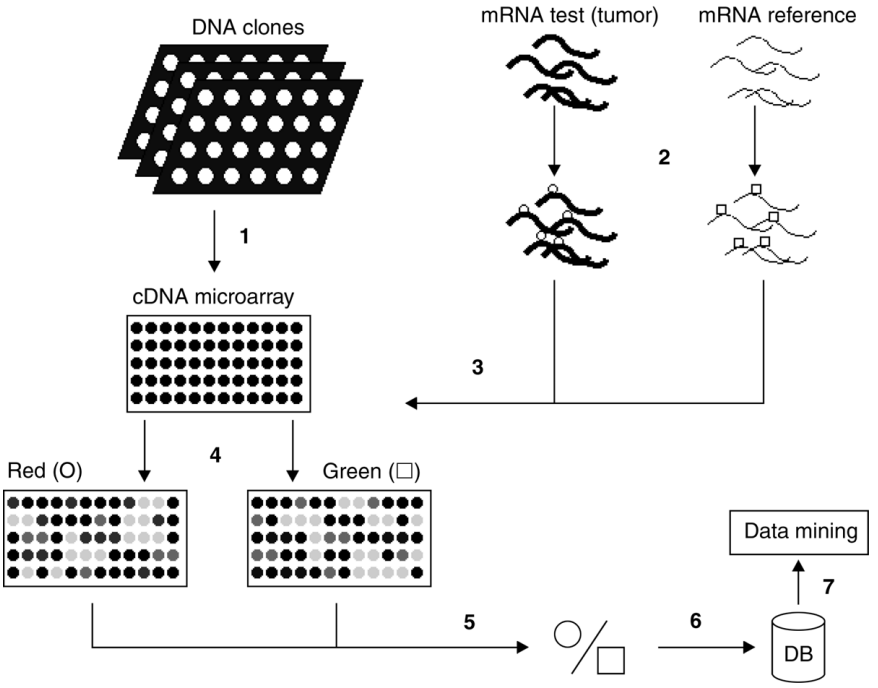


FIGURE 3.4. Schematic overview of experiment with cDNA microarray. (1) Spotting of presynthesized DNA probes (derived from genes to be studied) on glass slide. These probes are the purified products of the polymerase chain reaĉtion (PCR) amplification of the associated DNA clones. (2) Labeling (via reverse transcriptase) of total mRNA of test sample (red channel, 0) and reference sample (green channel, □). (3) Pooling of two samples and hybridization. (4) Readout of red and green intensities separately (measure for hybridization by test and reference sample) in each probe. (5) Calculation of relative expression levels (intensity in red channel divided by intensity in green channel). (6) Storage of results in database. (7) Data mining.

3.4. PREPROCESSING OF DATA

Before any substantial analysis can be performed on a set of microarray data, it is necessary to preprocess the data. A correct preprocessing strategy is almost as important as the cluster analysis itself.

3.4.1. Why Preprocessing Is Essential

First, it is necessary to normalize the hybridization intensities within a single array experiment. In a two-channel cDNA microarray experiment, for example, normalization adjusts for differences in labeling, detection efficiency, and the quantity of initial RNA within the two channels [1, 8, 9]. Normalization is necessary before

one can compare the results from different microarray experiments [9]. Second, transformation of the data using a nonlinear function (often the logarithm is used, especially for two-channel cDNA microarray experiments where the values are expression ratios) can be useful [1]. Third, expression data often contain numerous missing values and many clustering algorithms are unable to deal with them [10]. It is therefore imperative either to use appropriate procedures that can estimate and replace these missing values or to adapt existing clustering algorithms, enabling them to handle missing values directly (without actually replacing them [5, 11]). Fourth, it is common to (crudely) filter the gene expression profiles (removing the profiles that do not satisfy some simple criteria) before proceeding with the actual clustering [2]. A fifth preprocessing step is standardization or rescaling of the gene expression profiles (e.g., multiplying every expression vector with a scale factor so that its length is 1 [1]). This makes sense because the aim is to cluster gene expression profiles with the same relative behavior (expression levels go up and down at the same time) and not only the ones with the same absolute behavior.

3.4.2. Preprocessig Steps

Let us now look at some of those preprocessing steps in more detail.

Normalization The first step is the normalization of the hybridization intensities within a single array experiment. In a two-channel cDNA microarray experiment, several sources of noise (such as differences in labeling, detection efficiency, and the quantity of initial RNA within the two channels) create systematic sources of biases. The biases can be computed and removed to correct the data. As many sources can be considered and as they can be estimated and corrected in a variety of ways, many different normalization procedures exist. We therefore do not cover this topic further here and refer to [1] for more details.

Nonlinear Transformations It is common practice to pass expression values through a nonlinear function. Often the logarithm is used for this nonlinear function. This is especially suited when dealing with expression ratios (coming from two-channel cDNA microarray experiments, using a test and reference sample) since expression ratios are not symmetrical [1]. Upregulated genes have expression ratios between 1 and infinity, while downregulated genes have expression ratios squashed between 1 and 0. Taking the logarithms of these expression ratios results in symmetry between expression values of up- and downregulated genes.

Missing-Value Replacement Microarray experiments often contain missing values (measurements absent because of technical reasons). The inability of many cluster algorithms to handle such missing values necessitates the replacement of these values. Simple replacements such as a replacement by 0 or by the average of the expression profile often disrupt these profiles. Indeed replacement by average values relies on the unrealistic assumption that all

expression values are similar across different experimental conditions. Because of an erroneous missing-value replacement, genes containing a high number of missing values can be assigned to the wrong cluster. More advanced techniques of missing-value replacement have been described [10, 12] that take advantage of the rich information provided by the expression patterns of other genes in the data set. Finally, note that some implementations of algorithms only make use of the measured values to derive the clusters and as such obviate the need for missing-value replacement [5].

Filtering As stated in the overview section, a set of microarray experiments, generating gene expression profiles, frequently contains a considerable number of genes that do not really contribute to the biological process that is being studied. The expression values of these profiles often show little variation over the different experiments (they are called constitutive with respect to the biological process studied). Moreover, these constitutive genes will have seemingly random and meaningless profiles after standardization (division by a small standard deviation results in noise inflation), which is also a common preprocessing step (see below). Another problem comes from highly unreliable expression profiles containing many missing values. The quality of the clusters would significantly degrade if these data were passed to the clustering algorithms as such. Filtering [2] removes gene expression profiles from the data set that do not satisfy some simple criteria. Commonly used criteria include a minimum threshold for the standard deviation of the expression values in a profile (removal of constitutive genes) and a threshold on the maximum percentage of missing values.

Standardization or Rescaling Biologists are mainly interested in grouping gene expression profiles that have the same relative behavior; that is, genes that are up- and downregulated together. Genes showing the same relative behavior but with diverging absolute behavior (e.g., gene expression profiles with a different baseline or a different amplitude but going up and down at the same time) will have a relatively high Euclidean distance. Cluster algorithms based on this distance measure will therefore wrongfully assign these genes to different clusters. This effect can largely be prevented by applying standardization or rescaling to the gene expression profiles to have zero mean and unit standard deviation. Gene expression profiles with the same relative behavior will have a smaller Euclidean distance after rescaling [1].

3.5. CLUSTERING OF GENE EXPRESSION PROFILES

Using microarrays, we can measure the expression levels of thousands of genes simultaneously. These expression levels can be determined for samples taken at different time points during a certain biological process (e.g., different phases of the cycle of cell division) or for samples taken under different conditions (e.g., cells originating from tumor samples with a different histopathological diagnosis). For each gene,

the arrangement of these measurements into a (row) vector leads to what is generally called an expression profile. These expression profiles or vectors can be regarded as data points in a high-dimensional space.

3.5.1. Rationale of Clustering Expression Profiles

Because relatedness in biological function often implies similarity in expression behavior (and vice versa) and because several genes might be involved in the process under study, it will be possible to identify subgroups or clusters of genes that will have similar expression profiles (i.e., according to a certain distance function, the associated expression vectors are sufficiently close to one another). Genes with similar expression profiles are said to be coexpressed. Conversely, coexpression of genes can thus be an important observation to infer the biological role of these genes. For example, coexpression of a gene of unknown biological function with a cluster containing genes with known (or partially known) function can give an indication of the role of the unknown gene. Also, coexpressed genes are more likely to be coregulated (see Section 3.7).

Cluster analysis in a collection of gene expression profiles aims at identifying subgroups (i.e., clusters) of such coexpressed genes which thus have a higher probability of participating in the same pathway. Note that cluster analysis of expression data is only a first rudimentary step preceding further analysis, which includes motif finding [13–15], functional annotation, genetic network inference, and class discovery in the microarray experiments or samples themselves [4, 16]. Moreover, clustering often is an interactive process where the biologist or medical doctor has to validate or further refine the results and combine the clusters with prior biological or medical knowledge. Full automation of the clustering process is here still far away. The first generation of cluster algorithms (e.g., direct visual inspection [17], *K*-means [3], self-organizing maps (SOMs) [18], hierarchical clustering [2]) applied to gene expression profiles were mostly developed outside biological research. Although it is possible to obtain biologically meaningful results with these algorithms, some of their characteristics often complicate their use for clustering expression data [19]. They require, for example, the predefinition of one or more user-defined parameters that are hard to estimate by a biologist (e.g., the predefinition of the number of clusters in *K*-means and SOM—this number is almost impossible to predict in advance). Moreover, changing these parameter settings will often have a strong impact on the final result. These methods therefore need extensive parameter fine tuning, which means that a comparison of the results with different parameter settings is almost always necessary—with the additional difficulty that comparing the quality of the different clustering results is hard. Another problem is that first-generation clustering algorithms often force every data point into a cluster. In general, a considerable number of genes included in the microarray experiment do not contribute to the biological process studied, and these genes will therefore lack coexpression with other genes (they will have seemingly constant or even random expression profiles). Including these genes in one of the clusters will contaminate their content (these genes represent noise)

and make these clusters less suitable for further analysis. Finally, the computational and memory complexity of some of these algorithms often limit the number of expression profiles that can be analyzed at once. Considering the nature of our data sets (number of expression profiles often running up into the tens thousands), this constraint is often unacceptable. Recently, many new clustering algorithms have started to tackle some of the limitations of earlier methods (e.g., self-organizing tree algorithm, or SOTA [20], quality-based clustering [21], adaptive quality-based clustering [5], model-based clustering [22–26], simulated annealing [27], gene shaving [4], CAST [28], Fuzzy *C*-means [29]). Also, some procedures were developed that could help biologists to estimate some of the parameters needed for the first generation of algorithms (such as the number of clusters present in the data [23, 26, 27, 30, 31]). We will discuss a selection of these clustering algorithms in the following sections.

3.5.2. First-Generation Algorithms

Notwithstanding some of the disadvantages of these early methods, it must be noted that many good implementations (see Table 3.1) of these algorithms exist ready for use by biologists (which is not always the case with the newer methods).

3.5.2.1. Hierarchical Clustering Hierarchical clustering [1, 2, 19] is the most widely used method for clustering gene expression data and can be seen as the de facto standard. Hierarchical clustering has the advantage that the results can be nicely visualized (see Fig. 3.5). Two approaches are possible: a top-down approach (divisive clustering, see [32] for an example) and a bottom-up approach (agglomerative clustering, see [2]). The latter is the most commonly used and will be discussed here. In the agglomerative approach, each gene expression profile is initially assigned to a single cluster. The distance between every couple of clusters is calculated according to a certain distance measure (this results in a pairwise distance matrix). Iteratively (and starting from all singletons as clusters), the two closest clusters are merged and the distance matrix is updated to take this cluster merging into account. This process gives rise to a tree structure where the height of the branches is proportional to the pairwise distance between the clusters. Merging stops if only one

TABLE 3.1 Availability of Clustering Algorithms

Package	URL
Cluster	http://rana.lbl.gov/EisenSoftware.htm
Expression profiler	http://ep.ebi.ac.uk/
SOTA	http://bioinfo.cnio.es/sotarray
MCLUST	http://www.stat.washington.edu/fraley/mclust
AQBC	http://homes.esat.kuleuven.be/~dna/BioI/Software.html
GIMM	http://homepages.uc.edu/~medvedm/GIMM.htm

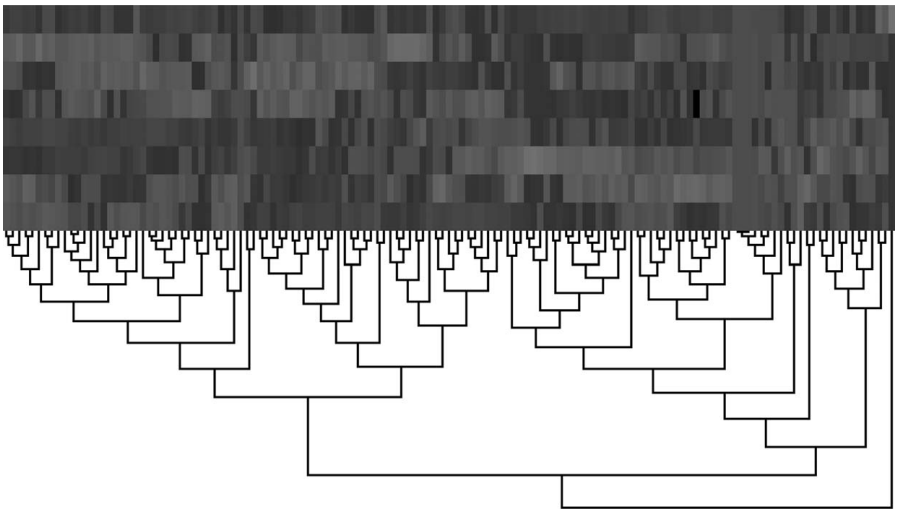


FIGURE 3.5. Result from hierarchical clustering 137 expression profiles of dimension 8 (plant wounding data of Reymond et al. [33]). The terminal branches of the tree are linked with the individual genes and the height of all the branches is proportional to the pairwise distance between the clusters. The so-called heat map corresponds to the expression matrix where, in this example, each column represents an expression profile, each row a microarray experiment, and the individual values are represented on a color (green to red) or grey scale.

cluster is left. Finally, clusters are formed by cutting the tree at a certain level or height. Note that this level corresponds to a certain pairwise distance that in its turn is rather arbitrary (it is difficult to predict which level will give the best biological results). Finally, note that the memory complexity of hierarchical clustering is quadratic in the number of gene expression profiles which can be a problem when considering the current size of the data sets.

3.5.2.2. K-Means Clustering *K*-means clustering [3, 34] results in a partitioning of the data (every gene expression profile belongs to exactly one cluster) using a predefined number K of partitions or clusters. *K*-means starts by dividing up all the gene expression profiles among N initial clusters. Iteratively, the center (which is nothing more than the average expression vector) of each cluster is calculated, followed by a reassignment of the gene expression vectors to the cluster with the closest cluster center. Convergence is reached when the cluster centers remain stationary.

Note that the predefinition of the number of clusters by the user also is rather arbitrary (it is difficult to predict the number of clusters in advance). In practice, this makes it necessary to use a trial-and-error approach where a comparison and biological validation of several runs of the algorithm with different parameter settings are necessary.

3.5.2.3. Self-Organizing Maps In SOMs [18, 35], the user has to predefine a topology or geometry of nodes (e.g., a two-dimensional grid, one node for each cluster), which again is not straightforward. These nodes are then mapped into the gene expression space, initially at random and iteratively adjusted. In each iteration, a gene expression profile is randomly picked and the node that maps closest to it is selected. This selected node (in gene expression space) is then moved into the direction of the selected expression profile. The other nodes are also moved into the direction of the selected expression profile but to an extent proportional to the distance from the selected node in the initial two-dimensional node topology.

3.5.3. Second-Generation Algorithms

In this section we describe several of the newer clustering methods that have specifically been designed to cluster gene expression profiles.

3.5.3.1. Self-Organizing Tree Algorithm The SOTA [20] combines both SOMs and divisive hierarchical clustering. The topology or node geometry here takes the form of a dynamic binary tree. Similar to SOMs, the gene expression profiles are sequentially and iteratively presented to the terminal nodes (located at the base of the tree—these nodes are also called cells). Subsequently, the gene expression profiles are associated with the cell that maps closest to it and the mapping of this cell plus its neighboring nodes are updated (moved into the direction of the expression profile). The presentation of the gene expression profiles to the cells continues until convergence. After convergence the cell containing the most variable population of expression profiles (variation is defined here by the maximal distance between two profiles that are associated with the same cell) is split into two sister cells (causing the binary tree to grow) and then the entire process is restarted. The algorithm stops (the tree stops growing) when a threshold of variability is reached for each cell, which involves the actual construction of a randomized data set and the calculation of the distances between all possible pairs of randomized expression profiles.

The approach described in [20] has some properties that make it potentially useful for clustering gene expression profiles. The clustering procedure itself is linear in the number of gene expression profiles (compare this with the quadratic complexity of standard hierarchical clustering). The number of clusters does not have to be known in advance. Moreover, the procedure provides for a statistical procedure to stop growing the tree. Therefore, the user is freed from choosing an (arbitrary) level where the tree has to be cut (as in standard hierarchical clustering). In our opinion, this method, however, also has some disadvantages. The procedure for finding the threshold of variability is time consuming. The entire process described in [20] is in fact quadratic in the number of gene expression profiles and there is no biological validation provided showing that this algorithm indeed produces biologically relevant results.

3.5.3.2. Model-Based Clustering Model-based clustering [23, 26, 36] is an approach that is not really new and has already been used in the past for other applications outside bioinformatics. In this sense it is not really a true second-generation algorithm. However, its potential use for cluster analysis of gene expression profiles has only been proposed recently and we thus we treat it in this text as a second-generation method. A Bayesian approach to model-based clustering Gaussian infinite mixture model (GIMM) was presented by Medvedovic [24, 25] (not discussed here).

Model-based clustering assumes that the data are generated by a finite mixture of underlying probability distributions, where each distribution represents one cluster. The covariance matrix for each cluster can be represented by its eigenvalue decomposition, which controls the orientation, shape, and volume of each cluster. Note that simpler forms for the covariance structure can be used (e.g., by having some of the parameters take the same values across clusters), thereby decreasing the number of parameters that have to be estimated but also decreasing the model flexibility (capacity to model more complex data structures).

In a first step, the parameters of the model are estimated with an EM algorithm using a fixed value for the number of clusters and a fixed covariance structure. This parameter estimation is then repeated for different numbers of clusters and different covariance structures. The result of the first step is thus a collection of different models fitted to the data and all having a specific number of clusters and a specific covariance structure. In a second step the best model in this group of models is selected (i.e., the most appropriate number of clusters and a covariance structure is chosen here). This model selection step involves the calculation of the Bayesian information criterion (BIC) [37] for each model, which is not further discussed here.

Yeung et al. [26] reported good results using their MCLUST software [36] on several synthetic data sets and real expression data sets. They claimed that the performance of MCLUST on real expression data was at least as good as could be achieved with a heuristic cluster algorithm (CAST [16], not discussed here).

3.5.3.3. Quality-Based Clustering In [21], a clustering algorithm (called QT_Clust) is described that produces clusters that have a quality guarantee that ensures that all members of a cluster should be coexpressed with all other members of this cluster. The quality guarantee itself is defined as a fixed and user-defined threshold for the maximal distance between two points within a cluster. Briefly said, QT_Clust is a greedy procedure that finds one cluster at a time satisfying the quality guarantee and containing a maximum number of expression profiles. The algorithm stops when the number of points in the largest remaining cluster falls below a prespecified threshold. Note that this stop criterion implies that the algorithm will terminate before all expression profiles are assigned to a cluster.

This approach was designed with cluster analysis of expression data in mind and has some properties that could make it useful for this task. First, by using a stringent quality control, it is possible to find clusters with tightly related expression profiles

(containing highly coexpressed genes). These clusters might therefore be good "seeds" for further analysis. Second, genes not really coexpressed with other members of the data set are not included in any of the clusters. There are, however, also some disadvantages. The quality guarantee of the clusters is a user-defined parameter that is hard to estimate and too arbitrary. This method is therefore, in practice, hard to use by biologists and extensive parameter fine tuning is necessary. This algorithm also produces clusters all having the same fixed diameter not optimally adapted to the local data structure. Furthermore, the computational complexity is quadratic in the number of expression profiles and no ready-to-use implementation is available.

3.5.3.4. Adaptive Quality-Based Clustering Adaptive quality-based clustering [5] was developed starting from the principles of quality-based clustering (finding clusters with a quality guarantee containing a maximal number of members) but was designed to circumvent some of its disadvantages.

Adaptive quality-based clustering is a heuristic iterative two-step approach. In the first step a quality-based approach is followed. Using an initial estimate of the quality of the cluster, a cluster center is located in an area where the density of gene expression profiles is locally maximal. Contrary to the original method [21], the computational complexity of this first step is only linear in the number of expression profiles. In the second step, called the adaptive step, the quality of the cluster—given the cluster center, found in the first step, that remains fixed—is reestimated so that the genes belonging to the cluster are, in a statistical sense, significantly coexpressed (higher coexpression that could be expected by chance, according to a significance level S). To this end, a bimodal and one-dimensional probability distribution (the distribution consists of two terms, one for the cluster and one for the rest of the data) is fitted to the data using an EM algorithm. Note that the computational complexity of this step is negligible with respect to the computational complexity of the first step. Finally, steps 1 and 2 are repeated using the reestimation of the quality as the initial estimate needed in the first step, until the relative difference between the initial and reestimated quality is sufficiently small. The cluster is subsequently removed from the data and the whole procedure is restarted. Note that only clusters whose size exceeds a predefined number are presented to the user.

The AQBC approach has some additional advantages over standard quality-based clustering that make it suited for the analysis of gene expression profiles. First, the user has to specify a significance level S. This parameter has a strict statistical meaning and is therefore much less arbitrary (contrary to the quality guarantee used in standard quality-based clustering). It can be chosen independently of a specific data set or cluster, and it allows for a meaningful default value (95%) that in general gives good results. This makes this approach user friendly without the need for extensive parameter fine tuning. Second, the algorithm produces clusters adapted to the local data structure (the clusters do not have the same radius). Third, the computational complexity of the algorithm is linear in the number of expression profiles. Finally, AQBC was extensively biologically validated.

However, the method also has some limitations. So is it a heuristic approach not proven to converge in every situation. Because of the model structure used in the second step, some additional constraints have to be imposed. They include the fact that only standardized expression profiles are allowed and that the method has to be used in combination with the Euclidean distance and cannot directly be extended to other distance measures.

3.5.4. Availability of Clustering Algorithms

As a conclusion to this overview of clustering algorithms, Table 3.1 gives an overview of some clustering methods for which the software is available for download or can be accessed online.

3.6. CLUSTER VALIDATION

Validation is another key issue when clustering gene expression profiles. The biologist using the algorithm is of course mainly interested in the biological relevance of these clusters and wants to use the results to discover new biological knowledge. This means that we need methods to (biologically and statistically) validate and objectively compare the results produced by new and existing clustering algorithms. Some methods for cluster validation have recently emerged (figure of merit [26], (adjusted) Rand index [38], silhouette coefficient [11], and looking for enrichment of functional categories [34]) and will be discussed below.

Ultimately, the relevance of a cluster result should be assessed by a biological validation. Of course it is hard, not to say impossible, to select the best cluster output since "the biologically best" solution will only be known if the biological system studied is completely characterized. Although some biological systems have been described extensively, none such completely characterized benchmark system is now available. A common method to biologically validate cluster outputs is to search for enrichment of functional categories within a cluster. Detection of regulatory motifs (see [34]) is also an appropriate biological validation of the cluster results. Some of the recent methodologies described in the literature to validate cluster results will be highlighted in the following.

Note that no real benchmark data set exists to unambiguously validate novel algorithms (however, the measurements produced by Cho et al. [17] on the cell cycle of yeast are often used for this purpose).

3.6.1. Testing Cluster Coherence

Based on biological intuition, a cluster result can be considered reliable if the within-cluster distance is small (i.e., all genes retained are tightly coexpressed) and the cluster has an average profile well delineated from the remainder of the data set (maximal intercluster distance). Such criteria can be formalized in several ways,

such as the sum-of-squares criterion of K-means [3], silhouette coefficients [11], or Dunn's validity index [39]. These can be used as stand-alone statistics to mutually compare cluster results. They can also be used as an inherent part of cluster algorithms if their value is optimized during the clustering process.

Here we briefly discuss the statistical method of silhouette coefficients [11]. Suppose g_i is an expression profile that belongs to cluster C_k. Call $v(g_i)$ (also called the within dissimilarity) the average distance of g_i to all other expression profiles from C_k. Suppose C_l is a cluster different from C_k and call $d(g_i, C_l)$ the average distance from g_i to all expression profiles of C_l. Now define $w(g_i)$ (also called the between dissimilarity) as

$$w(g_i) = \min_{C_l \neq C_k} d(g_i, C_l)$$

The silhouette $s(g_i)$ of g_i is now defined as

$$s(g_i) = \frac{w(g_i) - v(g_i)}{\max(w(g_i), v(g_i))}$$

Note that $-1 \leq s(g_i) \leq 1$. Consider two extreme situations now. First, suppose that the within dissimilarity $v(g_i)$ is significantly smaller than the between dissimilarity $w(g_i)$. This is the ideal case and $s(g_i)$ will be approximately 1. This occurs when g_i is *well clustered* and there is little doubt that g_i is assigned to an appropriate cluster. Second, suppose that $v(g_i)$ is significantly larger than $w(g_i)$. Now $s(g_i)$ will be approximately and g_i has in fact been assigned to the wrong cluster (worst-case scenario). We can now define two other measures: the average silhouette width of a cluster and the average silhouette width of the entire data set. The first is defined as the average of $s(g_i)$ for all expression profiles of a cluster and the second is defined as the average of $s(g_i)$ for all expression profiles in the data set. This last value can be used to mutually compare different cluster results and can be used as an inherent part of clustering algorithms if its value is optimized during the clustering process.

3.6.2. Rand Index: Validation Against an External Criterion

The Rand index [38, 40] is a measure that reflects the level of agreement of a cluster result with an external criterion, that is, an existing partition or a known cluster structure of the data. This external criterion could, for example, be an existing functional categorization, a predefined cluster structure if one is clustering synthetic data where the clusters are known in advance, or another cluster result obtained using other parameter settings for a specific clustering algorithm or obtained using other clustering algorithms. The latter could be used to investigate how sensitive a cluster result is to the choice of the algorithm or parameter setting. If this result proves to be relatively stable, one can assume that pronounced signals

are present in the data possibly reflecting biological processes. Suppose we want to compare two partitions (the cluster result at hand and the external criterion) of a set of N genes. Suppose that A is the number of gene pairs that are placed in the same subset (or cluster) in both partitions. Suppose that D is the number of gene pairs that are placed in different subsets in both partitions. The Rand index, which lies between 0 and 1, is then defined as the fraction of agreement between both partitions:

$$\frac{A+D}{\binom{N}{2}}$$

3.6.3. Figure of Merit

The figure of merit (FOM) [40] is a simple quantitative data-driven methodology that allows comparisons between outputs of different clustering algorithms. The methodology is related to the jackknife and leave-one-out cross validation. The method goes as follows. The clustering algorithm (for the genes) is applied to all experimental conditions (the data variables) except for one left-out condition. If the algorithm performs well, we expect that if we look at the genes from a given cluster, their values for the left-out condition will be highly coherent. Therefore, we compute the FOM for a clustering result by summing, for the left-out condition, the squares of the deviations of each gene relative to the mean of the genes in its cluster *for this condition*. The FOM measures the within-cluster similarity of the expression values of the removed experiment and therefore reflects the predictive power of the clustering. It is expected that removing one experiment from the data should not interfere with the cluster output if the output is robust. For cluster validation, each condition is subsequently used as a validation condition and the aggregate FOM over all conditions is used to compare cluster algorithms.

3.6.4. Sensitivity Analysis

Gene expression levels are the superposition of real biological signals and experimental errors. A way to assign confidence to a cluster membership of a gene consists in creating new in silico replicas of the microarray data by adding to the original data a small amount of artificial noise (similar to the experimental noise in the data) and clustering the data of those replicas. If the biological signal is stronger than the experimental noise in the measurements of a particular gene, adding small artificial variations (in the range of the experimental noise) to the expression profile of this gene will not drastically influence its overall profile and therefore will not affect its cluster membership. In this case, the cluster membership of that particular gene is robust with respect to sensitivity analysis, and a reliable confidence can be assigned to the clustering result of that gene. However, for genes with low signal-to-noise ratios, the outcome of the clustering result will be more sensitive to adding artificial noise. Through some robustness statistic [41], sensitivity analysis

let us detect which clusters are robust within the range of experimental noise and therefore trustworthy for further analysis.

The main issue in this method is to choose the noise level for sensitivity analysis. Bittner et al. [41] perturb the data by adding random Gaussian noise with zero mean and a standard deviation that is estimated as the median standard deviation for the log ratios for all genes across the experiments. This implicitly assumes that ratios are unbiased estimators of relative expression, yet reality often shows otherwise.

The bootstrap analysis methods described by Kerr and Churchill [42] to identify statistically significant expressed genes or to assess the reliability of a clustering result offers a more statistically founded basis for sensitivity analysis and overcomes some of the problems of the method described by Bittner et al. [41]. Bootstrap analysis uses the residual values of a linear analysis-of-variance (ANOVA) model as an estimate of the measurement error. By using an ANOVA model, nonconsistent measurement errors can be separated from variations caused by alterations in relative expression or by consistent variations in the data set. These errors are assumed to be independent with mean 0 and constant variance σ^2, but no explicit assumption on their distribution is made. The residuals are subsequently used to generate new replicates of the data set by bootstrapping (adding residual noise to estimated values).

3.6.5. Use of Different Algorithms

Just as clustering results are sensitive to adding noise, they are sensitive to the choice of clustering algorithm and to the specific parameter settings of a particular algorithm. Many clustering algorithms are available, each of them with different underlying statistics and inherent assumptions about the data. The best way to infer biological knowledge from a clustering experiment is to use different algorithms with different parameter settings. Clusters detected by most algorithms will reflect the pronounced signals in the data set. Again statistics similar to that of Bittner et al. [41] are used to perform these comparisons.

Biologists tend to prefer algorithms with a deterministic output since this gives the illusion that what they find is "right". However, nondeterministic algorithms offer an advantage for cluster validation since their use implicitly includes a form of sensitivity analysis.

3.6.6. Enrichment of Functional Categories

One way to *biologically* validate results from clustering algorithms is to compare the gene clusters with existing functional classification schemes. In such schemes, genes are allocated to one or more functional categories [23, 34] representing their biochemical properties, biological roles, and so on. Finding clusters that have been significantly enriched for genes with similar function is proof that a specific clustering technique produces biologically relevant results.

As stated in the overview section, the results of the expression profiling experiment of Cho et al. [17] studying the cell development cycle of yeast in a synchronized culture are often used as a benchmark data set. It contains 6220 expression profiles taken over 17 time points (measurements over 10-min intervals, covering nearly two cell cycles, also see http://cellcycle-www.stanford.edu). One of the reasons that these data are so frequently used as benchmark data for the validation of new clustering algorithms is because of the striking cyclic expression patterns and because the majority of the genes included in the data have been functionally classified [43] (MIPS database, see http://mips.gsf.de/proj/yeast/catalogues/funcat/index.html), making it possible to biologically validate the results.

Assume that a certain clustering method finds a set of clusters in the Cho et al. data. We could objectively look for functionally enriched clusters as follows: Suppose that one of the clusters has n genes where k genes belong to a certain functional category in the MIPS database and suppose that this functional category in its turn contains f genes in total. Also suppose that the total data set contains g genes (in the case of Cho et al. [17], g would be 6220). Using the cumulative hypergeometric probability distribution, we could measure the degree of enrichment by calculating the probability, or P-value, of finding by chance at least k genes in this specific cluster of n genes from this specific functional category that contains f genes out of the whole g annotated genes:

$$P = 1 - \sum_{i=0}^{k-1} \frac{\binom{f}{i}\binom{g-f}{n-i}}{\binom{g}{n}} = \sum_{i=k}^{\min(n,f)} \frac{\binom{f}{i}\binom{g-f}{n-i}}{\binom{g}{n}}$$

These P-values can be calculated for each functional category in each cluster. Since there are about 200 functional categories in the MIPS database, only clusters where the P-value is smaller than 0.0003 for a certain functional category are said to be significantly enriched (level of significance 0.05). Note that these P-values can also be used to compare the results from functionally matching clusters identified by two different clustering algorithms on the same data.

As an example of cluster validation and as an illustration of our AQBC, we compare K-means and AQBC on the Cho et al. data. We performed AQBC [5] using the default setting for the significance level (95%) and compared these results with those for K-means reported by Tavazoie et al. [34]. As discussed above, the genes in each cluster have been mapped to the functional categories in the MIPS database and the negative base-10 logarithm of the hypergeometric P-values (representing the degree of enrichment) have been calculated for each functional category in each cluster. In Table 3.2, we compare enrichment in functional categories for the three most significant clusters found by each algorithm. To compare K-means and AQBC, we identified functionally matching clusters

TABLE 3.2 Comparison of Functional Enrichment for Yeast Cell Cycle Data of Cho et al. Using AQBC and *K*-Means

Cluster Number		Number of Genes		MIPS Functional Category	Number of Genes in Category		*P*-Value	
AC	KM	AC	KM		AC	KM	AC	KM
1	1	302	164	Ribosomal proteins	101	64	80	54
				Organization of cytoplasm	146	79	77	39
				Protein synthesis	119	NR	74	NR
				Cellular organization	211	NR	34	NR
				Translation	17	NR	9	NR
				Organization of chromosome structure	4	7	1	4
2	4	315	170	Mitochondrial organization	62	32	18	10
				Energy	35	NR	8	NR
				Proteolysis	25	NR	7	NR
				Respiration	16	10	6	5
				Ribosomal proteins	24	NR	4	NR
				Protein synthesis	33	NR	4	NR
				Protein destination	49	NR	4	NR
5	2	98	186	DNA synthesis and replication	20	23	18	16
				Cell growth and division, DNA Synthesis	48	NR	17	NR
				Recombination and DNA repair	12	11	8	5
				Nuclear organization	32	40	8	4
				Cell cycle control and mitosis	20	30	7	8

Note: NR = not reported.

manually. The first column (AC) gives the index of the cluster identified by AQBC. The second column (KM) gives the index of the matching cluster for *K*-means as described in Tavazoie et al. [34]. The third column (AC) gives the number of genes of in the cluster for AQBC. The fourth column (KM) gives the number of genes of in the cluster for *K*-means. The fifth column (MIPS functional category) lists the significant functional categories for the two functionally matching clusters. The sixth column (AC) gives the number of genes of the corresponding functional category in the cluster for AQBC. The seventh column (KM) gives the number of genes of the corresponding functional category in the cluster for *K*-means. The eighth column (AC) gives the negative logarithm in base 10 of the hypergeometric *P*-value for AQBC. The ninth column (KM) gives the negative logarithm in base 10 of the hypergeometric *P*-value for *K*-means. Although we do not claim to draw any conclusion from this single table, we observe that the enrichment in functional categories is stronger for AQBC than for *K*-means. This result and several others are discussed extensively in [5].

3.7. SEARCHING FOR COMMON BINDING SITES OF COREGULATED GENES

In the previous sections, we described the basic ideas underlying several clustering techniques together with their advantages and shortcomings. We also discussed the preprocessing steps necessary to make microarray data suitable for clustering. Finally, we described methodologies for validating the result of a clustering algorithm. We can now make the transition toward looking at the groups of genes generated by clustering and study the sequences of these genes to detect motifs that control their expression (and cause them to cluster together in the first place).

Given a cluster of genes with highly similar expression profiles, the next step in the analysis is the search for the mechanism that is responsible for their coordinated behavior. We basically assume that coexpression frequently arises from transcriptional coregulation. As coregulated genes are known to share some similarities in their regulatory mechanism, possibly at transcriptional level, their promoter regions might contain some common motifs that are binding sites for transcription regulators. A sensible approach to detect these regulatory elements is to search for statistically overrepresented motifs in the promoter region of such a set of coexpressed genes [14, 34, 44–46]. In this section we describe the two major classes of methods to search for overrepresented motifs. The first class of methods is comprised of string-based methods that mostly rely on counting and comparing oligonucleotide frequencies. Methods in the second class are based on probabilistic sequence models. For these methods, the model parameters are estimated using maximum likelihood or Bayesian inference. We start with a discussion of the important facts that we can learn by looking at a realistic biological example. Prior knowledge about the biology of the problem at hand will facilitate the definition of a good model. Next, we discuss the different string-based methods, starting from a simple statistical model and gradually refining the models and the statistics to handle more complex configurations. Then we switch to the probabilistic methods and introduce EM for motif finding. Next, we discuss Gibbs sampling for motif finding. This method has been proven to be very effective for motif finding in DNA sequences. We therefore explain the basic ideas underlying this method and overview the extensions, including our own work, that are necessary for the practical use of this method.

A recent assessment of motif-finding tools organized by Tompa [47] showed that there is still a lot of work to do in the field of motif finding. The setup of the assessment was that different blind sequence sets in different organisms were provided by the organizers and those sets were analyzed by the participating teams. Each algorithm was run by its own developer to make sure that the tools were used as intended. Most tools had a similar (rather low) performance and only Weeder [48] was doing a better job than the rest. MotifSampler, our own implementation, turned out to be the only algorithm that performed better on real sequence data compared to the performance on artficial data.

3.7.1. Realistic Sequence Models

To search for common motifs in sets of upstream sequences, a realistic model should be proposed. Simple motif models are designed to search for conserved motifs of fixed length, while more complex models will incorporate variability like insertions and deletions into the motif model. Not only is the model of the binding site itself important but also the model of the background sequence in which the motif is hidden and the number of times a motif occurs in the sequence play important roles.

To illustrate this complexity, we look at an example in *Saccharomyces cerevisiae*. Figure 3.6 gives a schematic representation of the upstream sequences from 11 genes in *S. cerevisiae* that are regulated by the Cbfl–Met4p–Met28p complex and Met31p or Met32p in response to methionine (selected from [49]). The consensus, which is the dominant DNA pattern describing the motif, for these binding sites is given by TCACGTG for the Cbfl–Met4p–Met28p complex and AAAACTGTGG for Met31p or Met32p [49]. A logo representation [50] of the aligned instances of the two binding sites is shown in Figure 3.7. Such a logo represents the frequency of each nucleotide at each position, the relative size of the symbol represents the relative frequency of each base at this position, and the total height of the symbols represents the magnitude of the deviation from a uniform (noninformative) distribution. Figure 3.6 shows the locations of the two binding sites in the region 800 bp upstream of translation start. It is clear from this picture that there are several possible configurations of the two binding sites present in this data set. First it is important to note that motifs can occur on both strands. Transcription factors indeed bind directly on the double-stranded DNA and therefore motif detection software should take this fact into account. Second, sequences could have either zero, one, or multiple copies of a motif. This example gives an indication of the kind of data that come with a realistic biological data set. *Palindromic* motifs are, from a computational point of view, a special type of transcription factor binding site as it is a subsequence that is exactly the same as

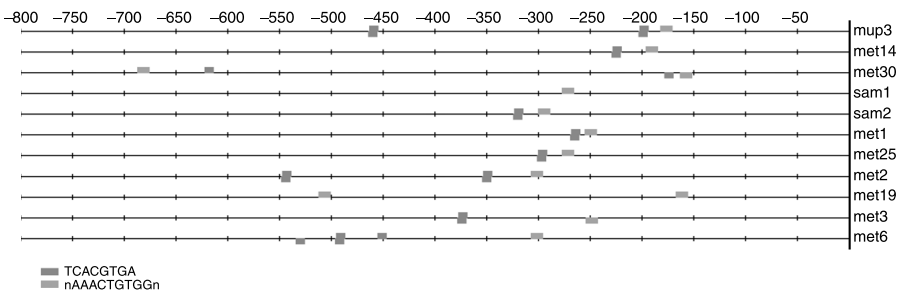


FIGURE 3.6. Schematic representation of upstream region of set of coregulated genes. Several possible combinations of the two motifs are present: (1) motifs occur on both strands, (2) some sequences contain one or more copies of the two binding sites, or (3) some sequences do not contain a copy of a motif.

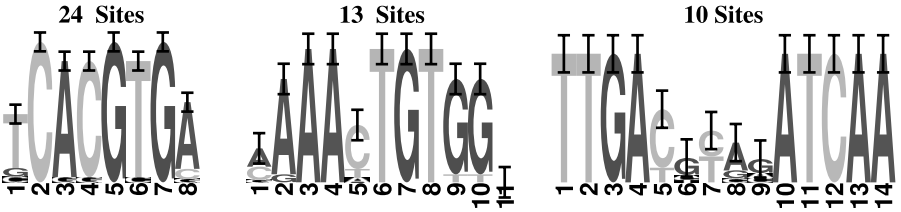


FIGURE 3.7. Logo representation of three sets of known TFBSs in *S.cerevisae* and *Salmonela typhimurium*: (*a*) binding site of Cbfl–Met4p–Met28p; (*b*) binding site of Met31p or Met32p; (*c*) FNR binding site.

its own reverse complement. An example of a palindromic motif is found in the DNA binding protein that regulates genes involved in cellular respiration (FNR) motif of Figure 3.7*c*. The left part of consensus TTGA is the reverse complement of the right part, TCAA.

A second class of special motifs is comprised of *gapped* motifs or spaced dyads. Such a motif consists of two smaller conserved sites separated by a gap or spacer. The spacer occurs in the middle of the motif because the transcription factor binds as a dimer. This means that the transcription factor is made out of two subunits that have two separate contact points with the DNA sequence. The parts where the transcription factor binds to the DNA are conserved but are typically rather small (3 to 5 bp). These two contact points are separated by a nonconserved gap or spacer. This gap is mostly of fixed length but might be slightly variable. Figure 3.7*c* shows a logo representation of the FNR binding site in bacteria.

Currently another important research topic is the search for cooperatively binding factors [51]. When only one of the transcription factors binds, there is no or a low level of activation, but the presence of two or more transcription factors activates the transcription of a certain gene. If we translate this into the motif-finding problem, we could search for individual motifs and try to find, among the list of possible candidates, motifs that tend to occur together. Another possibility is to search for multiple motifs at the same time.

3.7.2. Oligonucleotide Frequency Analysis

The most intuitive approach to extract a consensus pattern for a binding site is a string-based approach, where typically overrepresentation is measured by exhaustive enumeration of all oligonucleotides. The observed number of occurrences of a given motif is compared to the expected number of occurrences. The expected number of occurrences and the statistical significance of a motif can be estimated in many ways. In this section we give an overview of the different methods.

3.7.2.1. Basic Enumerations Approach A basic version of the enumeration methods was implemented by van Helden et al. [49]. They presented a simple and fast method for the identification of DNA binding sites in the upstream regions from

families of coregulated genes in *S. cerevisiae*. This method searches for short motifs of 5 to 6 bp long. First, for each oligonucleotide of a given length, we compute the expected frequency of the motif from all the noncoding, upstream regions in the genome of interest. Based on this frequency table, we compute the expected number of occurrences of a given oligonucleotide in a specific set of sequences. Next, the expected number of occurrences is compared to the actual, counted, number of occurrences in the data set. Finally, we compute a significance coefficient that takes into account the distinct number of oligonucleotides. A binomial statistic is appropriate in the case where there are nonoverlapping segments.

Van Helden et al. [15] have extended their method to find spaced dyads; these are motifs consisting of two small conserved boxes separated by a fixed spacer. The spacer can be different for distinct motifs and therefore the spacer is systematically varied between 0 and 16. The significance of this type of motif can be computed based on the combined score of the two conserved parts in the input data or based on the estimated complete dyad frequency from a background data set.

The greatest shortcoming of this method is that there are no variations allowed within an oligonucleotide. Tompa [52] addressed this problem when he proposed an exact method to find short motifs in DNA sequences. Tompa used a measure that differs from the one used by van Helden et al. to calculate the statistical significance of motif occurrences. First, for each k-mer s with an allowed number substitutions, the number of sequences in which s is found is calculated. Next, the probability p_s of finding at least one occurrence of s in a sequence drawn from a random distribution is estimated. Finally, the associated z-score is computed as

$$z_s = \frac{N_s - Np_s}{\sqrt{Np_s(1 - p_s)}}$$

The score z_s gives a measure of how unlikely it is to have N_s occurrences of s given the background distribution. Tompa proposed an efficient algorithm to estimate p_s from a set of background sequences based on a Markov chain.

3.7.2.2. Combinatorial Approaches Another important contribution in this field was made by Pevzner and Sze [53], who defined the motif finding in terms of a computationally challenging problem. The assumption from which they start is that motifs that can be considered as implanted are similar up to a given number of mutations c to a certain consensus sequence of length l. Keich and Pevzner [54] elaborated on the concept of (l, c)-motifs and defined a twilight zone where all motif finders would have a hard time finding the correct answers. In [53] a combinatorial approach is presented as WINNOWER and SP-STAR to solve this problem. WINNOWER is an iterative graph-based approach and uses substantial computational power. SP-STAR is an extension of this procedure by adding a heuristic to separate random signals from true signals. In [55] this method was further refined by introducing MULTIPROFILER, which incorporates two extensions: the utilization of the neighborhood of a word and the use of multipositional

profiles. With MULTIPROFILER they managed to further push the performance of their motif-finding algorithms. Another combinatorial method was presented by Buhler and Tompa [56], who used random projections to define a set of instances that can be used to initialize EM for motif finding.

3.7.2.3. Suffix Trees Another interesting string-based approach is based on the representation of a set of sequences with a suffix tree [57, 58]. Vanet et al. [58] have used suffix trees to search for single motifs in whole bacterial genomes. Marsan and Sagot [57] later extended the method to search for combinations of motifs. The proposed configuration of a structured motif is a set of p motifs separated by a spacer that might be variable. The variability is limited to ± 2 bp around an average gap length. They also allow for variability within the binding site. The representation of upstream sequences as suffix trees resulted in an efficient implementation despite the large number of possible combinations.

3.7.3. Probabilistic Methods

While in the previous section a binding site was modeled as a set of strings, the following methods are all based on a representation of the motif model by a position weight matrix.

3.7.3.1. Probabilistic Model of Sequence Motifs In the simplest model, we have a set of DNA sequences where each sequence contains a single copy of the motif of fixed length. (For the sake of simplicity, we will consider here only models of DNA sequences, but the whole presentation applies directly to sequences of amino acids.) Except for the motif, a sequence is described as a sequence of independent nucleotides generated according to a single discrete distribution $\theta_0 = (q_0^A, q_0^C, q_0^G, q_0^T)^T$ called the *background model*. The motif θ_W itself is described by what we call a *position frequency matrix*, which are W independent positions generated according to different discrete distributions q_i^b:

$$\theta_W = \begin{pmatrix} q_1^A & q_2^A & \cdots & q_W^A \\ q_1^C & q_2^C & \cdots & q_W^C \\ q_1^G & q_2^G & \cdots & q_W^G \\ q_1^T & q_2^T & \cdots & q_W^T \end{pmatrix}$$

If we know the location a_i of the motif in a sequence S_i, the probability of this sequence given the motif position, the motif matrix, and the background model is

$$P(S_i|a_i, \theta_W, \theta_0) = \prod_{j=1}^{a_i-1} q_0^{S_{ij}} \prod_{j=a_i}^{a_i+W-1} q_{j-a_i+1}^{S_{ij}} \prod_{j=a_i+W}^{L} q_0^{S_{ij}}$$

Wherever appropriate, we will pool the motif matrix and the background model into a single set of parameters $\theta = (\theta_0, \theta_W)$. For a set of sequences, the probability of

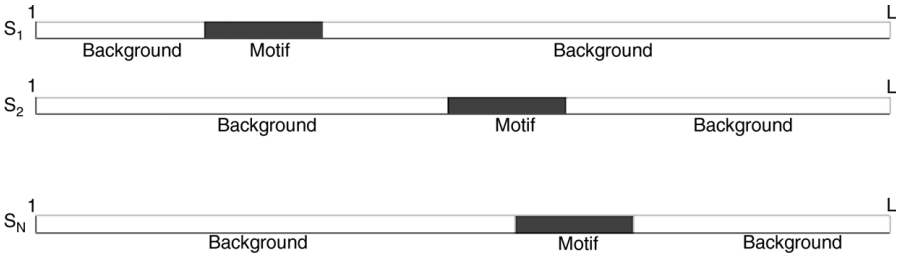


FIGURE 3.8. In this basic sequence model, each sequence contains one and only one copy of the motif. The first part of the sequence is generated according to the background model θ_0, then the motif is generated by the motif matrix θ_W, after which the rest of the sequence is again generated according to the background model.

the whole set $S = \{S_1, \ldots, S_N\}$ given the *alignment* (i.e., the set of motif positions), the motif matrix, and the background model is

$$P(S \mid A, \theta) = \prod_{i=1}^{N} P(S_i \mid a_i, \theta) \tag{3.1}$$

The sequence model is illustrated in Figure 3.8. The idea of the EM algorithm for motif finding is to find simultaneously the motif matrix, the alignment position, and the background model that maximize the likelihood of the weights and alignments. Gibbs sampling for motif finding extends EM in a stochastic fashion by not looking for the maximum-likelihood configuration but generating candidate motif matrices and alignments according to their posterior probability given the sequences.

3.7.3.2. Expectation–Maximization One of the first implementation to find a matrix representation of a binding site was a greedy algorithm by Hertz et al. [59] to find the site with the highest information content (which is the entropy of the discrete probability distribution represented by the motif matrix). This algorithm was capable of identifying a common motif that is present once in every sequence. This algorithm has been substantially improved over the years [60]. In their latest implementation, CONSENSUS, Hertz and Stormo [60] have provided a framework to estimate the statistical significance of a given information content score based on large-deviation statistics.

Within the maximum-likelihood estimation framework, EM is the first choice of optimization algorithm. Expectation–maximization is a two-step iterative procedure for obtaining the maximum-likelihood parameter estimates for a model of observed data and missing values. In the expectation step, the expectation of the data and missing values is computed given the current set of model parameters. In the maximization step, the parameters that maximize the likelihood are computed. The algorithm is started with a set of initial parameters and iterates over the two described steps until the parameters have converged. Since EM is a

gradient-ascent method, EM strongly depends on the initial conditions. Poor initial parameters may lead EM to converge to a local minimum.

For motif finding EM was introduced by Lawrence and Reilly [61] and was an extension of the greedy algorithm of Hertz et al. [59]. It was primarily intended for searching motifs in related proteins, but the method described could also be applied to DNA sequences. The basic model assumption is that each sequence contains exactly one copy of the motif, which might be reasonable in proteins but is too strict in DNA. The starting position of each motif instance is unknown and is considered as being a missing value from the data. If the motif positions are known, then the observed frequencies of the nucleotides at each position in the motif are the maximum-likelihood estimates of model parameters. To find the starting positions, each subsequence is scored with the current estimate of the motif model. These updated probabilities are used to reestimate the motif model. This procedure is repeated until convergence. Since assuming there is exactly one copy of the motif per sequence is not really biological sound, Bailey and Elkan proposed an advanced EM implementation for motif finding called MEME [62, 63]. Although MEME was also primarily intended to search for protein motifs, MEME can also be applied to DNA sequences.

To overcome the problem of initialization and getting stuck in local minima, MEME proposes to initialize the algorithm with a motif model based on a contiguous subsequence that gives the highest likelihood score. Therefore, each substring in the sequence set is used as a starting point for a one-step iteration of EM. Then the motif model with the highest likelihood is retained and used for further optimization steps until convergence. The corresponding motif positions are then masked and the procedure is repeated. Finally, Cardon and Stormo proposed an EM algorithm to search for gapped motifs [64]. However, while performing well for extended protein motifs, EM often suffers badly from local minima for short DNA motifs. An even more intelligent initialization procedure is the random-projections method of Buhler and Tompa [56].

3.7.3.3. Basic Algorithm for Gibbs Sampling for Motif Finding The probabilistic framework led to another important approach to solve the motif-finding problem. Gibbs sampling for motif finding was presented by Lawrence et al. [65] and was later described in more technical details by Liu et al. [66]. Gibbs sampling is a Markov chain Monte Carlo procedure that fits perfectly within the missing-data problem. While EM gives the maximum-likelihood estimates, the goal here is to model the posterior distribution and to generate data accordingly. Shortly said, the proposed algorithm is basically a collapsed Gibbs sampler which involves a Markov chain of the form:

$$\text{Sample } a_1^{(t+1)} \text{ from } \pi(a_1 \mid a_2^{(t)}, \ldots, a_N^{(t)}, S).$$

$$\text{Sample } a_2^{(t+1)} \text{ from } \pi(a_2 \mid a_1^{(t+1)}, a_3^{(t)}, \ldots, a_N^{(t)}, S).$$

$$\vdots$$

$$\text{Sample } a_N^{(t+1)} \text{ from } \pi(a_N \mid a_1^{(t+1)}, \ldots, a_{N-1}^{(t+1)}, S).$$

In words, the alignment position in sequence i is sampled according to a probability distribution dependent on the current set of alignment position in all other sequences. It can be shown that this Markov chain has the distribution $\pi(A \mid S)$ as its equilibrium state [66]. The computation of these probability distributions involves the use of multinomial probability distributions (for the probability of the data based on the likelihood function presented in Section 3.7.3.1 and on the motif matrix and the background model) and Dirichlet probability distributions (for the probability of the parameters of the motif matrix). The derivation of the collapsed Gibbs sampler involves several properties of integrals of Dirichlet distributions and a number of approximations are used to speed up the algorithm further. To be concrete, we present here the resulting algorithm:

1. Input: A set of sequences S and the length W of the motif to search.
2. Compute the background model θ_0 from the nucleotide frequencies observed in S.
3. Initialize the alignment vector $A = \{a_i \mid i = 1, \ldots, N\}$ uniformly at random.
4. For each sequence $S_z,\ z = 1, \ldots, N$:
   (a) Create subsets $\widetilde{S} = \{S_i \mid i \neq z\}$ and $\widetilde{A} = \{a_i \mid i \neq z\}$.
   (b) Compute θ_W from the segments indicated by $\widetilde{A}$.
   (c) Assign to each possible alignment start $(x_{zj},\ j = 1, \ldots, L_i - W + 1)$ in S_z a weight $W(x_{zj})$ given by the probability that the corresponding segment is generated by the motif versus the background:

$$W(x_{zj}) = \frac{P(S_{zj}, \ldots, S_{z(j+W-1)} \mid \theta_W)}{P(S_{zj}, \ldots, S_{z(j+W-1)} \mid \theta_0)} = \prod_{k=1}^{W} \frac{q_z^{S_z(j+k-1)}}{q_0^{S_z(j+k-1)}}$$

   (d) Draw new alignment positions a_z according to the normalized probability distribution $W(x_{zj}) / \sum_{k=1}^{L_i - W + 1} W(x_{zj})$.
5. Repeat step 4 until the Markov chain reaches stochastic convergence (fixed number of iterations).
6. Output: A motif matrix θ_W and an alignment A.

3.7.3.4. Extended Gibbs Sampling Methods Several groups proposed advanced methods to fine tune the Gibbs sampling algorithm for motif finding in DNA sequences. A first version of the Gibbs sampling algorithm that was especially tuned toward finding motif in DNA sequences is AlignACE [14], and this version was later refined [67]. This algorithm was the first reported to be used for the analysis of gene clusters. Several modifications were made in AlignACE with respect to the original Gibbs sampling algorithm. First, one motif at the time was retrieved and the positions were masked instead of simultaneous multiple motif searching.

Second, AlignACE was implemented with a fixed single-nucleotide background model based on base frequency (SNF) in the sequence set. Also, both strands were included in the search. Finally, in the latest version, the *maximum* a posteriori likelihood score was used to judge different motifs.

To have a more robust motif-finding tool, we designed MotifSampler [68, 69] as part of INCLUSive [13]. This implementation was specifically developed to search in sets of upstream sequences from groups of coexpressed genes. Since the results of clustering are known to be subject to noise, only a subset of the set of coexpressed genes will actually be coregulated. This implies that only part of the sequences have one or more copies of the binding site while others have no binding site at all. Therefore it is important to have an algorithm that can cope with this form of noise. MotifSampler uses the framework of the probabilistic sequence model to estimate the number of copies of a motif in a sequence. For each sequence in the data set the number of copies of the motif is estimated, which is more accurate than earlier methods. Furthermore, while the original Gibbs sampler, and also AlignACE and MEME, uses only the single-nucleotide frequency to model the background, we used a higher order Markov model to represent the background sequence in which the binding sites are hidden. In [68] and [69] we demonstrated that the use of a higher order background model built from an independent data set significantly improves the performance of the Gibbs sampling algorithm in *Arabidopsis thaliana*. Later, we also proved that the species-specific background models have a profound impact on the motif detection in prokaryotes [70]. To exemplify the improvements obtained by further refinements of the Gibbs sampling strategy, we report here briefly the use of higher order background models on a data set of coregulated genes from plants. The data set consists of 33 genes known to be regulated in part by the G-box transcription factor, which is linked to the light response of plants. Additionally, noisy sequences not suspected to contain an active motif are added gradually to this set. The results of this analysis are shown in Table 3.3). We can observe that the performance of the higher order algorithms is more robust to the addition of noisy sequence than that of the zero-order algorithm. The improved robustness of the method due to the higher order background model is discussed extensively in [68].

Currently, we provide our precompiled background models for all fully sequenced prokaryotes and most eukaryotes (see http://homes.esat.kuleuven.be/~dna/BioI/Software.html). To make MotifSampler as user friendly as possible,

TABLE 3.3 Number of Times Motif CACGTG Is Found in Increasing Noisy G-Box Data Sets

Noise[a]	0	10	20	30	40	50	60
SNF genes	85 (1)	76 (1)	67 (1)	69 (2)	37 (4)	33 (4)	6 (4)
Third order	92 (1)	84 (1)	87 (1)	81 (1)	64 (2)	67 (2)	41 (2)

[a]Number of added noisy sequence to the G-box data set.

Note: Numbers in parentheses are the number of times the G-box motif is found in 100 runs (rank of G-box motif).

we provided a Web interface where only a limited number of easy-to-understand parameters have to be specified.

BioProspector [71] also uses a Gibbs sampling strategy to search for common motifs in the regulatory region of coexpressed genes. In this implementation various extensions are proposed. First, BioProspector also uses zero- to third-order Markov background models. Second, the core sampling step was replaced by a *threshold sampler*. This threshold-sampling step was incorporated to estimate the number of copies of a motif in a sequence. The program defines two threshold T_L and T_H. Instances with a score higher than T_H will be automatically selected while there will be one motif sampled from those motifs that have a score between T_L and T_H. Threshold T_H is set proportional to the product of the average length of the input sequences and the motif width; T_L is initialized at 0 and linearly increases until it reaches the value of $T_H/8$. This threshold-sampling step ensures faster convergence. As another modification, BioProspector proposes two possible alternative motif models. The first possibility is to search for palindromic motifs. The second possibility is to search for a gapped version of the motif model, where the motif consists of two blocks separated by a gap of variable length. The gapped version searches for two motifs at the same time that occur within a given range.

Ann_spec [72] has a slightly different approach to model the motif. The motif model is represented with a sparsely encoded perceptron with one processing unit. The weights of the perceptron resemble the position weight matrix. This model is based on the approximation of the total protein-binding energy by the sum of partial binding energies at the individual nucleotides in the binding sites. The use of a perceptron is also justified by the fact that it can be used to approximate posterior probability distributions. A gradient-descent training method is used to find the parameters of the perceptron. For the training set for the perceptron, positive examples are selected using a Gibbs sampling procedure. Negative examples can be constructed either from random sequences or from genomic data. To improve the specificity of the motif model, a background model based on an independent data set is preferred.

3.7.4. Recent Advances in Motif Finding

This section about motif finding would not be complete without a short note about the recent advances in module search and phylogenetic footprinting.

***3.7.4.1. Searching for Modules of* cis*-Regulatory Elements** Another very important aspect of gene regulation is that the great diversity of cells in higher organisms does not come from the increasing number of genes but rather from the growing complexity of the system controlling the gene expression. When dealing with data from such a higher organism, it is no longer sufficient to only search for individual binding sites but combinations or modules of cis-regulatory elements come in to play. Most of these approaches start from a module of known TFBSs and screen sequences for the presence of this module.

As one of the only unsupervised methods, Ann_spec was extended to search for cooperatively acting transcription binding factors by GuhaThakurta and Stormo [73]. Co-Bind searches for two motifs simultaneously by combining the weights that optimize the objective functions of the two individual perceptrons. The identification of two motifs simultaneously improved significantly the detection of the true motifs compared to the classical methods of searching for one motif at a time. Most programs have been developed to find modules of known TFBSs. The basic methods developed methods to assess the statistical significance of the number of occurrences of a given module in a sliding window [74–78]. Some of the programs use a hidden Markov model (HMM) to model a module of TFBSs in a given region. The first implementation was done by Crowley et al. [79], and other flavors are Ahab [80], Cister [81], COMET [82] with statistical signifance, MCAST [83], and Stubb [84]. We have developed ModuleSearcher [85], which uses an A* algorithm to find a module that is specific to a set of coregulated genes.

3.7.4.2. Phylogenetic Footprinting Given the increasing availability of newly sequenced genomes, cross-species comparison becomes a more important aspect of bioinformatics research. Phylogenetic footprinting is the methodology for the discovery of TFBSs in a set of orthologous regulatory regions from multiple species. The basic idea is that selective pressure causes functional elements to evolve at a slower rate than nonfunctional sequences. This means that unusually well conserved sites among a set of orthologous regulatory regions might be functional regulatory elements. Applying motif search algorithms to such a set of orthologous promoter sequences should reveal these conserved patterns. This approach has been implemented by several groups already.

McCue et al. [86] have used a Gibbs motif-finding algorithm for phylogenetic footprinting. They also proposed a motif model that accounts for palindromic motifs. Their most important contribution lies in the use of a position-specific background model estimated with a Bayesian segmentation model [87]. This model accounts for the varying composition of the DNA upstream of a gene. Blanchette et al. [88–90] developed FootPrinter in which they combine the oligomer count with the phylogenetic tree to assess the statistical significance. In [91], first conserved blocks are found in sets of orthologuous genes. In the next step blocks are aligned to each other to find motifs common to a specific set of genes. ConREAL [92] first screens sequences for known TFBSs and then combines these hits with respect to the phylogeny of the data set. Cliften et al. [93] found functional features in six *Saccharomyces* genomes. ConSite [94] is a suite of methods for the identification and visualization of conserved TFBSs and reports putative TFBSs that are located in conserved regions and located as pairs of sites in alignments between two orthologous sequences.

We have recently published a study on the phylogenetic footprinting of PmrAB targets in *Salmonella thypimurium* [95]. In this study we applied our MotifSampler to a set of orthologous genes known to be regulated by PmrAB. We then screened the whole genome of *S. thypimurium* for new targets, some of which were validated in vivo.

TABLE 3.4 Selection of Available Motif-Finding Algorithms

Package	URL	Ref.
RSA tools	http://rsat.ulb.be/rsat/	96
YMF	http://abstract.cs.washington.edu/~blanchem/cgi-bin/YMF.pl	97
REDUCE	http://bussemaker.bio.columbia.edu/reduce/	98
Consensus	http://ural.wustl.edu/softwares.html	59
MEME	http://meme.sdsc.edu/meme/website/	62
Gibbs Sampler	http://bayesweb.wadsworth.org/gibbs/gibbs.html	65
AlignACE	http://atlas.med.harvard.edu/	14
BioProspector	http://bioprospector.stanford.edu/	71
MotifSampler	http://homes.esat.kuleuven.be/~dna/BioI/Software.html	68
MultiProfiler	http://www-cse.ucsd.edu/groups/bioinformatics/software.html	54
Ann_spec	http://www.cbs.dtu.dk/services/DNAarray/ann-spec.php	72
Weeder	http://www.pesolelab.it/Tool/ind.php	48

3.7.5. Availability of Motif-Finding Software

As a conclusion to this overview of motif-finding algorithms, Table 3.4 gives an overview of some popular motif-finding methods for which the software is available for download or can be accessed online.

3.8. INCLUSIVE: ONLINE INTEGRATED ANALYSIS OF MICROARRAY DATA

Analysis of microarray experiment is not restricted to a single cluster experiment. Inferring ''biological knowledge'' from a microarray analysis usually involves a complete analysis going from the sequential use of distinct data preparation steps to the use of different complex procedures that make predictions on the data. Clustering predicts whether genes behave similarly while motif finding aims at retrieving the underlying mechanism of this similar behavior. These data-mining procedures thus make predictions about the same biological system. These predictions are in the best case consistent with each other, but they can also contradict each other. Combining these methods into a global approach therefore increases their relevance for biological analysis. Moreover, this integration also allows the optimal matching of the different procedures (such as the quality requirements in AQBC that reduce the noise level for Gibbs sampling for motif finding). Furthermore, such global approaches require extensive integration at the information technology level. Indeed, as is often underestimated, the collection of data from multiple data sources and transformation of the output of one algorithm to the input of the next algorithm are often tedious tasks.

To make such an integrated analysis of microarray data possible, we have developed and made publicly available our INCLUSive Web tool (originally INCLUSive

stands for INtegrated CLustering, Upstream sequence retrieval, and motif Sampling; http://homes.esat.kuleuven.be/~dna/BioI) (see also the flowchart of Fig. 3.1). As an illustration of the results obtained by combined AQBC and MotifSampler, we show the results of motif finding on a microarray experiment in plants. The data are a microarray experiment on the response to mechanical wounding of the plant *A. thaliana*. The microarray consists of 150 genes related to stress response in plants. The experiment consists of expression measurements for those 150 genes at seven time points following wounding (after 30 min, 60 min, 90 min, 3 h, 6 h, 9 h, and 24 h). The expression data were clustered using AQBC with a significance level of 95%.

Four clusters were identified that contained at least five genes and those were selected for motif finding. MotifSampler was used to search for six motifs of length 8 bp and for six motifs of length 12 bp. A background model of order 3 was selected as it gave the most promising results. The analysis was repeated 10 times and only the motifs identified in at least five runs were retained. Table 3.5

TABLE 3.5 Results of Motif Search in Four Clusters From a Microarray Experiment on Mechanical Wounding in *A. Thaliana* for the Third-order Background Model

Cluster[a]	Consensus	Runs	PlantCARE	Descriptor
1	TAArTAAGTCAC	7/10	TGAGTCA	Tissue-specific GCN4 Motif
(11)			CGTCA	MeJA-responsive element
	ATTCAAATTT	8/10	ATACAAAT	Element associated to GCN4 motif
	CTTCTTCGATCT	5/10	TTCGACC	Elicitor-responsive element
2	TTGACyCGy	5/10	TGACG	MeJa-responsive element
(6)				
			(T)TGAC(C)	Box-W1, elicitor-responsive element
	mACGTCACCT	7/10	CGTCA	MeJA-responsive element
			ACGT	Abcissic acid response element
3	wATATATATmTT	5/10	TATATA	TATA-box-like element
(5)	TCTwCnTC	9/10	TCTCCCT	TCCC motif, light response element
	ATAAATAkGCnT	7/10		
5	yTGACCGTCCsA	9/10	CCGTCC	Meristem-specific activation of *H4* gene
(4)				
			CCGTCC	A-box, light/elicitor-responsive element
			TGACG	MeJA-responsive element
	CACGTGG	5/10	CGTCA	MeJA-responsive element
			CACGTG	G-box light-responsive element
			ACGT	Abcissic acid response element
	GCCTymTT	8/10		
	AGAATCAAT	6/10		

Source: From [69].

[a]Numbers in parentheses are number of sequences.

presents the motifs found. In the first column, the cluster is identified together with the number of genes it contains. The second column gives the consensus of the motif found. The consensus of a motif is the dominant DNA pattern in the motif described using a degenerate alphabet (e.g., r = A/G); capitals are for strong positions while lowercase letters are for degenerate positions. The third column gives the number of times this motif was found in the 10 runs. The fourth column gives matching known motifs found in the PlantCARE database [99], if any. Finally, the last column gives a short explanation of the matching known motifs.

Since we have made our applications available through both a Web interface and later also a stand-alone program, other researchers, mainly biologists, have been using these programs extensively. During the three-year period that MotifSampler has been online, almost 13,000 hits by more than 600 different users were registered. Some of those users could publish biological meaningful results obtained with MotifSampler [95, 100–105]. The Web interface of AQBC was created a few months later and has been used less extensively than the MotifSampler site, but we still counted almost 3000 hits by more than 250 users.

3.9. FURTHER INTEGRATIVE STEPS

The flow represented in this chapter going from coexpression information to motif detection is only a first approach. As the information content of the available data becomes richer (genomes sequenced and novel molecular biological techniques), a more elaborate integration of the data at the algorithmic level becomes a prerequisite. One of those techniques is chromatin immunoprecipitation (ChIP) DNA microarray (ChIP-chip) technology which allows direct mapping of in vivo physical interactions between transcriptional regulators and their binding sites at a high-throughput level [106–108].

For instance, ChIP-chip and motif data both contain information about the direct interactions between a regulator and its target genes. Microarray data provide complementary information by indicating the expression behavior of the coregulated genes. Independent analysis of each of the individual data sets describes the structure of the studied transcriptional network from a different angle [4]. Most described methods, as also the approach described in this chapter to combine the data, proceed sequentially: A prediction is based on a first data set (e.g., identification of coexpression based on cluster analysis) and is validated by a second complementary data set (e.g., motif data). However, because they contain complementary information, the simultaneous analysis of the data sets or the combination of the individual predictions enhances the reliability of the individual predictions and increases the confidence in the final result [31] (i.e., principle of meta-analysis). The development of methods that perform such combined analysis will be one of the future bioinformatics challenges. We can follow an iterative approach where the predictions based on a first data set are used as a priori information for the analysis of one or more complementary data sets and vice versa. Bussemaker et al. [109, 110] introduced REDUCE, which uses a linear regression model to find motifs that

correlate with the level of gene expression. Conlon et al. [111] propose a similar approach, but they use matrices instead of oligomers. Lapidot and Pilpel [112] detect simultaneously motifs and clustering of expression data. Bar-Joseph et al. [113] implement an iterative analysis of ChIP-chip data and clustering of expression data. Liu et al. [114] developed MDSCan for the simultaneous motif detection and analysis of ChIP-chip data.

Alternatively, methods can be used that simultaneously analyze both data sets (e.g., by combining more data sets into a single matrix, which is subsequently analyzed). Examples of such methods are supervised clustering, kernel methods, generalized singular-value decomposition, and canonical correlation analysis [115, 116].

3.10. CONCLUSION

We have presented algorithmic methods for the analysis of microarray data for motif finding. Microarrays are a powerful technique to monitor the expression of thousands of genes, and they have become key techniques for biologists attempting to unravel the regulation mechanisms of genes in an organism. After introducing some concepts from molecular biology to describe how transcription factors recognize binding sites to control gene activation, we reviewed the basics of microarray technology. We then introduced the strategy of integrating clustering (to detect groups of potentially coregulated genes) with motif finding (to detect the DNA motifs that control this coregulation). We then discussed the preprocessing steps necessary to prepare microarray data for clustering: normalization, nonlinear transformation, missing-value replacement, filtering, and rescaling. We presented several clustering techniques (such as hierarchical clustering, *K*-means, SOMs, quality-based clustering, and our AQBC) and discussed their respective advantages and shortcomings. We also presented several strategies to validate the results of clustering biologically as well as statistically. Turning to motif finding, we described the two main classes of methods for motif finding: word counting and probabilistic sequence models. We focused on the particular technique of Gibbs sampling for motif finding, where we discussed several extensions that improve the effectiveness of this method in practice. We introduced our MotifSampler, which in particular includes the use of higher order background models that increase the robustness of Gibbs sampling for motif finding. To be complete, we also mentioned the most important aspects of phylogenetic footprinting and module searching. Finally, we briefly presented our integrated Web tool INCLUSive, which allows the easy analysis of microarray data for motif finding. Furthermore, we illustrated the different steps of this integrated data analysis with several practical examples.

As a conclusion, we emphasize that a major endeavor of bioinformatics is to develop methodologies that integrate multiple types of data (here expression data together with sequence data) to obtain robust and biologically relevant results in an efficient and user-friendly manner. Only such powerful tools can deliver the necessary support twenty-first-century molecular biology.

ACKNOWLEDGMENTS

Our research was supported by grants from several funding agencies and sources:

- Research Council KUL: GOA AMBioRICS, CoE EF/05/007 SymBioSys, IDO Genetic networks
- Flemish government: FWO: Ph.D./postdoctorul grants, projects (G.0407.02, G.0413.03, G.0388.03, G.0229.03, G.0241.04, G.0499.04, G.0232.05, G.0318.05, G.0553.06), research communities (ICCoS, ANMMM, MLDM); IWT: Ph.D. grants, GBOU-McKnow, GBOU-SQUAD, GBOU-ANA, TAD-BioScope, Silicos
- Belgian Federal Science Policy Office: IUAP P5/22 (2002-2006)
- EU-RTD: FP5-CAGE; ERNSI; FP6-NoE Biopattern; FP6-IP e-Tumours, FP6-MC-EST Bioptrain

REFERENCES

1. J. Quackenbush,"Computational analysis of microarray data," *Nature Rev. Genet.*, 2: 418–427, 2001.
2. M. B. Eisen, P. T. Spellman, P. O. Brown, and D. Botstein, "Cluster analysis and display of genome-wide expression patterns," *Proc. Natl. Acad. Sci. USA*, 95: 14863–14868, 1998.
3. J. T. Tou and R.C. Gonzalez, *Pattern Recognition Principles*, Addison-Wesley, Reading, MA, 1979.
4. T. Hastie, R. Tibshirani, M. B. Eisen, A. Alizadeh, R. Levy, L. Staudt, W. C. Chan, D. Botstein, and P. Brown, "'Gene shaving' as a method for identifying distinct sets of genes with similar expression patterns," *Genome Biol.*, 1(2): 1–21, 2000.
5. F. De Smet, J. Mathys, K. Marchal, G. Thijs, B. De Moor, and Y. Moreau, "Adaptive quality-based clustering of gene expression profiles," *Bioinformatics*, 18(5): 735–746, 2002.
6. D. J. Duggan, M. Bittner, Y. Chen, P. Meltzer, and J. M. Trent, "Expression profiling using cDNA microarrays," *Nature Genet.*, 21(1, Suppl.): 10–14, 1999.
7. R. J. Lipschutz, S. P. A. Fodor, T. R. Gingeras, and D. J. Lockheart, "High density synthetic oligonucleotide arrays," *Nature Genet.*, 21(Suppl.): 20–24, 1999.
8. K. Engelen, B. Coessens, K. Marchal, and B. De Moor, "MARAN: normalizing micro-array data," *Bioinformatics*, 19(7): 893–894, 2003.
9. K. Marchal, K. Engelen, J. De Brabanter, S. Aerts., B. De Moor, T. Ayoubi, and P. Van Hummelen, "Comparison of different methodologies to identify differentially expressed genes in two-sample cdna arrays," *J. Biol. Sys.*, 10(4): 409–430, 2002.
10. O. Troyanskaya, M. Cantor, G. Sherlock, P. Brown, T. Hastie, R. Tibshirani, D. Botstein, and R. B. Altman, "Missing value estimation methods for DNA microarrays," *Bioinformatics*, 17: 520–525, 2001.
11. L. Kaufman and P. J. Rousseeuw, *Finding Groups in Data: An Introduction to Cluster Analysis*, Wiley, New York, 1990.
12. X. Zhou, X. Wang, and E. R. Dougherty, "Missing-value estimation using linear and non-linear regression with bayesian gene selection," *Bioinformatics*, 19(17): 2302–2307, 2003.

13. G. Thijs, Y. Moreau, F. De Smet, J. Mathys, M. Lescot, S. Rombauts, P. Rouzé, B. De Moor, and K. Marchal, "INCLUSive: INtegrated CLustering, Upstream sequence retrieval and motif Sampling," *Bioinformatics*, 18(2): 331–332, 2002.
14. F. P. Roth, J. D. Hughes, P. W. Estep, and G. M. Church, "Finding DNA regulatory motifs within unaligned noncoding sequences clustered by whole genome mRNA quantitation," *Nature Biotechnal*, 16: 939–945, 1998.
15. J. Van Helden, A. F. Rios, and J. Collado-Vides, "Discovering regulatory elements in non-coding sequences by analysis of spaced dyads," *Nucleic Acids Res.*, 28(8): 1808–1818, 2000.
16. A. Ben-Dor, R. Shamir, and Z. Yakhini, "Clustering gene expression patterns," *J. Comput. Biol.*, 6: 281–297, 1999.
17. R. J. Cho, M. J. Campbell, E. A. Winzeler, L. Steinmetz, A. Conway, L. Wodicka, T. G. Wolfsberg, A. E. Gabrielian, D. Landsman, D. J. Lockhart, and R. W. Davis, "A genome-wide transcriptional analysis of the mitotic cell cycle," *Mol. Cell*, 2: 65–73, 1998.
18. P. Tamayo, D. Slonim, J. Mesirov, Q. Zhu, S. Kitareewan, E. Dmitrovsky, E. S. Lander, and T. R. Golub, "Interpreting patterns of gene expression with self-organizing maps: Methods and application to hematopoietic differentiation," *Proc. Natl. Acad. Sci. USA*, 96: 2907–2912, 1999.
19. G. Sherlock, "Analysis of large-scale gene expression data," *Curr. Opin. Immunol.*, 12: 201–205, 2000.
20. J. Herrero, A. Valencia, and J. Dopazo, "A hierarchical unsupervised growing neural network for clustering gene expression patterns," *Bioinformatics*, 17: 126–136, 2001.
21. L. J. Heyer, S. Kruglyak, and S. Yooseph, "Exploring expression data: Identification and analysis of coexpressed genes," *Genome Res.*, 9: 1106–1115, 1999.
22. M. Dugas, S. Merk, S. Breit, and P. Dirschedl, "mdclust–exploratory microarray analysis by multidimensional clustering," *Bioinformatics*, 20(6): 931–936, 2004.
23. D. Ghosh and A. M. Chinnaiyan, "Mixture modelling of gene expression data from microarray experiments," *Bioinformatics*, 18: 275–286, 2002.
24. M. Medvedovic and S. Sivaganesan, "Bayesian infinite mixture model based clustering of gene expression profiles," *Bioinformatics*, 18(9): 1194–1206, 2002.
25. M. Medvedovic, K. Y. Yeung, and R. E. Bumgarner, "Bayesian mixture model based clustering of replicated microarray data," *Bioinformatics*, 20(6): 1222–1232, 2004.
26. K. Y. Yeung, C. Fraley, A. Murua, A. E. Raftery, and W. L. Ruzzo, "Model-based clustering and data transformations for gene expression data," *Bioinformatics*, 17: 977–987, 2001.
27. A. V. Lukashin and R. Fuchs, "Analysis of temporal gene expression profiles: Clustering by simulated annealing and determining the optimal number of clusters," *Bioinformatics*, 17: 405–414, 2001.
28. A. Ben-Dor, N. Friedman, and Z. Yakhini, "Class discovery in gene expression data," in T. Lengauer, D. Sankoff, S. Istra, and P. Peoner (Eds.), *Proceedings of the Fifth International Conference on Computational Biology*, Vol. 5, ACM Press, Montreal, Canada, 2001, pp. 31–38.

29. D. Dembele and P. Kastner, "Fuzzy c-means method for clustering microarray data," *Bioinformatics*, 19(8): 973–980, 2003.
30. D. R. Bickel, "Robust cluster analysis of microarray gene expression data with the number of clusters determined biologically," *Bioinformatics*, 19(7): 818–824, 2003.
31. S. Dudoit and J. Fridlyand, "A prediction-based resampling method for estimating the number of clusters in a dataset," *Genome Biol.*, 3(7): 36.1–36.21, 2002.
32. U. Alon, N. Barkai, D. A. Notterman, K. Gish, S. Ybarra, D. Mack, and A. J. Levine, "Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays," *Proc. Natl. Acad. Sci. USA*, 96: 6745–6750, 1999.
33. P. Reymond, H. Weber, M. Damond, and E. E. Farmer, "Differential gene expression in response to mechanical wounding and insect feeding in Arabidopsis," *Plant Cell*, 12: 707–719, 2000.
34. S. Tavazoie, J. D. Hughes, M. J. Campbell, R. J. Cho, and G. M. Church, "Systematic determination of genetic network architecture," *Nature Genet.*, 22(7): 281–285, 1999.
35. T. Kohonen, *Self-Organizing Maps*, Springer-Verlag, Berlin, 1997.
36. C. Fraley and E. Raftery, "MCLUST: Software for model-based cluster analysis," *J. Classification*, 16: 297–306, 1999.
37. G. Schwarz, "Estimating the dimension of a model," *Ann. Stat.*, 6: 461–464, 1978.
38. K. Y. Yeung and W. L. Ruzzo, "Principal component analysis for clustering gene expression data," *Bioinformatics*, 17: 763–774, 2001.
39. F. Azuaje, "A cluster validity framework for genome expression data," *Bioinformatics*, 18: 319–320, 2002.
40. K. Y. Yeung, D. R. Haynor, and W. L. Ruzzo, "Validating clustering for gene expression data," *Bioinformatics*, 17: 309–318, 2001.
41. M. Bittner, P. Meltzer, Y. Chen, Y. Jiang, E. Seftor, M. Hendrix, M. Radmacher, R. Simon, Z. Yakhini, A. Ben-Dor, N. Sampas, E. Dougherty, E. Wang, F. Marincola, C. Gooden, J. Lueders, A. Glatfelter, P. Pollock, J. Carpten, E. Gillanders, D. Leja, K. Dietrich, C. Beaudry, M. Berens, D. Alberts, and V. Sondak, "Molecular classification of cutaneous malignant melanoma by gene expression profiling," *Nature*, 406: 536–540, 2000.
42. M. K. Kerr and G. A. Churchill, "Bootstrap cluster analysis: Assessing the reliability of conclusions from microarray experiments," *Proc. Natl. Acad. Sci. USA*, 98: 8961–8965, 2001.
43. H. W. Mewes, D. Frishman, C. Gruber, B. Geier, D. Haase, A. Kaps, K. Lemcke, G. Mannhaupt, F. Pfeiffer, C. Schuller, S. Stocker, and B. Weil, "MIPS: A database for genomes and protein sequences," *Nucleic Acids Res.*, 28: 37–40, 2000.
44. P. Bucher, "Regulatory elements and expression profiles," *Curr. Opin. Struct. Biol.*, 9: 400–407, 1999.
45. U. Ohler and H. Niemann, "Identification and analysis of eukaryotic promoters: Recent computational approaches," *Trends Genet.*, 17(2): 56–60, 2001.
46. J. Zhu and M. Q. Zhang, "Cluster, function and promoter: Analysis of yeast expression array," *Proc. Pacific Symp. Biocomput.*, 5: 467–486, 2000.
47. M. Tompa, N. Li, T. L. Bailey, G. M. Church, B. De Moor, E. Eskin, A. V. Favorov, M. C. Frith, Y. Fu, W. J. Kent, V. J. Makeev, A. A. Mironov, W. S. Noble,

G. Pavesi, G. Pesole, M. Regnier, N. Simonis, S. Sinha, G. Thijs, J. van Helden, M. Vandenbogaert, Z. Weng, C. Workman, C. Ye, and Z. Zhu, "Assessing computational tools for the discovery of transcription factor binding sites," *Nature Biotechnol.*, 23(1): 137–144, 2005.

48. G. Pavesi, P. Mereghetti, G. Mauri, and G. Pesole, "Weeder Web: Discovery of transcription factor binding sites in a set of sequences from co-regulated genes," *Nucleic Acids Res.*, 32(2, Suppl.): W199–203, 2004.
49. J. van Helden, B. André, and L. Collado-Vides, "Extracting regulatory sites from upstream region of yeast genes by computational analysis of oligonucleotide frequencies," *J. Mol. Biol.*, 281: 827–842, 1998.
50. T. D. Schneider and R. M. Stephens, "Sequence logos: A new way to display consensus sequences," *Nucleic Acids Res.*, 18(20): 6097–6100, 1990.
51. T. Werner, "Models for prediction and recognition of eukaryotic promoters," *Mamm. Genome*, 10: 71–80, 1999.
52. M. Tompa, "An exact method for finding short motifs in sequences, with application to the ribosome binding site problem," *Proc. Int. Conf. Intell. Sys. Mol. Biol.* (Heidelberg, Germany), 7: 262–271, 1999.
53. P. A. Pevzner and S. H. Sze, "Combinatorial approaches to finding subtle signals in dna sequences," *Proc. Int. Conf. Intell. Sys. Mol. Biol.*, 8: 269–278, 2000.
54. U. Keich and P. A. Pevzner, "Subtle motifs: Defining the limits of motif finding algorithms," *Bioinformatics*, 18(10): 1382–1390, 2002.
55. U. Keich and P. A. Pevzner, "Finding motifs in the twilight zone," *Bioinformatics*, 18(10): 1374–1381, 2002.
56. J. Buhler and M. Tompa, "Finding motifs using random projections," *J. Comput. Biol.*, 9(2): 225–242, 2002.
57. L. Marsan and M.-F. Sagot, "Algorithms for extracting structured motifs using a suffix tree with application to promoter and regulatory site consensus identification," *J. Comp. Biol.*, 7: 345–360, 2000.
58. A. Vanet, L. Marsan, A. Labigne, and M. F. Sagot, "Inferring regulatory elements from a whole genome. An analysis of Helicobacter pylori sigma^{80} family of promoter signals," *J. Mol. Biol.*, 297(2): 335–353, 2000.
59. G. Z. Hertz, G. W. Hartzell, and G. D. Stormo, "Identification of consensus patterns in unaligned *DNA* sequences known to be functionally related," *Comput. Appl. Biosci.*, 6: 81–92, 1990.
60. G. Z. Hertz and G. D. Stormo, "Identifying DNA and protein patterns with statistically significant alignments of multiple sequences," *Bioinformatics*, 15(7/8): 563–577, 1999.
61. C. E. Lawrence and A. A. Reilly, "An expectation maximization (EM) algorithm for the identification and characterization of common sites in unaligned biopolymer sequences," *Proteins*, 7: 41–51, 1990.
62. T. L. Bailey and C. Elkan, "Unsupervised learning of multiple motifs in biopolymers using expectation maximization," *Machine Learning*, 21: 51–80, 1995.
63. T. L. Bailey and C. Elkan, "The value of prior knowledge in discovering motifs with *MEME*," *Proc. Int. Conf. Intell. Sys. Mol. Biol.*, 3: 21–29, 1995.
64. L. R. Cardon and G. D. Stormo, "Expectation maximization for identifying protein-binding sites with variable lengths from unaligned *DNA* fragments," *J. Mol. Biol.*, 223: 159–170, 1992.

65. C. E. Lawrence, S. F. Altschul, M. S. Boguski, J. S. Liu, A. F. Neuwald, and J. C. Wootton, "Detecting subtle sequence signals: A *Gibbs* sampling strategy for multiple alignment," *Science*, 262: 208–214, 1993.

66. J. S. Liu, A. F. Neuwald, and C. E. Lawrence, "Bayesian models for multiple local sequence alignment and *Gibbs* sampling strategies," *J. Am. Stat. Assoc.*, 90(432): 1156–1170, 1995.

67. J. D. Hughes, P. W. Estep, S. Tavazoie, and G. M. Church, "Computational identification of cis-regulatory elements associated with groups of functionally related genes in *Saccharomyces cerevisiae*," *J. Mol. Biol.*, 296: 1205–1214, 2000.

68. G. Thijs, M. Lescot, K. Marchal, S. Rombauts, B. De Moor, P. Rouzé, and Y. Moreau, "A higher order background model improves the detection of promoter regulatory elements by *Gibbs* sampling," *Bioinformatics*, 17(12): 1113–1122, 2001.

69. G. Thijs, K. Marchal, M. Lescot, S. Rombauts, B. De Moor, P. Rouzé, and Y. Moreau, "A Gibbs sampling method to detect over-represented motifs in the upstream regions of co-expressed genes," *J. Comput. Biol.*, 9(2): 447–464, 2002.

70. K. Marchal, G. Thijs, S. De Keersmaecker, P. Monsieurs, B. De Moor, and J. Vanderleyden, "Genome-specific higher-order background models to improve motif detection," *Trends Microbiol.*, 11(2): 61–66, 2003.

71. X. Liu, D. L. Brutlag, and J. S. Liu, "BioProspector: Discovering conserved DNA motifs in upstream regulatory regions of co-expressed genes," *Proc. Pacific Symp. Biocomput.*, 6: 127–138, 2001.

72. C. T. Workman and G. D. Stormo, "Ann-spec: A method for discovering transcription binding sites with improved specificity," *Proc. Pacific Symp. Biocomput.* (Honolulu, Hawai), 5: 464–475, 2000.

73. D. GuhaThakurta and G. D. Stormo, "Identifying target sites for cooperatively binding factors," *Bioinformatics*, 17: 608–621, 2001.

74. B. P. Berman, Y. Nibu, B. D. Pfeiffer, P. Tomancak, S. E. Celniker, M. Levine, G. M. Rubin, and M. B. Eisen, "Exploiting transcription factor binding site clustering to identify cis-regulatory modules involved in pattern formation in the *Drosophila*," *Proc Natl. Acad. Sci. USA*, 99(2): 757–762, 2002.

75. M. S. Halfon, Y. Grad, G. M. Church, and A. M. Michelson, "Computation-based discovery of related transcriptional regulatory modules and motifs using an experimentally validated combinatorial model," *Genome Res.*, 12(7): 1019–1028, 2002.

76. O. Johansson, W. Alkema, W. W. Wasserman, and J. Lagergren, "Identification of functional clusters of transcription factor binding motifs in genome sequences: The mscan algorithm," *Bioinformatics*, 19(1, Suppl.): I169–I176, 2003.

77. M. Markstein and M. Levine, "Decoding cis-regulatory dnas in the *Drosophila* genome," *Curr. Opin. Genet. Dev.*, 12(5): 601–606, 2002.

78. M. Rebeiz, N. L. Reeves, and J. W. Posakony, "SCORE: A computational approach to the identification of cis-regulatory modules and target genes in whole-genome sequence data. Site clustering," *Proc. Natl. Acad. Sci. USA*, 99(15): 9888–9893, 2002.

79. E. M. Crowley, K. Roeder, and M. Bina, "A statistical model for locating regulatory regions in genomic DNA," *J. Mol. Biol.*, 268: 8–14, 1997.

80. N. Rajewsky, M. Vergassola, U. Gaul, and E. D. Siggia, "Computational detection of genomic cis-regulatory modules applied to body patterning in the early *Drosophila* embryo," *BMC Bioinformatics*, 3(1): 30, 2002.

81. M. C. Frith, U. Hansen, and Z. Weng, "Detection of cis-element clusters in higher eukaryotic dna," *Bioinformatics*, 17(10): 878–879, 2001.

82. M. C. Frith, J. L. Spouge, U. Hansen, and Z. Weng, "Statistical significance of clusters of motifs represented by position specific scoring matrices in nucleotide sequences," *Nucleic Acids Res.*, 30(14): 3214–3224, 2002.

83. T. L. Bailey and W. S. Noble, "Searching for statistically significant regulatory modules," *Bioinformatics*, 19(2, Suppl.): II16–II25, 2003.

84. S. Sinha, E. Van Nimwegen, and E. D. Siggia, "A probabilistic method to detect regulatory modules," *Bioinformatics*, 19(1, Suppl.): I292–I301, 2003.

85. S. Aerts, P. Van Loo, G. Thijs, Y. Moreau, and B. De Moor, "Computational detection of cis-regulatory modules," *Bioinformatics*, 19(2, Suppl.): II5–II14, 2003.

86. L. A. McCue, W. Thompson, C. S. Carmack, M. P. Ryan, J. S. Liu, V. Derbyshire, and C. E. Lawrence, "Phylogenetic footprinting of transcription factor binding sites in proteobacterial genomes," *Nucleic Acids Res.*, 29: 774–782, 2001.

87. J. S. Liu and C. E. Lawrence, "Bayesian inference on biopolymer models," *Bioinformatics*, 15: 38–52, 1999.

88. M. Blanchette, B. Schwikowski, and M. Tompa, "Algorithms for phylogenetic footprinting," *J. Comput. Biol.*, 9(2): 211–223, 2002.

89. M. Blanchette and M. Tompa, "Discovery of regulatory elements by a computational method for phylogenetic footprinting," *Genome Res.*, 12(5): 739–748, 2002.

90. M. Blanchette and M. Tompa, "Footprinter: A program designed for phylogenetic footprinting," *Nucleic Acids Res.*, 31(13): 3840–3842, 2003.

91. T. Wang and G. D. Stormo, "Combining phylogenetic data with coregulated genes to identify regulatory motifs," *Bioinformatics*, 19(18): 2369–2380, 2003.

92. E. Berezikov, V. Guryev, R. H. Plasterk, and E. Cuppen, "CONREAL: Conserved regulatory elements anchored alignment algorithm for identification of transcription factor binding sites by phylogenetic footprinting," *Genome Res.*, 14(1): 170–178, 2004.

93. P. Cliften, P. Sudarsanam, A. Desikan, L. Fulton, B. Fulton, J. Majors, R. Waterston, B. A. Cohen, and M. Johnston, "Finding functional features in *Saccharomyces* genomes by phylogenetic footprinting," *Science*, 301(5629): 71–76, 2003.

94. B. Lenhard, A. Sandelin, L. Mendoza, P. Engstrom, N. Jareborg, and W. W. Wasserman, "Identification of conserved regulatory elements by comparative genome analysis," *J. Biol.*, 2(2): 13, 2003.

95. K. Marchal, S. De Keersmaecker, P. Monsieurs, N. van Boxel, K. Lemmens, G. Thijs, J. Vanderleyden, and B. De Moor, "In silico identification and experimental validation of pmrab targets in *Salmonella typhimurium* by regulatory motif detection," *Genome Biol.*, 5(2): R9, 2004.

96. J. van Helden, B. Andre, and J. Collado-Vides, "Extracting regulatory sites from the upstream region of yeast genes by computational analysis of oligonucleotide frequencies," *J. Mol. Biol.*, 281(5): 827–842, 1998.

97. M. Blanchette, B. Schwikowski, and M. Tompa, "An exact algorithm to identify motifs in orthologous sequences from multiple species," *Proc. Int. Conf. Intell. Sys. Mol. Biol.*, 8: 37–45, 2000.

98. H. J. Bussemaker, H. Li, and E. D. Siggia, "Regulatory element detection using a probabilistic segmentation model," *Proc. Int. Conf. Intell. Sys. Mol. Biol.*, 8: 67–74, 2000.

99. M. Lescot, P. Déhais, G. Thijs, K. Marchal, Y. Moreau, Y. Van de Peer, P. Rouzé, and S. Rombauts, "PlantCARE, a database of plant cis-acting regulatory elements and a portal to tools for in silico analysis of promoter sequences," *Nucleic Acids Res.*, 30: 325–327, 2002.

100. W. Chen, N. J. Provart, J. Glazebrook, F. Katagiri, H.-S. Chang, T. Eulgem, F. Mauch, S. Luan, G. Zou, S. A. Whitham, P. R. Budworth, Y. Tao, Z. Xie, X. Chen, S. Lam, J. A. Kreps, J. F. Harper, A. Si-Ammour, B. Mauch-Mani, M. Heinlein, K. Kobayashi, T. Hohn, J. L. Dangl, X. Wang, and T. Zhu, "Expression profile matrix of *Arabidopsis* transcription factor genes suggests their putative functions in response to environmental stresses," *Plant Cell*, 14(3): 559–574, 2002.

101. E. J. Klok, I. W. Wilson, D. Wilson, S. C. Chapman, R. M. Ewing, S. C. Somerville, W. J. Peacock, R. Dolferus, and E. S. Dennis, "Expression profile analysis of the low-oxygen response in *Arabidopsis* root cultures," *Plant Cell*, 14: 2481–2494, 2002.

102. S. Le Crom, F. Devaux, P. Marc, X. Zhang, W. S. Moye-Rowley, and C. Jacq, "New insights into the pleiotropic drug resistance network from genome-wide characterization of the YRR1 transcription factor regulation system," *Mol. Cell. Biol.*, 22(8): 2642–2649, 2002.

103. I. M. Martinez and M. J. Chrispeels, "Genomic analysis of unfolded protein response in *Arabidopsis* shows its connection to important cellular processess" *Plant Cell*, 15(2): 561–576, 2003.

104. U. Ohler, G. Liao, H. Niemann, and G. M. Rubin, "Computational analysis of core promoters in the *Drosophila* genome," *Genome Biol.*, 3(12): 1–12, 2002.

105. J. B. Rossel, I. W. Wilson, and B. J. Pogson, "Global changes in gene expression in response to high light in *Arabidopsis*," *Plant Physiol.*, 130: 1109–1120, 2002.

106. M. T. Laub, H. H. McAdams, T. Feldblyum, C. M. Fraser, and L. Shapiro, "Global analysis of the genetic network controlling a bacterial cell cycle," *Science*, 290(5499): 2144–2148, 2000.

107. T. I. Lee, N. J. Rinaldi, F. Robert, D. T. Odom, Z. Bar-Joseph, G. K. Gerber, N. M. Hannett, C. T. Harbison, C. M. Thompson, I. Simon, J. Zeitlinger, E. G. Jennings, H. L. Murray, D. B. Gordon, B. Ren, J. J. Wyrick, J. B. Tagne, T. L. Volkert, E. Fraenkel, D. K. Gifford, and R. A. Young, "Transcriptional regulatory networks in *Saccharomyces cerevisiae*," *Science*, 298(5594): 799–804, 2002.

108. J. R. Pollack, C. M. Perou, A. A. Alizadeh, M. B. Eisen, A. Pergamenschikov, C. F. Williams, S. S. Jeffrey, D. Botstein, and P. O. Brown, "Genome-wide analysis of DNA copy-number changes using cDNA microarrays," *Nature Genet.*, 23(1): 41–46, 1999.

109. H. J. Bussemaker, H. Li, and E. D. Siggia, "Regulatory element detection using correlation with expression," *Nature Genet.*, 27(2): 167–171, 2001.

110. C. Roven and H. J. Bussemaker, "Reduce: An online tool for inferring cis-regulatory elements and transcriptional module activities from microarray data," *Nucleic Acids Res.*, 31(13): 3487–3490, 2003.

111. E. M. Conlon, X. S. Liu, J. D. Lieb, and J. S. Liu, "Integrating regulatory motif discovery and genome-wide expression analysis," *Proc. Natl. Acad. Sci. USA*, 100(6): 3339–3344, 2003.

112. M. Lapidot and Y. Pilpel, "Comprehensive quantitative analyses of the effects of promoter sequence elements on mrna transcription," *Nucleic Acids Res.*, 31(13): 3824–3828, 2003.

113. Z. Bar-Joseph, G. K. Gerber, T. I. Lee, N. J. Rinaldi, J. Y. Yoo, F. Robert, D. B. Gordon, E. Fraenkel, T. S. Jaakkola, R. A. Young, and D. K. Gifford, "Computational discovery of gene modules and regulatory networks," *Nature Biotechnol.*, 21(11): 1337–1342, 2003.

114. X. S. Liu, D. L. Brutlag, and J. S. Liu, "An algorithm for finding protein-DNA binding sites with applications to chromatin-immunoprecipitation microarray experiments," *Nature Biotechnol.*, 20(8): 835–839, 2002.

115. O. Alter, P. O. Brown, and D. Botstein, "Generalized singular value decomposition for comparative analysis of genome-scale expression data sets of two different organisms," *Proc. Natl. Acad. Sci. USA*, 100(6): 3351–3356, 2003.

116. J. Suykens, T. van Gestel, J. De Brabander, B. De Moor, and J. Vandewalle, *Least Square Support Vector Machines*, World Scientific Publishing, Singapore, 2002.

CHAPTER 4

Robust Methods for Microarray Analysis

GEORGE S. DAVIDSON, SHAWN MARTIN, KEVIN W. BOYACK, BRIAN N. WYLIE, JUANITA MARTINEZ, ANTHONY ARAGON, MARGARET WERNER-WASHBURNE, MÓNICA MOSQUERA-CARO, and CHERYL WILLMAN

4.1. INTRODUCTION

The analysis of a complex system within an environment that is only subject to incomplete control is nearly impossible without some way to measure a large fraction of the system's internal state information. As a result, it is only with the recent advent of high-throughput measurement technologies (able to simultaneously measure tens of thousands of molecular concentrations) that systems biology is really a possibility. As an example of the scope of this problem, consider that eukaryotic cells typically have on the order of tens of thousands of genes, each of which is likely to have several alternative splicing variants coding for the protein building blocks of the cell. These proteins undergo posttranslational modifications and have multiple phosphorylations such that there are likely to be hundreds of thousands, possibly millions, of variants. Hence the future of systems biology relies critically on high-throughput instruments, such as microarrays and dual mass spectrometers. The research reported here addresses the need for improved informatics to deal with the large volume of information from these techniques.

At present, microarray data are notoriously noisy. Hopefully this technology will improve in the future, but for the immediate present, it is important that the analysis tools and informatics systems developed for microarray analysis admit and adjust to significant levels of noise. In particular, such methods should be stable in the presence of noise and should assign measures of confidence to their own results. In this chapter we describe our efforts to implement and assess reliable methods for microarray analysis. We begin with the structure of the typical microarray experiment, normalization of the resulting data, and ways to find relationships. We then

Genomics and Proteomics Engineering in Medicine and Biology. Edited by Metin Akay

proceed to discuss the tools themselves as well as methods to assess the output of such tools.

Much of the work in this chapter is based on the VxInsight visualization software [1]. However, we have tried to minimize overlap with existing published results and have emphasized new work.

Throughout each section we will present results and examples from our research to motivate the specific approaches, algorithms, and analysis methods we have developed. The data sets we have analyzed will not be emphasized, although they were of course critical to the work. Most of the data sets we have used have been published, and these will be cited in the course of the chapter. The data sets so far unpublished have been provided by the University of New Mexico Cancer Research and Treatment Center. These data sets include an infant leukemia data set, a precursor-B childhood leukemia data set, and an adult leukemia data set, all of which have been presented at the American Society of Hematology conferences in 2002 and/or 2003. Publications concerning the biological implications of these data sets are forthcoming. Here we discuss methods only.

Finally, we emphasize that this chapter is by no means a survey of techniques for microarray analysis. Indeed, the literature on microarray analysis is vast, and new papers appear frequently. We will mention, however, some of the seminal papers, including [2, 3], which describe the original technology, as well as [4], which gives an overview of the earlier work on microarray technology. Of course, microarrays would not be terribly useful without computational analysis, and some of the original work on microarray analysis includes hierarchical clustering of the yeast cell cycle [5, 6], an analysis of leukemia microarray data using an original method of gene selection [7], and an application of the singular-value decomposition to microarray data [8]. In addition to innumerable additional papers, there are a variety of books available which emphasize different aspects of the microarrays and microarray analysis. A few of the more recent books include [9–12].

4.2. MICROARRAY EXPERIMENTS AND ANALYSIS METHODS

4.2.1. Microarray Experiments

Microarray experiments and their analyses seek to detect effects in gene expression under different treatments or natural conditions with the goal of clarifying the cellular mechanisms involved in the cells' differential responses. Uncertainty is the rule rather than the exception in these analyses. First, the underlying systems (cells and/or tissues) are incredibly complex whether viewed from the dynamic process perspective or their physical realizations in space and time. Second, there is abundant variability between cells experiencing exactly the same conditions as a result of genetic polymorphisms as well as the stochastic nature of these chemical systems. Third, the collection and initial preservation of these cells are not a precisely controlled process. For example, when a surgeon removes a cancer, the tissue may not be frozen or otherwise processed for minutes to hours, all the

while the cells continue to respond to these unnatural conditions. Further, the tissues, or partially processed extracts, are often sent to another laboratory several hours or even days away from the original collection site, all of which offers opportunities for chemical changes. Fourth, these measurements are not easy to make; they involve many processing steps with a wide variety of chemicals, and at every step variability arises. The processing is often (necessarily) divided across several days and among several technicians, with inherently different skills and training. Further, the chemicals are never exactly the same; they are created in different batches and age at different rates. All of these affect the laboratory yields and the quality of the processing. Finally, the arrays themselves are technological objects subject to all sorts of variability in their creation, storage, and final use. In effect, the simple measurement of mRNA concentrations that we would like to make is confounded by huge uncertainties. To be able to make good measurements, it is essential that all of the mentioned steps be subject to careful statistical process control, monitoring, and systematic improvement. Further, the actual experiments should be designed to avoid, randomize, or otherwise balance the confounding effects for the most important experimental measurements [13, 14]. These are *best practices*. Unfortunately, they are not often followed in the laboratory.

By the time the data are ready for analysis, they are typically presented in a numeric table recording a measurement for each gene across several microarrays. For example, one might be analyzing 400 arrays each with 20,000 gene measurements, which would be presented in a table with 20,000 rows and 400 columns. Often the table will have missing values resulting from scratched arrays, poor hybridizations, or scanner misalignments, to name just a few (from among a host) of the possible problems. The analysis methods should be able to gracefully deal with these incomplete data sets, while the analyst should approach these data with great skepticism and humility considering the complexity of the cellular processes and the error-prone nature of our microarray technology. Despite all of these issues, statisticians and informaticians, unlike mathematicians, are expected to say something about the structure and meaning of these data. Because, as Thompson has said, "[statisticians] *should be concerned with a reality beyond data-free formalism*. That is why statistics is not simply a subset of mathematics" [15, p. xv]. Here, we attempt to follow Thompson in discussing implications of, as well as our approaches to, analyzing these experiments, including considerable detail about the algorithms and the way the data are handled.

4.2.2. Preprocessing and Normalization

As discussed in the previous section, microarray data typically have a large number of missing data, or values otherwise deemed to be nonpresent. We typically drop genes with too many missing values, assigning a threshold for this purpose that is under the control of the analyst. The raw values are then scaled to help with the processing.

The distributions of microarray measurement values typically have extremely long tails. There are a few genes with very large expressions, while most of the

others are quite small. Tukey and Mosteller suggested a number of transforms to make data from such distributions less extreme and more like the normal Gaussian distribution [16]. In particular, taking logarithms of the raw data is a common practice to make microarray data more symmetric and to shorten the extreme tail (see Fig. 4.1*a*).

However, we frequently use another transform to compress the extreme values, which is due to [17]. This rank-order-based score is an increasing function of expression level, for which the smaller values are compressed to be very nearly the same. This is particularly useful with array data, as many of these smaller values are due purely to noise. The Savage score is computed as follows: If the absolute expression levels are rank ordered from smallest to largest, $X_{(1)} \leq X_{(2)} \leq \cdots \leq X_{(n)}$, then the score for $X_{(k)}$ is given by

$$\mathrm{SS}(X_{(k)}) = \sum_{j=1}^{k} \frac{1}{n+1-j}$$

Figure 4.1*b* shows how this score affects the resulting distribution. In particular, Savage scoring compresses the extreme tail and emphasizes the intermediate and large counts. In this case, about 60% of the savage scores are below 1.

The use of this scoring has two advantages over correlations using raw counts. First, because it is based on rank ordering, data from arrays processed with very different scales can still be compared. Second, because the noisiest fraction of the measurements is aggressively forced toward zero, the effect of the noise is suppressed without completely ignoring the information (it has been taken into account during the sorting phase). Finally, large differences in rank order will still be strong enough to be detected.

The normalization of array data is controversial, although some form of centering and variance normalization is often applied [18, 19]. However, it has been argued

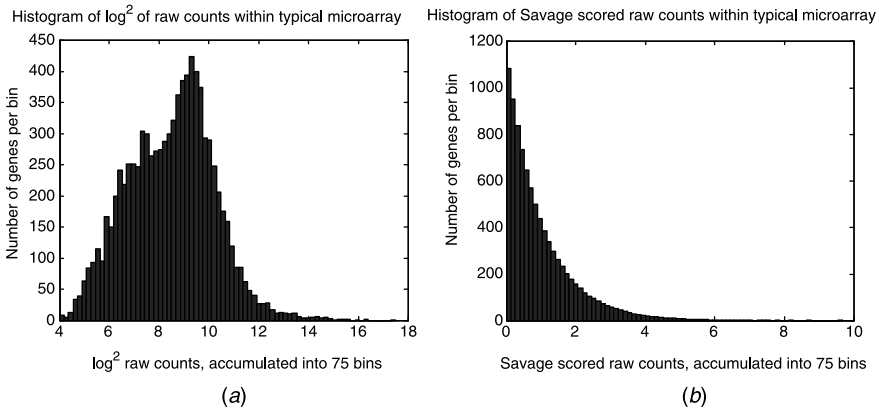


FIGURE 4.1. (*a*) Distribution of log-transformed data from typical Affymetrix U94A microarray. (*b*) Distribution of the same data after Savage scores.

that for many experiments there is no intrinsic reason to expect the underlying mRNA concentrations to have the same mean (or median) values, and hence variance adjustment is suspect. Nevertheless, the analyst has the option to do such normalizations, if desired. In general, we avoid this issue by working with order statistics and Savage scores.

4.2.3. Overview of Basic Analysis Method

After preprocessing the measurements with thresholding, rescaling, and various other transforms, we often perform our analysis as follows. First, we compare the genes or arrays under investigation by computing pairwise similarities with several techniques. These similarities are used to cluster (or assign spatial coordinates to) the genes in ways that bring similar genes closer together. These clusters are then visualized with VxInsight [1, 20].

Although we typically use genes in our analyses, we often use arrays as well. In fact, the analysis is the same, as all our techniques use either the gene matrix initially described, or the transpose of the gene matrix. Hence, throughout the remainder of this chapter, we will use the terms *genes* and *arrays* interchangeably.

Next, the clusters are tested with statistical methods to identify genes and groups of genes that are differentially expressed in the identified clusters or genes otherwise identified with respect to experimental questions and hypotheses. The expression values for the identified genes are plotted and tested for stability. Those genes which seem particularly diagnostic or differentiating are studied in detail by reading the available information in online databases and in the original literature. Each of these analysis steps will be presented in an order approximately following the analysis order we use in practice.

This type of analysis is quite typical for microarrays and is usually divided into two categories. The method first described for clustering genes is known as cluster analysis, or *unsupervised* learning. The term unsupervised is used because we are attempting to divine groups within the data set while avoiding the use of a priori knowledge. In contrast, the second method described for the identification of gene lists is usually called *supervised* learning. The term supervised refers to the fact that we are using prior information in our analysis, in this case attempting to discover which genes are differentially expressed between known groups. An unsupervised method asks, in essence: Are there groups in the data set and, if so, what are they? A supervised method asks: Given groups, can we predict group membership and can we learn what is most important in distinguishing the groups?

4.3. UNSUPERVISED METHODS

4.3.1. Overview of Clustering for Microarray Data

Organizing large groups of data into clusters is a standard and powerful practice in exploratory data analysis. It has also become a standard practice in microarray

analysis. Typically, the first step after the initial data transformations involves the pairwise comparison of the data elements to determine their relative similarity or dissimilarity. A single comparison usually results in a single number. Thus when comparing the expressions of n genes across multiple experimental conditions, one might compute $n(n-1)/2$ correlations using the similarity measure between each possible pair of genes. After the data pairs have been assigned similarities, various grouping algorithms can be used.

In the case of microarray analysis, there are a variety of methods that have been applied and/or developed. These include hierarchical clustering [5], the singular value decomposition/principal-component analysis [8], and self-organizing maps [21]. Our method (discussed next) belongs to the class of graph-theoretic clustering methods. Other such methods include those presented in [22–24]. Finally, a general overview of clustering methods can be found in [25].

In this chapter we focus on a tool known as VxOrd [20], which uses a force-directed graph layout algorithm. Our method requires, as do most clustering algorithms, a single similarity value for each of the data pairs. In the next section we show one way to compute the similarities and then consider the actual clustering algorithm.

4.3.2. Clustering Using VxOrd

4.3.2.1. Choosing a Similarity Measure One obvious candidate for measuring similarities is the simple correlation coefficient R due to Pearson [26],

$$R_{xy} = \frac{\sum_{i=1}^{d} (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{d} (x_i - \bar{x})^2 \sum_{i=1}^{d} (y_i - \bar{y})^2}}$$

where, of course, $-1 \leq R_{xy} \leq 1$. Pearson's R is just a dot product of two mean-centered and normalized d-dimensional vectors. Thus, one can think of R as a measure of the extent to which the two vectors point in the same direction in the d-dimensional space. Of course, the vectors might lie along the same line, but point in opposite directions, which is the meaning of $R_{xy} = -1$. If the vectors are completely orthogonal, the correlation will be zero.

In fact, Pearson's correlation is the measure of similarity that we, and the rest of the microarray community, use most often. It is, however, not the only possibility and in fact has some features that do not recommend it under certain situations. For instance, Pearson's correlation is known to be sensitive to outliers, especially when n is small. Technically, R has a breakdown point of $1/n$, meaning that as few as one extreme outlier in the n points can make the statistic completely different from the true measure of correlation for two random but correlated variables [26]. In practice, we have not found this to be a real problem, since we typically use hundreds of arrays. However, early in the development of microarray technology, many data sets were published with order tens of arrays. In these cases, it was deemed valuable to apply more computationally expensive but more robust

measures of correlation. Details about robust measures of correlation, including the percentage-bend correlation coefficient, can be found in [27].

We have also found occasion to use very different similarity measures. For example, we clustered genes based on the textual similarity of their annotations in the Online Mendelian Inheritance in Man (OMIM) [28] (http://www.ncbi.nlm.nih.gov/omim). In this case, the text was processed with the RetrievalWare search and retrieval package from Convera. RetrievalWare computes the similarity between two text documents with a proprietary algorithm. For each gene annotation the strongest 19 other annotations were accumulated to create a similarity file and then processed to produce a clustering. The more typical processing for microarrays is discussed in the following section.

4.3.2.2. Postprocessing for Similarity Measures While Pearson's correlation has a breakdown point of $1/n$ (a single outlier can distort the statistic from one extreme to the other [26]), it is easy to compute and has been widely accepted in the microarray community. Because Savage-scored expression values are bounded, the influence of outliers is less important. As a result, the correlation coefficient is usually the basis of our similarity computations. When too few arrays are available to have confidence in R, the percentage-bend coefficient [27] is to be used instead.

It is common to cluster directly with these coefficients. However, doing so ignores much of the available information because R is such a nonlinear function. For example, there is a slight change in significance when comparing two pairs of genes that have $R = 0.5$ and $R = 0.55$, respectively, but the relative significance between $R = 0.90$ and $R = 0.95$ can be quite large. Consequently, it is better to transform these correlations with a measure of their relative significance. This transform can be done by converting to the t-statistic for the observed correlation R between the pairs of values [26]:

$$t = \frac{R\sqrt{d-2}}{\sqrt{1-R^2}}$$

Both R and t have computational issues that should be addressed. In particular, R is undefined when the variance of either x or y vanishes, and hence a minimum, acceptable variance must be determined. We typically require that

$$\sum_{i=1}^{d}(x_i - \bar{x})^2 \sum_{i=1}^{d}(y_i - \bar{y})^2 > 0.0001$$

Otherwise, no correlation is computed for the pair of expression vectors. A related issue occurs with t when R approaches ± 1.0. In this case, the t-statistic becomes arbitrarily large. Because clustering will be comparing similarities, the strength of an extreme outlier will distort the clustering. Hence t should be clipped to avoid such extremes (we typically truncate values greater than 300, though even this value may be extreme).

Missing data also present concerns in computing R. Certainly, if too many values are missing, any computed similarity would be suspect. Recourse to the analyst's experience and judgment is the best way to choose how many values can be missing before the comparison is not attempted. For large collections of arrays, requiring that at least 70 measurements be simultaneously present for both expression vectors has been acceptable in our experience.

Computing all of these correlations produces a huge file of similarity comparisons. For example, the computation for an experiment [29] around *Clostridium elegans*, which has about 20,000 genes, required the computation of about 2×10^8 correlations. Using all of the correlations for clustering is neither necessary nor desirable. Most of the correlations will be modest and including them slows the clustering computation and introduces a great deal of resistance to the process that separates the mass of genes into clusters. If a particular gene has strong correlations with a few tens of other genes, they should eventually be clustered together. However, if there are hundreds or thousands of correlations weakly linking that particular gene to others, then the net sum of these weak correlations may overwhelm the few strong correlations.

If only a few of the correlations will be used for clustering, some method of choice is required. The analyst can use all correlations above some threshold or just the strongest few correlations for each gene. We have found the latter approach to be sufficient. We have found that using the 20 strongest correlations is often a good starting point. However, even fewer correlations may suffice, especially with the methods discussed next for finding the most central ordination from a distribution of stochastic clustering results.

4.3.2.3. Clustering by Graph Layout Once the similarities have been computed, the VxOrd algorithm is used to cluster the data. VxOrd considers the data set as an abstract graph. In the case of microarrays, we usually think of the genes as nodes in a graph, and the edges as similarities between genes. The relationship between the data elements and their similarity values can be visualized as an abstract, weighted graph $G(V_i, E_{i,j}, W_{i,j})$ consisting of a set of vertices V (the genes) and a set of edges E with weights W (the similarities between the genes). This graph is only topologically defined, as the vertices have not been assigned spatial locations. Spatial coordinates, called *ordinations*, are computed from the weighted graph using VxOrd, which places vertices into clusters on a two-dimensional plane, as shown in Figure 4.2. The ordinations are computed such that the sum of two opposing forces is minimized. One of these forces is repulsive and pushes pairs of vertices away from each other as a function of the density of vertices in the local area. The other force pulls pairs of similar vertices together based on their degree of similarity. The clustering algorithm stops when these forces are in equilibrium.

Although the algorithm has been discussed in detail in a previous paper [20], we provide here a brief overview for convenience. VxOrd is based on the approach in [30]. Fruchtermann and Rheingold compute a force term for both attraction and repulsion. These terms are then used to generate new positions for the graph vertices. The algorithm combines the attraction and repulsion terms into one potential energy equation, shown below. The first term, in brackets, is due to the attraction

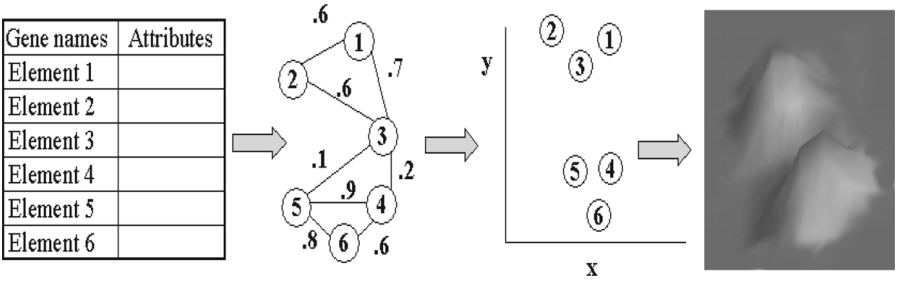


FIGURE 4.2. Data elements are nodes and similarities are edge values, which are clustered and assigned (x, y) coordinates by VxOrd. These coordinates are then used to visualize clusters, as shown on the far right. (From [20].)

between connected vertices; the second term is a repulsion term. The equation is given by

$$K_{i(x,y)} = \left[\sum_{j=1}^{n_i} (w_{i,j} \times l_{i,j}^2) \right] + D_{x,y}$$

where $K_{i(x,y)}$ is the energy of a vertex at a specific (x, y) location, n_i is the number of edges connected to vertex i, $w_{i,j}$ is the edge weight between vertex i and the vertex connected by edge j, $l_{i,j}^2$ is the squared distance between vertex i and the vertex at the other end of edge j, and $D_{x,y}$ is a force term proportional to the density of vertices near (x, y).

In our ordinations, the energy equation is minimized iteratively in three phases. The first phase reduces the free energy in the system by expanding vertices toward the general area where they will ultimately belong. The next phase is similar to the quenching step that occurs in simulated annealing algorithms [31], where the nodes take smaller and smaller random jumps to minimize their energy equations. The last phase slowly allows detailed local corrections while avoiding any large, global adjustments.

VxOrd also includes additional improvements. These improvements include barrier jumping, which keeps the algorithm from getting trapped in local minima; a grid-based method for computing $D_{x,y}$, which reduces the computation of the repulsion term from $\Theta(|V|^2)$ to $\Theta(|V|)$; and edge cutting, which encourages exposure of clusters in the final stages of the optimization.

4.3.2.4. Clustering Parameters The analyst has two important controls over the VxOrd algorithm:

1. The number of similarities used for the clustering
2. The degree of edge cutting permitted, where edge cutting refers to removing key edges in order to expose cliques in the graph

The first control concerns how many similarities are passed to the clustering algorithm. Every gene has some correlation with every other gene; however, most of

these are not strong correlations and may only reflect random fluctuations. By using only the top few genes most similar to a particular gene, we obtain two benefits: The algorithm runs much faster and, as the number of similar genes is reduced, the average influence of the other, mostly uncorrelated genes diminishes. This change allows the formation of clusters even when the signals are quite weak. However, when too few genes are used in the process, the clusters break up into tiny random islands containing only two or three very similar genes, so selecting this parameter is an iterative process. One trades off confidence in the reliability of the cluster against refinement into subclusters that may suggest biologically important hypotheses. These clusters are only interpreted as suggestions and require further laboratory and literature work before we assign them any biological importance. However, without accepting this trade-off, it may be impossible to uncover any suggestive structure in the collected data.

The second control tells the algorithm how many edges may be removed so that cliques in the graph, or clusters in the ordination, may be exposed. This parameter must also be balanced for reliability of the clique against actual exposure of the clique.

As an example of the impact of these parameters, consider Figure 4.3. Here we are clustering a set of 126 arrays, each with about 12,000 genes. First consider

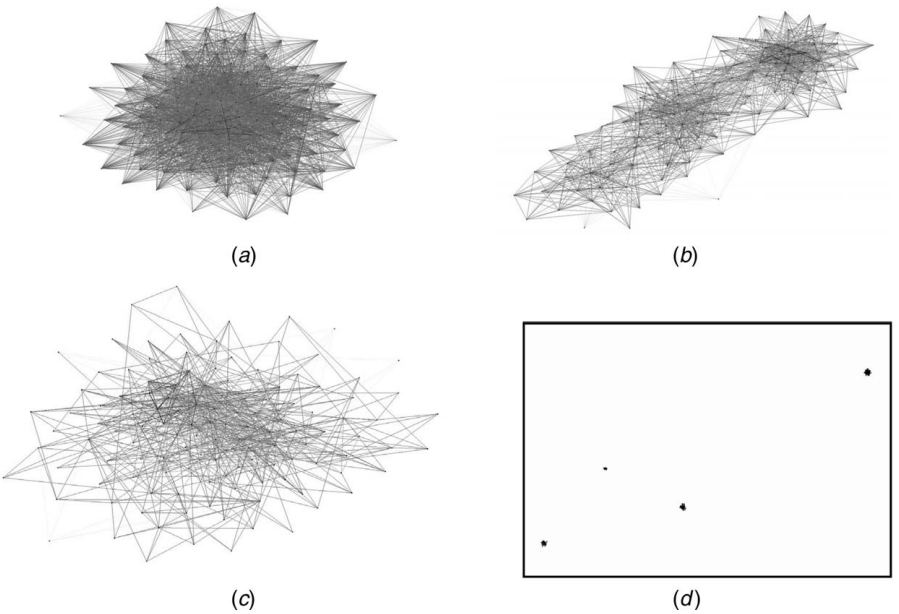


FIGURE 4.3. Effects of using different numbers of similarity links and different parameters for edge cutting. (*a*) Using too many similarity links, in this case 30, and only a single undifferentiated group is formed. (*b*) Using 15 similarity links and the data are no longer completely undifferentiated; some stronger similarities are beginning to force the emergence of structure. (*c*) Using 30 links but with the maximum edge cutting enabled, and clusters are still not apparent. (*d*) Data can be separated the with 15 similarity links and aggressive cutting.

the effect of using too many similarities. Figure 4.3*a* shows the result when 30 similarities per array are used. However, when only the top 15 strongest similarities are used, as in Figure 4.3*b*, three groups begin to be apparent.

When a set of elements has a relatively uniform set of similarities, it can be very difficult to separate them in to subclusters. However, there may be subsets of stronger similarities that would divide the data into clusters if allowed to express their influence in the absence of the other, mostly homogeneous similarities. In other words, we can reveal small cliques of vertices by cutting similarity relationships that have been constraining the vertices to remain an undifferentiated agglomeration. Figure 4.3*c* shows that no cliques are apparent when using 30 similarities per vertex, even with extremely aggressive edge cutting. On the other hand, the suggestive clusters seen in Figure 4.3*b* readily break into more detailed cliques when only 15 similarities per vertex are used and when aggressive edge cutting is enabled, as shown in Figure 4.3*d*.

4.3.2.5. Evaluating Utility and Significance of Clustering Clustering algorithms are designed to find clusters. However, one's initial stance should be that there is no reason to suppose that the clusters found are more than artifacts. We have expended significant effort devising methods for evaluating the clusters produced by VxOrd. These efforts are described in detail in our previous publications, but for completeness we provide a short overview here.

The first, most intuitive approach is to check that gene expressions are correlated within clusters and to investigate the biological meaning of the clusters. One of our first efforts in this direction was an analysis of the Spellman yeast data [32]. In this paper we compared the typical expression histories of the genes in each cluster to assure ourselves that genes in the cluster had, generally, uniform expression patterns and that these patterns were different in the various clusters. Figure 4.4 shows Spellman's yeast cellcycle data clustered with VxOrd, overlaid with expression traces for typical genes in the various clusters. Not only do these traces seem homogeneous within the clusters and different between clusters, but they also have biological significance as the cells move through their replication cycle. Surprisingly, the various states in the cell cycle correspond to a clockwise progression around the roughly circular layout of gene clusters in this figure.

This visual inspection was also recast in the same paper in a statistically more rigorous manner. Although we do not provide the details, we concluded that the VxOrd clusters were not artifacts. A statistical test used in [32] showed that a subset of genes associated with cell cycle phase G1 were collocated with $p < 0.001$ and further that *CLB*6, *RNR*1, *CLN*2, *TOS*4, and *SVS*1 collocated with $p < 0.0001$ for cells exiting from long-term stationary phase.

Another test falling into the category of intuitive verification was performed in [33]. This work tested VxOrd clusters of *C. elegans* genes for clusters enriched in genes known to be involved in various biological processes. Stuart et al. [33] found significant statistical enrichments. These enrichments suggested that other genes in the same clusters could be expected to be involved in the indicated processes. This hypothesis was confirmed with laboratory experiments.

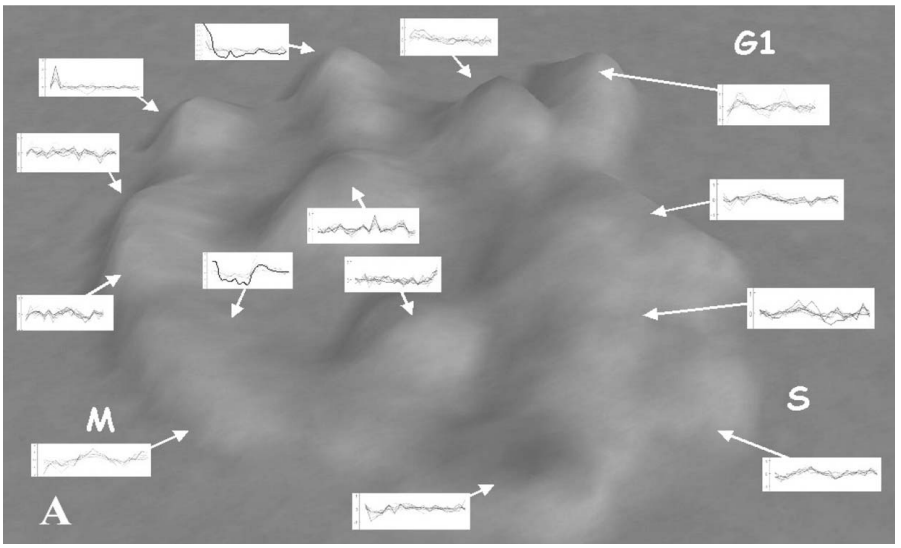


FIGURE 4.4. Cell cycle data with typical expression traces from each cluster. Interestingly, the clusters lay out in a circle corresponding to the temporal cell cycle phases.

Another evaluation method to investigate the clustering algorithm uses exactly the same processing parameters but with randomly permuted versions of the measurements. If the clustering algorithm finds clusters or structures in this randomized data, then the results with the original data should be suspect. The processing methods discussed above have been tested in this way, and randomized data do not exhibit any organized structure; see, for example, the analysis in [29], where the randomized data resulted in a single, symmetric, and otherwise unorganized group of genes, which revealed structure in the data as well as lack of structure in the randomized data. If randomized data show no structure, then the structures in the actual data become more interesting and may possibly be useful.

These methods have been useful in showing that the clusterings are not chance occurrences and have led to scientific insights. However, these approaches have not addressed two other important issues related to clustering: (1) how stable these clusters are with respect to variations in the measurements and (2) how stable they are with respect to different random initializations of the VxOrd clustering algorithm, which has an inherently stochastic nature. We investigated these two issues in [20].

To test the stability of the algorithm with respect to random starting points, we ran 100 reordinations of the Spellman cell cycle data [6], which had about 6000 genes. Each reordination was started with a different seed for the random-number generator. We then visually marked the elements of a cluster in one ordination and looked to see if they were still visually clustered together in the other ordinations. The results of this analysis were generally positive and are shown in

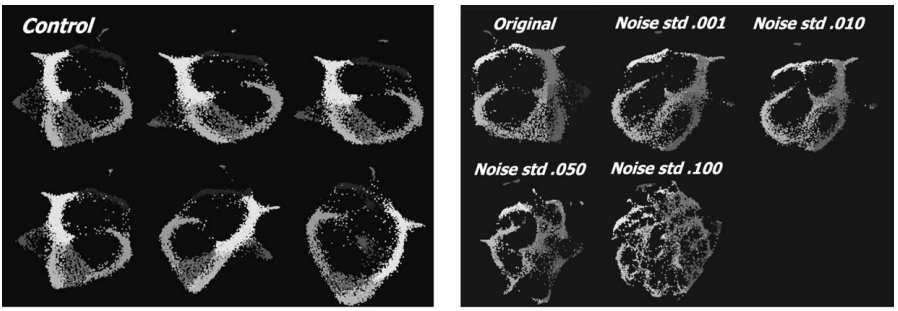


FIGURE 4.5. (*a*) Ordinations with different random starting conditions. (*b*) Effect of increasing edge noise on cluster stability. (From [20].)

Figure 4.5*a*. We also performed a more rigorous neighborhood analysis (see discussion below) with good results.

To determine if small changes or noise in the similarities would give small changes in the ordination results, we ran 80 reordinations where we added noise drawn from a Gaussian distribution with mean zero and standard deviations 0.001, 0.010, 0.050, and 0.100 and recomputed the ordinations. These different ordinations were also compared visually and statistically. These results generally showed that our the clusterings held together remarkably well, even when a large fraction of noise was added. The visual results are shown in Figure 4.5*b*.

4.3.2.6. Finding Most Representative Clustering Each randomly restarted ordination by VxOrd represents a sample from a distribution of possible ordinations arising from a particular similarity file. From this perspective, one might want to identify the *best ordination*, which is particularly hard because it is an extreme and further because the concept of *best cluster* or *best ordination* is not particularly well defined. However, the idea of a *most representative ordination* or *most central ordination* (MCO) can be defined with respect to the sample of observed randomly restarted ordinations. In this case, two ordinations are compared by neighborhood analysis to create a single measure of overall similarity between the two ordinations. With this method for comparing two ordinations, one can make all possible comparisons of the available randomly restarted ordinations and then select the ordination that is, on average, most like all the remaining reordinations. This idea of centrality of the distribution of ordinations might be further extended to the second moment to compute some measure of dispersion, which perhaps could eventually be extended to allow some sort of hypothesis testing about these ordinations. However, we have only investigated the centrality issue.

As an example, we used massively parallel computers to calculate hundreds or in some cases thousands of reclustered ordinations with different seeds for the random-number generator. We compared pairs of ordinations by counting, for

every gene, the number of common neighbors found in each ordination. Typically, we looked in a region containing the 20 nearest neighbors around each gene, in which case one could find a minimum of 0 common neighbors in the two ordinations or a maximum of 20 common neighbors. By summing across every one of the genes, an overall comparison of similarity of the two ordinations was computed. We computed all pairwise comparisons between the randomly restarted ordinations and used those comparisons to find the ordination that had the largest count of similar neighbors. Note that this corresponds to finding the ordination whose comparison with all the others has minimal entropy and in a general sense represents the MCO of the entire set. Although not shown here, plots of the entropies, intersection counts, and cross plots of entropy showed that there were central ordinations, even for a data set that we found very difficult to break into stable clusters [34].

It is possible to use these comparison counts (or entropies) *as a derived similarity measure* in order to compute another round of ordinations. For example, given that 200 random reordinations have been computed, one can compute the total number of times gene g_j turns up in the neighborhood of gene g_k in the available 200 ordinations. This count, or the average number of times the two genes are near each other, will be high when the two genes are generally collocated (which should be a reflection of similar expression profiles for g_j and g_k). The clusters from this recursive use of the ordination algorithm are generally smaller, much tighter, and generally more stable with respect to random starting conditions than any single ordination. In fact, we found that most single ordinations were more similar to the MCO when using the derived similarity than when using Pearson's R.

We typically use all of these methods (computing the MCO from among about 100 to 200 random reordinations and computing neighborhood set sizes ranging from 10 to 30 by steps of 5) during exploratory data analysis in order to develop intuition about the data.

As an interesting aside, the process of comparing pairs of ordinations results in similarity values between every ordination. These similarities can be used to create *clusters of the clusterings*! Figure 4.6 shows an example where we found that the random reclusterings seemed to fall into two different attractor basins, which may be interpreted as a sign that there were two different but perhaps equally valuable ways to look at the data and that no single cluster was able to represent both of these views.

4.3.3. Enhancing VxOrd

In addition to developing the MCO algorithm, which is layered on top of the VxOrd algorithm, we put some effort into enhancing VxOrd itself. For clarity, we will hereby denote the enhanced version by VxOrd 2.0 and the original verision by VxOrd 1.5. The original motivation for developing VxOrd 2.0 was to cluster larger data sets, although we also found the algorithm to be more stable and more accurate on certain data sets.

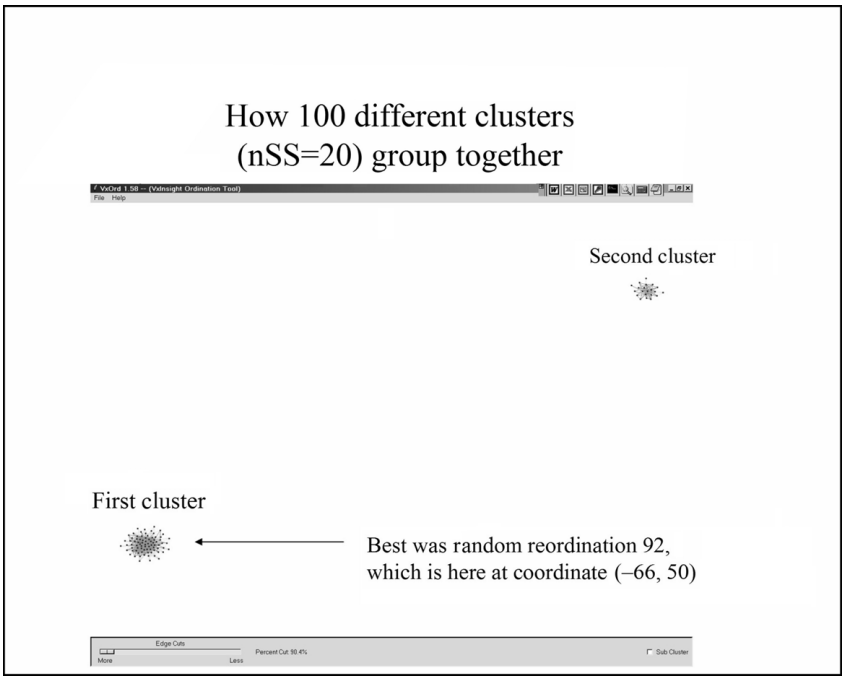


FIGURE 4.6. Process of comparing individual clusters results in similarity values between each ordination. These similarities can be used to create a cluster of clusters. In this case, there seem to be two attractors, suggesting the data may have two useful visualizations.

4.3.3.1. Adding Graph Coarsening to VxOrd VxOrd 2.0 is based on graph coarsening, which has previously been applied to other graph layout algorithms in order to draw larger graphs more efficiently [35, 36]. Graph coarsening, also known as multilevel mesh refinement, works by replacing the original graph with a smaller but still representative graph. In the context of graph layout, we draw the smaller graph and use this initial layout as a starting point for drawing the original graph. In fact, we use an iterative approach which typically provides a succession of smaller graphs and hence a succession of graph layouts.

Suppose the initial graph is denoted by G_0. A coarsened version G_1 of G_0 is obtained by randomly visiting and subsequently merging nodes as follows:

1. Pick a node v_i at random.
2. Choose a neighbor v_j of v_i such that the edge e_{ij} has maximal weight. If no such v_j exists or v_j has been previously visited/merged, go to step 4.
3. Merge v_j into v_i by adding edge weights of common neighbors or creating new edges if there are no common neighbors.
4. Repeat until all nodes have been visited/merged.

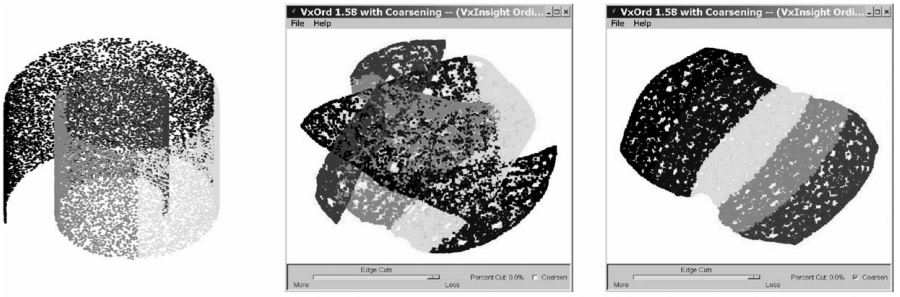
This algorithm is fast and has been observed [37] to produce much smaller but still representative versions of the original graph.

Once the initial graph G_0 has been coarsened, we can draw the smaller version G_1 using VxOrd 1.5 and then reverse our coarsening to obtain a layout of G_0. In fact, we repeatedly apply the coarsening algorithm to get an even smaller graph. The algorithm that we use is as follows:

1. Apply the coarsening algorithm until we obtain a suitably small graph (say 50 nodes) or until the algorithm fails to make any more progress. We obtain a sequence of graphs $G_0, G_1, \ldots, G_m$ from this process.
2. Draw G_m using VxOrd.
3. Refine G_m to obtain G_{m-1}. Place the additional nodes obtained by this refinement in the same positions as their counterparts (merged nodes in G_m) and adjust with VxOrd.
4. Repeat step 3 using $G_{m-2}, \ldots, G_0$.

This algorithm requires additional adjustments in terms of the grid-based density calculations and the various parameters involved in the simulated annealing. With proper adjustments, however, we obtain better accuracy and stability with this algorithm (VxOrd 2.0) than with the previous version (VxOrd 1.5), as will be demonstrated in the following section.

4.3.3.2. Benchmarking VxOrd 2.0 We first benchmarked VxOrd 2.0 on the so-called swiss roll data set. Although this is not microarray data, it provides a useful example of the essential difference between VxOrd 2.0 and VxOrd 1.5. This data set was used in [38, 39] to test two different nonlinear dimensionality reduction algorithms. The data set is provided as points in three dimensions, which give a spiral embedding of a two-dimensional ribbon (see Fig. 4.7*a*). To test VxOrd, we considered each point to be a node in an abstract graph, and



(*a*) (*b*) (*c*)

FIGURE 4.7. The swiss roll is a two-dimensional manifold embedded nonlinearly in three dimensions: (*a*) actual data set; layouts of associated graph using (*b*) VxOrd 1.5 and (*c*) 2.0 VxOrd.

we connected each node to its 20 nearest neighbors. In principle, VxOrd should draw the graph as a two-dimensional ribbon.

The swiss roll is a useful benchmark because it is very easy to visualize but hard enough so that it will confound (at a minimum) any linear algorithm (such as principal–component analysis). It also confounded the original VxOrd 1.5, as can be seen in Figure 4.7*b*. Looking closely at Figure 4.7*b*, we can see that VxOrd 1.5 did well on a local scale (i.e., the colors are together and in the correct order) but failed on a global scale (the ribbon is twisted). Once graph coarsening was added to VxOrd, however, the global structure was also ascertained correctly, as shown in Figure 4.7*c*. In our subsequent analysis we found a similar story: VxOrd 2.0 does well on large data sets on a global scale but otherwise does not improve VxOrd 1.5.

For our next benchmark, we again used the swiss roll data set, an adult leukemia data set provided by the University of New Mexico Cancer Research and Treatment Center, and the yeast microarray data set [6] used previously. In order to test the stability of our modification to VxOrd, we performed multiple runs of both VxOrd 1.5 and VxOrd 2.0 with different random starting conditions. We then compared the ordinations using a similarity metric obtained as a modification of a metric discussed in [40].

Our similarity metric is a function $s_e(U, V)$, where U, V are two VxOrd layouts of the same m-node data set $\mathbf{x}_1, \ldots, \mathbf{x}_n$. The metric is computed by first constructing neighborhood incidence matrices $N_{U,e}$ and $N_{V,e}$, where $N_{*,e}$ is an $n \times n$ matrix $N_{*,*} = (n_{ij})$, with

$$n_{ij} = \begin{cases} 1 & \text{if } \|\mathbf{x}_i - \mathbf{x}_j\| < e \\ 0 & \text{otherwise} \end{cases}$$

Now

$$s_e(U, V) = \frac{N_{U,e} \cdot N_{V,e}}{\|N_{U,e}\| \|N_{V,e}\|}$$

where $N_{U,e} \cdot N_{V,e}$ is the dot product of $N_{U,e}$ and $N_{V,e}$ when both matrices are considered to be vectors of length n^2. Finally, we note that in order to make sure we can form reasonable e neighborhoods of a given node $\mathbf{x}_i$, we first scale both U and V to lie in the area $[-1, 1] \times [-1, 1]$.

To see how we can use this metric to assess the stability of VxOrd 2.0, we first revisit the swiss roll data set. In the top row of Figure 4.8, we show an all-versus-all similarity matrix for 10 different random runs of both VxOrd 1.5 and VxOrd 2.0. This figure confirms the results from Figure 4.7 and shows that the metric is valid. In particular, we see that VxOrd 2.0 arrives at a consistent solution (indicated by higher similarities) while VxOrd 1.5 is less consistent.

Next we computed the same similarity matrices for the adult leukemia data set and for the yeast time-series data, as shown in the second and third rows of

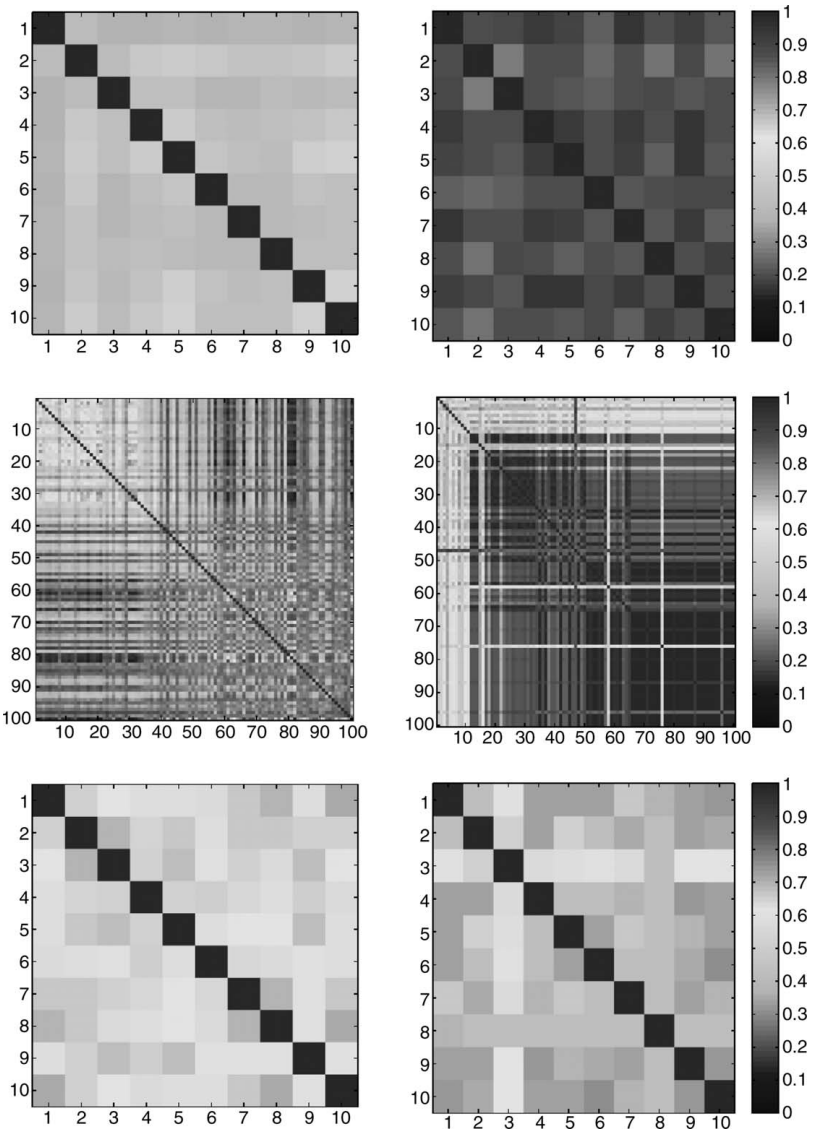


FIGURE 4.8. Comparison of the similarity matrices for random runs of VxOrd 1.5 and 2.0 using swiss roll, adult leukemia, and yeast data sets. On the left we show the runs produces by VxOrd 1.5, and on the right we see the runs produced by VxOrd 2.0. From top to bottom we have the swiss roll with 10 runs, the adult leukemia data set with 100 runs (10 runs for 10 different edge cutting parameters—more aggressive up and left), and the yeast data set with 10 runs. The color scale is shown on the right and is the same for all images. If we look at this figure as a whole, we see that the right-hand column has more red than the left-hand column and hence that VxOrd 2.0 (right column) is generally more stable than VxOrd 1.5 (left column).

TABLE 4.1 Average Values (Excluding the Diagonal) of the Similarity Matrices Shown in Figure 17

	Swiss Roll	AML	Yeast
VxOrd 1.5	0.43	0.33	0.62
VxOrd 2.0	0.87	0.80	0.69

Figure 4.8. We arrived at similar results in each case. We also computed the average similarity across the matrices (excluding the diagonal) for the different cases, shown in Table 4.1.

In the case of the adult leukemia data set, we also experimented with the edge cutting feature of VxOrd. In particular, we computed 10 random runs for each of the 10 most aggressive edge cuts. We found that even though VxOrd 2.0 was more consistent overall, it still was not consistent with the most aggressive cut.

The yeast data set was larger (6147 nodes compared to 170 nodes in the adult leukemia data set) and the results of both VxOrd 1.5 and 2.0 were fairly consistent. In particular this suggests VxOrd 1.5 still does well on larger data sets without an inherently gridlike structure (as in the swiss roll).

4.4. SUPERVISED METHODS

Clustering a microarray data set is typically only the first step in an analysis. While we have presented tools and techniques to help assure a reasonable and stable ordination, we have not yet discussed the most important part of the analysis: the steps necessary to determine the actual biological meaning of the proposed clusters. While we do not typically address the actual biology in a given analysis, we often provide the appropriate tools for such analysis. These tools must be both informative and accessible to the average biologist.

4.4.1. Using VxInsight to Analyze Microarray Data

As stated earlier, most of our work is built upon a database with a visual interface known as VxInsight [1]. VxInsight was originally developed for text mining but has been extended for the purpose of microarray analysis. Once an ordination has been obtained (using the methods described previously), the ordination is imported into VxInsight, along with any annotation or clinical information.

VxInsight uses a terrain metaphor for the data, which helps the analyst find and memorize many large-scale features in the data. The user can navigate through the terrain by zooming in and out, labeling peaks, displaying the underlying graph structure, and making queries into the annotation or clinical data. In short, VxInsight provides an intuitive visual interface that allows the user to quickly investigate and propose any number of hypotheses. Details about and applications of VxInsight can be found in [1, 20, 29, 32, 33].

4.4.1.1. Typical Steps in Analysis Using VxInsight VxInsight is very useful for an initial sanity check of a data set. We will typically cluster the arrays to look for mistakes in the scanning or data processing which might have duplicated an array. A duplication will often be apparent in the experiment because the pair of duplicated arrays will cluster directly on top of each other and will typically be far from the other clusters. We have discovered that many data sets cluster more by the day the samples were processed, or even by the technician processing the samples, than because of biologically relevant factors. Further investigation is needed, for example, if almost 100% of a particular processing set clusters by itself. In one case we found a very stable ordination consisting of two groups. After much confusion we discovered that the groups were divided by experimental batch and that one of the groups consisted of patients whose samples contained only dead or dying cells (perhaps due to bad reagents or problems with the freezing process). When the experiments were redone, the original clusters dissolved into more biologically meaningful clusters.

One can often see the effect of confounding experimental conditions using this same method. For example, if a set of arrays is processed by the date they were collected and the date corresponds to separate individual studies, then the processing set (date) will be confounded with the study number. Well-designed studies control such confounding by randomizing the processing order or by carefully balancing the processing order. However, it is always wise to use these exploratory analysis methods to ensure that your main effect has not, somehow, been confounded.

A more interesting phase of analysis begins after obviously bad data have been culled and the remaining data have been reclustered. The data may be clustered in either of two ways. In one approach, the genes are clustered in an effort to identify possible functions for unstudied genes. See, for example, [29, 32].

In the other approach, which is often seen in clinical studies, we cluster the arrays (the patients) by their overall expression patterns. These clusters will hopefully correspond to some important differentiating characteristic, say, something in the clinical information. As the analysis proceeds, various hypotheses are created and tested. VxInsight has plotting features that are helpful here, including a browser page with various plots as well as links to external, Web-based information.

Although useful information can be gleaned by simply labeling different peaks in VxInsight, a more systematic method is even more informative. At the highest level, one may wish to select two clusters of arrays and ask: *Which genes have significantly differential expressions between these two clusters*? Given any method for identifying such genes, it is useful to display them within the context of the cluster-by-genes map. Sometimes the most strongly differentiating genes for the clusters of arrays may not have been previously studied. In this case, it can be very useful to find known genes that cluster around these unstudied genes using the cluster-by-genes map.

This process is illustrated in Figure 4.9, which shows the original table of array data, clustered both by arrays and by genes. The lower map represents the result after clustering by arrays and shows two highlighted clusters (colored white and green, respectively). The genes with strongly differential expressions between the groups

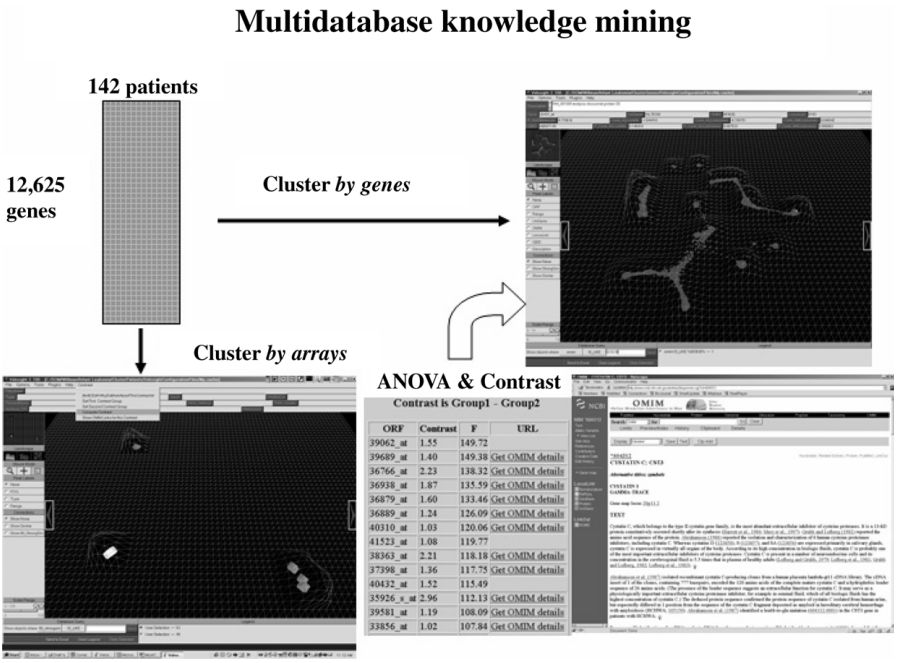


FIGURE 4.9. Array of expression data for large number of experiments shown clustered by genes and by arrays. A list of genes with different expressions between two groups of arrays is shown. This list includes a short annotation and links to more extensive, Web-based annotations.

of arrays are shown to the right of this map. Note that the list is sorted by a statistical score and also contains links to the available Web-based annotations. A curved arrow in the figure suggests the path between the gene list and the cluster-by-genes image. That connection is implemented with sockets and forms the basis of a more general analysis tool, which allows an arbitrary gene list to be sent from the analysis of the arrays to the analysis of the genes.

4.4.1.2. Generating Gene Lists There are many methods for generating gene lists or finding genes which are expressed differently in different groups. As stated in the introduction, this process is known as supervised learning, since we are using known groups to learn about (and make predictions about) our data set. Finding gene lists in particular is known as *feature* or *variable selection*, where the features in this case are genes.

There are a wide variety of methods for feature selection, and we do not provide here an extensive survey of this area. We do, however, mention some of the methods developed specifically to select genes for microarray analysis. The method in [7] was one of the first gene selection methods proposed, the method in [41] applied feature selection for support vector machines to microarray data, and [42] discusses a variety of feature selection methods applied to microarray data.

For our purposes, we use a simple statistical method for gene selection. A gene-by-gene comparison between two groups (1 and 2) can be accomplished with a simple *t*-test. However, we wanted to eventually support comparisons between more than two groups at a time, so we actually used analysis of variance (ANOVA). This processing results in an *F*-statistic for each gene. The list of genes is sorted to have decreasing *F*-scores, and then the top 0.01% of the entire list are reported in a Web page format, with links to the associated OMIM pages. The OMIM pages are then examined manually to hypothesize biological differences between the clusters.

4.4.1.3. Gene List Stability An analysis using the gene list feature of VxInsight typically progresses as follows. First, a question is posed within the VxInsight framework and a statistical contrast is computed for that question. The gene list is initially examined to see if any genes are recognized by their short descriptions, which, if available, are included with the genes. The plots are examined, and the OMIM annotations are read. If the gene appears to be important, the literature links and other relevant National Center for Biotechnology Information (NCBI) resources are studied. This analysis step is very labor and knowledge intensive; it requires the bulk of the time needed to make an analysis. As such, it is very important to not waste time following leads that are only weakly indicated. That is to say, before one invests a great deal of time studying the top genes on a list, it is important to know that those highly ranked genes would likely remain highly ranked if the experiment could be repeated or if slight variations or perturbations of the data had occurred.

The critical issue about any ordered list of genes is whether we can have any confidence that this list reflects a nonrandom trend. To be very concrete, suppose that My Favorite Gene (MFG) is at the top of the list in our ANOVA calculations, that is, MFG had the largest observed *F*-statistic from the ANOVA. What can we conclude about the observed ranking for MFG? Certainly, a naive use of the *F*-statistic has no support because we tested, say, 10,000 genes and found the very largest statistic from all of those tests. So, an *F*-value for $p = 0.001$ would likely be exceeded about 10 times in our process, even if all the numbers were random. Hence, the reported *F*-statistic should only be considered to be an index for ordering the values.

However, if we could repeat the experiment and if MFG was truly important, it should, on average, sort into order somewhere near the top of the gene list. We cannot actually repeat the experiment, but we can treat the values collected for a gene as a representative empirical distribution. If we accept that this distribution is representative, then we can draw a new set of values for each of the two groups by resampling the corresponding empirical distributions repeatedly (with replacement), as shown in Figure 4.10. This process is due to Efron and is known as bootstrapping [43].

Now consider Figure 4.11, where we resample for every gene across all of the arrays in the two groups to create, say, 100 new experiments. These experiments are then processed exactly the same way as the original measurements were processed. We compute ANOVA for each gene and then sort the genes by their *F*-value. As we construct these bootstrapped experiments, we accumulate the

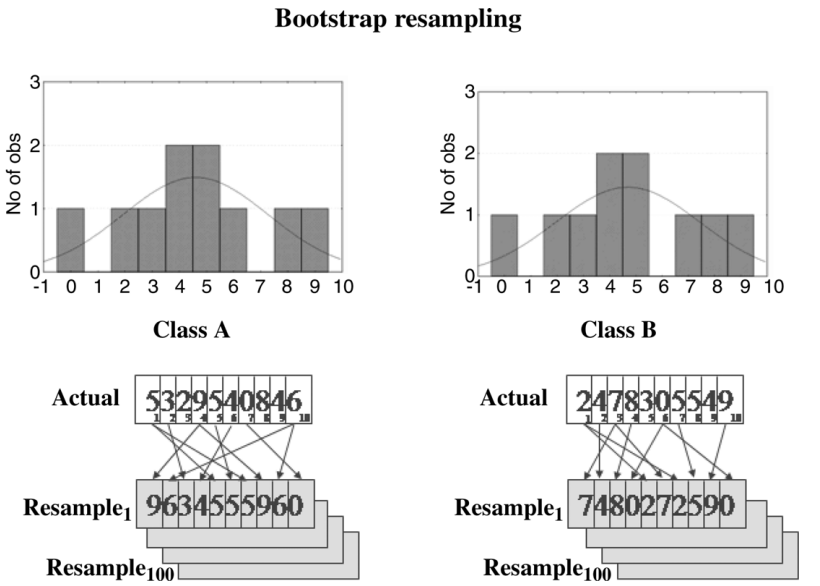


FIGURE 4.10. A bootstrap method uses the actual measured data as estimates for the underlying distribution from which the data were drawn. One can then sample from that estimated underlying distribution by resampling (with replacement) from the actual measurements.

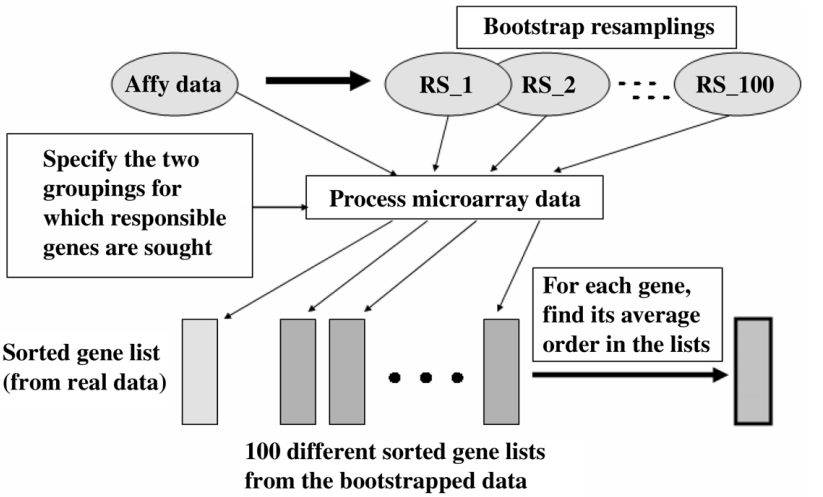


FIGURE 4.11. Actual data are processed to create the gene list, shown at the bottom left. The actual data are then resampled to create several bootstrapped data sets. These data sets are processed exactly the same way as the real data to produce a set of gene lists. The average order and the confidence bands for that order can be estimated from this ensemble of bootstrapped gene lists.

distribution of the location in the list where each gene is likely to appear. Using these bootstrap results one can determine, for each gene, its average order in the gene lists. Although the distributions for such order statistics are known, they are complex. On the other hand, the bootstrapped distributions are easily accumulated, and they are acceptable for our needs.

In addition to the average ranking, we count the 95% confidence bands for each gene's ranking as estimated by the bootstraps. We report both the upper 95% confidence band and the centered 95% confidence interval for each of the genes. The lower limit of this upper 95% confidence band (LLUCB) is recorded for later use (note that 5% of the time we would observe a ranking below LLUCB by random chance, even when our hypothesis is false, given the two empirical distributions).

Next, we investigate the p-values for the observed rankings of these genes under the null hypothesis, H_0, that there is no difference in gene expression between the two groups (1 and 2). In this case (when H_0 is in fact true), *the best empirical distribution would be the unordered combination of all the values without respect to their group labels*. To test this hypothesis, we create, for example, 10,000 synthetic distributions by bootstrapping from this combined empirical distribution and process them exactly as we did the original data.

We are interested in what fraction of the time we observed a particular gene ranking higher in the bootstrapped results than the appropriate critical value. There are several reasonable choices for this critical value. We could use the actual observed ranking or the average ranking from the bootstraps under the assumption that H_0 was false. Instead, we take an even more conservative stance and choose a critical value using a power analysis to control our chance of a type II error. We set $\beta = 0.05$, or 5%.

If H_0 were false (i.e., if the groups do have different means), then the earlier bootstrapping experiments suggest that one might randomly observe a ranking as low as LLUCB about 5% of the time. Hence, we examine the later bootstrap experiments (under H_0 assumed true and thus no group differences) and find the fraction of the times that we observe a ranking at or above LLUCB. This value is reported, gene by gene, as the p-value for the actual rankings. In essence, we are saying that if H_0 is true, then by random chance we would have seen the gene ranking above LLUCB with probability p. *As LLUCB is much lower than the actual ranking, this p-value is very conservative for the actual ranking*.

To investigate the meaning of the actual F-statistics used to index these gene lists, we computed another bootstrap experiment. We were interested in the effect of scaling the original expression values by their Savage-scored order statistics. As previously discussed, this scoring is felt to be more robust than taking logarithms. However, we were concerned that this might influence our p-values, so we developed a code to estimate the expected F-statistic for the mth ranked gene in a gene list from two groups (1 and 2) respectively having j and k arrays. This code computes a large bootstrap after randomizing the Savage scores within each of the $j + k$ arrays. The code then computes the ANOVA for each gene and eventually sorts the resulting genes into decreasing order by F-statistics. The final result is a p-value (by bootstrap) for the two groups with the specific number of arrays. This

computation is rather intensive and should either be fully tabulated or run only as needed for genes uncovered by the earlier methods. We have not run extensive simulations of this code against the p-values or the list order distributions, but the limited checks did suggest that genes which always ranked near the top of the differentiating gene lists do have rare F-statistics based on the Savage-scored orders relative to the expected random distributions (data not shown).

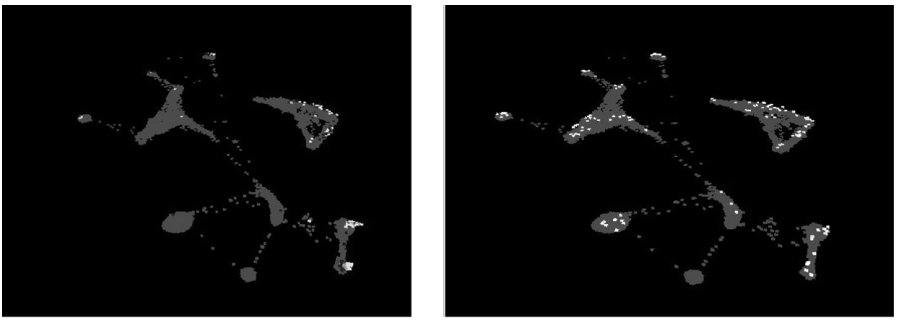
4.4.1.4. Comparing Gene Lists As mentioned previously, the ANOVA plus bootstrap approach described above is only one way to find genes which may have important roles with respect to particular biological questions. Our collaborators, for example, have used support vector machine recursive feature elimination (SVM RFE) [41], a Bayesian network approach using a feature selection method known as TNoM [44], and a technique based on fuzzy-set theory as well as more classical techniques, such as discriminant analysis. By using several of these methods, one might hope to find a consensus list of genes. Our experience has shown that this is possible. While the lists from different methods are usually not exactly the same, they often have large intersections. However, the simultaneous comparison of multiple lists has been a difficult problem.

We have developed a number of methods which have helped us understand that the lists may be different in the details but still very similar biologically. This makes sense considering that different methods might identify different but closely related elements of regulation or interaction networks. In that case, the methods suggest the importance of the network and the particular region in that network, even though they do not identify exactly the same elements. This relatedness suggests something similar to the kind of "guilty-by-association" method that has been used to impute gene functions for unstudied genes that cluster near genes with known function, as in [29]. Indeed, something similar can be used to evaluate the similarity of multiple gene lists.

Figure 4.12*a* shows a VxInsight screen for clusters of genes. Highlighted across the clusters are genes identified by different methods (shown in different colors). In this particular case, one can see that the various methods do identify genes that are generally collocated, which suggests that gene regulations and interacting networks probably do play a strong role with respect to the question under consideration. Here, for example, the question was, "which genes are differentially expressed in two types of cancers [acute lymphoblastic/myeloid leukemia (ALL/AML)]?".

However, multiple methods do not always produce such strong agreement, as shown in Figure 4.12*b*. In this case the question was, "which genes are predictive for patients who will ultimately have successful treatment outcomes (remission/failure)?" Unfortunately, this question had no clear answer. Interestingly, the ANOVA-plus-bootstrap method suggests a very stable set of genes for the first question, while the list for the second question is not stable and has confidence bands spanning hundreds of rank-order positions (data not shown).

Finally, we have discovered that when two methods produce similar gene lists, the coherence may be due to the underlying similarity of the methods more than to any true biological significance. We discovered this fact using a visualization of the gene lists using principal-component analysis (PCA), a common technique used for



(*a*) (*b*)

FIGURE 4.12. (*a*) General collocation of genes identified by different algorithms (shown with different colors). This collocations suggests that the different methods are in reasonable agreement. (*b*) Genes selected by each method are widely separated and show no coherence, suggesting that there is a lack of consensus among the methods.

dimensionality reduction that involves projections of the original data onto the *principal components*. These components are ordered according to the amount of data captured by each component. The first component is the most informative, the second is the next most informative, and so on. Further information on PCA can be found in [45, 46]. An example of PCA applied to microarray data can be found in [8].

In our PCA-based visualization of multiple gene lists, each gene is considered to be a point in patient space, where each dimension corresponds to a different patient. Since, in this case, there were $\sim$12,000 genes and 126 patients, the spatial representation had 12,000 points (samples) in a 126-dimensional space. Of the 12,000 genes we only considered about 600 that occurred in the different gene lists, reducing our problem to 600 genes in 126 dimensions. Furthermore, because we were mainly interested in how the genes compared as discriminators, and not how their actual expression levels compared, we projected the genes onto the 126-dimensional unit sphere in patient space, as suggested in Figure 4.13*a*. Geometrically, this corresponds to comparing the directions of the genes in the various gene lists as opposed to their magnitudes.

In order to understand this visualization, is it useful to imagine a sphere with a plane passing through the origin. The sphere corresponds to the unit sphere (the sphere with radius 1 centered at the origin) in the patient space and the plane corresponds to the plane determined by the first two principal components. The first principal component points in the radial direction of the sphere and the second principal component is tangential to the sphere at the sphere's intersection with the first principal component. The vector representing a particular gene and it will intersect the unit sphere, and it will be near the equator of the sphere (unit circle) if it lies in the plane of the first two principal components. To the extent that the gene lies above or below the plane of the first two principal components, the projection of the intersection back down onto the plane will lie further inside the equator. The distribution of these projections onto the principal-component plane suggests how a given method of gene selection identifies important genes.

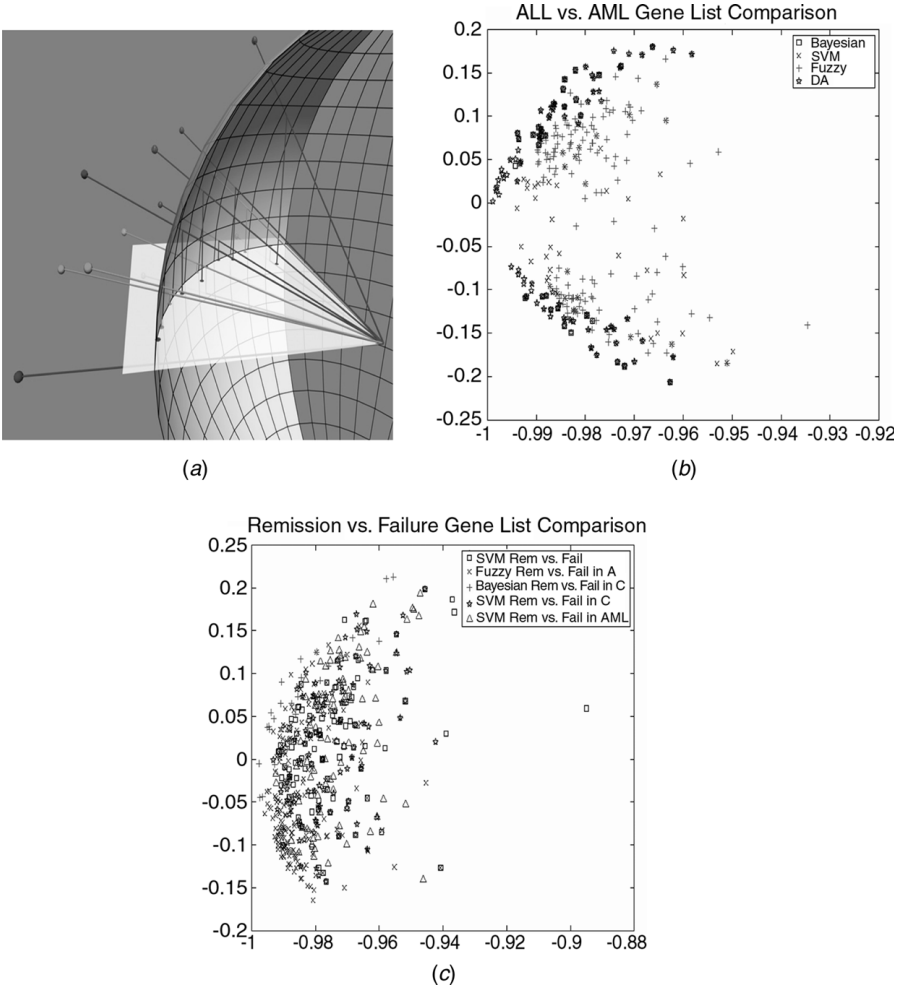


FIGURE 4.13. (*a*) A few genes from three different methods are shown intersecting the unit sphere, along with the projections of those intersections down onto the plane of the first two principal components. Note that genes near that plane will have projections that fall close to the arc of the sphere, while those above or below the plane will have intersections that fall well within the equator of the sphere. (*b*) The ALL-vs-AML gene list comparison. The gene lists that characterize ALL vs. AML are shown, with a different color for each of the methods used to obtain them. In distinguishing ALL from AML we found that most of the genes in the list were colocalized in our representative visualization. (*c*) Gene lists that characterize remission vs. failure are shown, with a different color for each of the methods used to obtain them. It can be seen in this figure that distinguishing remission from failure is a difficult task.

For instance, discriminant analysis and the ANOVA methods are much more similar to each other than to the Bayesian network approach. If we use PCA (see [46] for an introduction to PCA), we see further that many methods will be heavily influenced by differences in the first few principal components of the gene expression data. On the other hand, methods such as SVM RFE [41] are able to examine the simultaneous efficacy of groups of genes, some of which, individually, may not be discriminatory in the first or second principal component. One way to understand these differences is by considering where selected genes project onto the plane of the first two principal components: see Figure 4.13*a*, which schematically represents a few genes from three methods, identified by different colors.

It is evident from Figure 4.13*b* that the gene lists selected for the ALL/AML problem are related. Unfortunately, it is equally obvious that the gene lists selected for the remission/failure problem are unrelated, as shown using the same analysis in Figure 4.13*c*.

When distinguishing ALL from AML, we found that most of the lists were colocalized in our representative visualization (see Figs. 4.12*a* and 4.13*b*). When distinguishing remission from failure, on the other hand, we could not arrive at a satisfactory conclusion (see, Figs. 4.12*b* and 4.13*c*), which is also consistent with the results from ANOVA plus bootstrapping (data not shown).

4.4.2. Unifying Gene Lists

Although it is useful to compare gene lists, the task of sifting through five or six such lists can be very time consuming for the biologist. For this reason, we also developed a quick-and-easy way to combine multiple lists into a single master list.

In order to collate and compare the different gene lists we used a weighted voting scheme. In this scheme, we consider genes to be candidates and gene lists to be votes. In other words, each method suggests, in order of preference, which genes should be elected. Our method for combining the gene lists ranks the candidate genes according to the geometric mean of the voting order in each list, where

TABLE 4.2 A Simple Combination of Many Gene Lists

Rank	Geo Mean	SVM	Stepwise	ROC	ANOVA	TNoM
1	1	1	1	1	1	1
2	6.17	4	2	6	6	
3	6.49	3		2	2	
4	8.57	2		8	3	
5	10.14	5		3	10	23
6	10.24	6		5	11	11
7	10.9	10		4	4	
8	13.34	8		11	5	
9	13.66	11	3	15		
10	13.91	12	4	14	25	

Note: The overall rank was obtained by using the geometric mean of the ranks provided in columns 3–7. An empty cell in the array indicates a value of 31.

each method is allowed only 30 votes (the length of our shortest list) and all other genes are given a vote of 31. An example is shown in Table 4.2.

4.5. CONCLUSION

At this point in the analysis it may seem that the biology has dissolved into a sea of numbers and statistical methods. However, these methods are our only guideposts when we begin reading the known information about the indicated genes. Without them we could easily waste very valuable time and people in the study of genes which are only weakly, if at all, related to the central questions of the research. Guided by these methods, we can approach the literature with greater confidence and are much more likely to see the important biology reemerge in the gene annotations and the cited literature.

However, even after these statistical filters, this literature is vast and is not organized to make our searching particularly easy. We have come to recognize that this step (where very knowledgeable scientists must read extensively) is the critical, rate-limiting step for our research. As a result, we (and many others) have begun work with the natural language processing (NLP) community to build tools that find, summarize, and reorder important parts of the available online literature to make that reading process simpler and more focused toward our research needs. Although we do not discuss such work here, a demonstration of a preliminary automatic Gene List Exploration Environment (GLEE) can be found at http://aiaia.nmsu.edu/. See also [34]

Regardless, gene expression studies are providing new insights into molecular mechanism and hold the promise of deeper biological understanding. However, the speed at which groups of genes generated by microarray analysis can be put together in pathways is one of the limiting steps in the translation of these discoveries to applications. Mistakes and dead ends due to faulty microarray analysis tools are a particularly frustrating way to slow this analysis.

The methods presented here are potentially useful in uncovering groups of genes that serve to fingerprint biologically important subtypes; further aiding biological discoveries; and refining diagnosis and improving assessment of prognosis. To provide greater confidence in our tools, we have also benchmarked our methods extensively for reliability. In fact, we believe that *both* of these factors (usefulness and reliability) are equally important, particularly for the analysis of microarray data.

ACKNOWLEDGMENTS

The authors of this chapter gratefully acknowledge the immense contributions of our collaborators, without which this work could not have been accomplished. Here we reported on the methods and techniques developed at Sandia National Laboratories. However, much of the work discussed sprang from our collaborators; we thank them for their generous time and patience in teaching us enough biology to be helpful and for including us in their laboratories. While developing these methods we have worked directly with and have had fruitful discussions with dozens of researchers from these laboratories and have benefited

by many generous introductions to other researchers. We owe each of these people our thanks for their critiques of our work and encouragements to continue. Our collaborations with the University of New Mexico have been very fruitful. The informatics work reported here stems directly from close interactions with Vickie Peck, who opened the world of genomics to us. Certainly, without the continuing encouragements and contributions of Maggie Werner-Washburne we would not have created useful tools or have been able to explain them to life scientists. One of the most important of these was Stuart Kim from Stanford University, who was able to see through the primitive nature of our early efforts and recognize what they could become. Our collaboration with Stuart stretched and improved all of our tools and led to an important, joint publication that continues to have impact. The microarray work with Cheryl Willman's laboratory at the University of New Mexico Cancer Center drove the development of many of the statistical techniques presented here. Cheryl included us in her weekly laboratory meetings, which must surely have been made more tedious by our presence and the continuing need to teach us the rudiments of leukemia biology. Each of these groups is large and we have learned something from every one of you, for which we say thank you. We would especially like to acknowledge the help and collaborations of Moni Kiraly, Jim Lund, Kyle Duke, Min Jiang, Joshua M. Stuart, and Andreas Eizinger from the Kim Laboratory and, of course, Edwina Fuge, Jose Weber, Juanita Martinez, Anthony Aragon, and Angela Rodriguez from the Werner-Washburne Laboratory. From the Willman Laboratory, we must certainly acknowledge Susan Atlas, Paul Helman, Robert Veroff, Erik Andries, Kerem Ar, Yuexian Xu, Huining Kang, Xuefei Wang, Fred Schultz, Maurice Murphy, and particularly Mónica Mosquera-Caro and Jeffrey Potter, who have been immensely helpful to our research. We would particularly like to thank Jon C. Helton for suggesting the use of Savage scoring as a means to normalize microarray data. If we have unfortunately omitted someone to whom we owe our thanks and gratitude, please accept our apologies and recognize that we do value your help. As always, any misrepresentation or error with respect to our collaborators' work is purely our own fault. Finally, we would like to thank the W. M. Keck foundation for funding the W. M. Keck Genomics Center at the University of New Mexico, which was particularly important in our work. This work was supported by grants from the National Science Foundation, (NSF) (MCB-0092374) to M.W.W., an NSF Minority Post-doctoral fellowship to M.J.M, and U.S. Department of Agriculture (99-38422-8034) to A. D. A. Development of the algorithms described here was funded in part by the W. M. Keck foundation and a Laboratory Directed Research and Development program from Sandia National Laboratories. In addition, a portion of this work was funded by the U.S. Department of Energy's Genomics: GTL program (www.doegenomestolife.org) under project "Carbon Sequestration in *Synechococcus* Sp.: From Molecular Machines to Hierarchical Modeling" (www.genomes-to-life.org). Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the U.S. Department of Energy under Contract DE-AC04-94AL85000.

REFERENCES

1. G. S. Davidson et al., "Knowledge mining with VxInsight: Discovery through interaction," *J. Intell. Inform. Sys.*, 11(3): 259–285, 1998.
2. M. Schena et al., "Quantitative monitoring of gene expression patterns with a complementary DNA microarray," *Science*, 270(5235): 467–470, 1995.
3. J. L. DeRisi, V. R. Iyer, and P. O. Brown, "Exploring the metabolic and genetic conrol of gene expression on a genomic scale," *Science*, 278(25): 14863–14868, 1997.

4. P. O. Brown and D. Botstein, "Exploring the new world of the genome with DNA microarrays," *Nature Genet.*, 21: 33–37, 1999.
5. M. Eisen et al., "Cluster analysis and display of genome-wide expression patterns," *Proc. Natl. Acad. Sci.*, 95(25): 14863–14868, 1998.
6. P. Spellman et al., "Comprehensive identification of cell cycle-regulated genes of the yeast Saccharomyces cerevisiae by microarray hybridization," *Mol. Biol. Cell*, 9(12): 3273–3297, 1998.
7. T. Golub et al., "Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring," *Science*, 286(5439): 531–537, 1999.
8. O. Alter, P. O. Brown, and D. Botstein, "Singular value decompositon for genome-wide expression data processing and modeling," *Proc. Natl. Acad. Sci.*, 97(18): 10101–10106, 2000.
9. H. C. Causton, J. Quackenbush, and A. Brazma, *Microarray Gene Expression Data Analysis: A Beginner's Guide*, Blackwell Publishers, Malden, MA, 2003.
10. S. Draghici, *Data Analysis Tools for DNA Microarrays*, Chapman & Hall, Boca Raton, FL, 2003.
11. M.-L. T. Lee, *Analysis of Microarray Gene Expression Data*., Kluwer Academic, Boston, MA, 2004.
12. M. Schena, *Microarray Analysis*, Wiley, New York, 2002.
13. G. E. P. Box, J. S. Hunter, and W. G. Hunter, *Statistics for Experimenters, Design, Innovation, and Discovery*, Wiley, Hoboken, NJ, 2005.
14. J. F. Zolman, *Biostatistics*, Oxford University Press, New York, 1993.
15. J. R. Thompson, *Simulation: A Modeler's approach*, Wiley, New York, 2000.
16. J. W. Tukey and F. Mosteller, Data analysis and regression, in F. Mosteller (Ed.), *Addison-Wesley Series in Behavioral Science: Quantitative Methods*, Addison-Wesley, Reading, MA, 1977.
17. I. R. Savage, "Contributions to the theory of rank order statistics—the two-sample case," *Ann. Math. Stat.*, 27: 590–615, 1956.
18. M. Bilban et al., "Normalizing DNA microarray data," *Curr. Issues Mol. Biol.*, 4(2): 57–64, 2002.
19. J. Quackenbush, "Microarray data normalization and transformation," *Nature Genet*, 32: 496–501, 2002.
20. G. Davidson, B. Wylie, and K. Boyack. "Cluster stability and the use of noise in interpretation of clustering", in *7th IEEE Symposium on Information Visualization (InfoVis 2001)*, San Diego, CA, 2001.
21. P. Tamayo et al., "Interpreting patterns of gene expression with self-organizing maps: Methods and application to hematopoietic differentiation," *Proc. Natl. Acad. Sci.*, IEEE Computer Society, Washington, D.C., 96(6): 2907–2912, 1999.
22. A. J. Enright and C. A. Ouzounis, "BioLayout—an automatic graph layout algorithm for similarity visualization," *Bioinformatics*, 17(9): 853–854, 2001.
23. R. Shamir and R. Sharan. "CLICK: A clustering algorithm with applications to gene expression analysis," in *Proc. Int. Conf. Intell. Syst. Mol. Biol.* (*ISMB*), 8:307–316, AAAI Press, Menlo Park, CA, 2000.
24. Y. Xu, V. Olman, and D. Xu, "Clustering gene expression data using a graph-theoretic approach: an application of minimum spanning trees," *Bioinformatics*, 18(2): 536–545, 2002.

25. A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: A review," *ACM Comput. Surv.*, 31(3): 264–323, 1999.
26. R. R. Wilcox, *Fundamentals of Modern Statistical Methods: Substantially Improving Power and Accuracy*, Springer-Verlag, New York, 2001, p. 110.
27. R. R. Wilcox, "Introduction to robust estimation and hypothesis testing", in G. J. L. a.I. Olkin (Ed.), *Statistical Modeling and Decision Science*, Academic, San Diego, CA, 1977, p. 188.
28. *Online Mendelian Inheritance in Man (OMIM)*, McKusick-Nathans Institue for Genetic Medicine, John Hopkins University (Baltimore, MD), National Center for Biotechnology Information, and National Library of Medicine (Bethesda, MD), 2000.
29. S. Kim et al., "A gene expression map for Caenorhabditis elegans," *Science*, 293(5537): 2087–2092, 2001.
30. T. Fruchtermann and E. Rheingold, *Graph Drawing by Force-Directed Placement*, University of Illinois, Urbana-Champagne, IL, 1990.
31. S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, 220(4598): 671–680, 1983.
32. M. Werner-Washburne et al., "Comparative analysis of multiple genome-scale data sets," *Genome Res.*, 12(10): 1564–1573, 2002.
33. J. M. Stuart et al., "A gene co-expression network for global discovery of conserved genetic modules," *Science*, 302: 249–255, 2003.
34. G. S. Davidson et al., *High Throughput Instruments, Methods, and Informatics for Systems Biology*, Sandia National Laboratories, Albuquerque, NM, 2003.
35. Y. Koren, L. Carmel, and D. Harel, "Drawing huge graphs by algebraic multigrid optimization," *Multiscale Modeling and Simulation*, 1(4): 645–673, 2003.
36. C. Walshaw, "A multilevel algorithm for force-directed graph drawing", in *Proceedings of the Eighth International Symposium on Graph Drawing (GD)*, Springer-Verlag, London, UK, 2000.
37. B. Hendrickson and R. Leland, "A Multi-Level Algorithm for Partitioning graphs", in *Supercomputing*, ACM Press, New York, 1995.
38. S. Roweis and L. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, 290(5500): 2323–2326, 2000.
39. J. B. Tenenbaum, V. D. Silva, and C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science*, 290(5500): 2319–2323, 2000.
40. A. Ben-Hur, A. Elisseeff, and I. Guyon. "A stability based method for discovering structure in clustered data," in R. Attman, A. K. Dunker, L. Hunter, T. E. Klein (Eds.), *Pacific Symposium on Biocomputing*, World Scientific Hackensack, NJ, 2002.
41. I. Guyon et al., "Gene selection for cancer classification using support vector machines," *Machine Learning*, 46: 389–422, 2002.
42. J. Jaeger, R. Sengupta, and W. L. Ruzzo. "Improved gene selection for classification of microarrays," in R. Attman, A. K. Dunker, L. Hunter, T. E. Klein (Eds.), *Pacific Symposium on Biocomputing*, World Scientific Hackensack, NJ, 2003.
43. B. Efron, "Bootstrap methods: Another look at the jackknife," *Ann. Stat.*, 7(37): 1–26, 1979.
44. P. Helman, R. Veroff, S. R. Atlas, C. Willman, "A Bayesian network classification methodology for gene expression data," *J. Comput. Biol.*, 11(4): 581–615, 2004.
45. I. T. Jolliffe, *Principal Component Analysis*, Springer-Verlag, New York, 2002.
46. M. Kirby, *Geometric Data Analysis*, Wiley, New York, 2001.

CHAPTER 5

In Silico Radiation Oncology: A Platform for Understanding Cancer Behavior and Optimizing Radiation Therapy Treatment

G. STAMATAKOS, D. DIONYSIOU, and N. UZUNOGLU

5.1. *PHILOSOPHIAE TUMORALIS PRINCIPIA ALGORITHMICA*: ALGORITHMIC PRINCIPLES OF SIMULATING CANCER ON COMPUTER

Completion of the sequencing of the human genome and cataloging and analysis of every protein in the human body (proteomics) that are currently underway have shaped a completely new and promising environment in the vast area of biomedical sciences and technology. Detailed analytical understanding of a plethora of molecular mechanisms has already been successfully exploited for diagnostic and therapeutic purposes (e.g., computer drug design, gene therapy). Nevertheless, in many critical cases such as in the case of cancer, understanding disease at the molecular level, although imperative, is not generally a sufficient condition for a successful treatment. Cancer [1, pp. 1247–1294; 2, pp. 1006–1096; 3–5] is the second most frequent cause of death in the developed countries. The astonishing complexity and degree of interdependence among the elementary biological mechanisms involved in tumor growth and response to therapeutic modalities as well as the partly stochastic character of cancer behavior dictate an extension of the analytical understanding of the disease to higher levels of biological complexity. Subcellular, cellular, tissue, organ, system, organism, and population levels should also be addressed with rigor analogous to the one characterizing the molecular approach. This is by no means an easy task. The challenge to mathematically describe cancer either analytically or algorithmically might well be paralleled to the challenge of mathematically describing planetary motion as was posed millenia ago.

Genomics and Proteomics Engineering in Medicine and Biology. Edited by Metin Akay

Nevertheless, *cancer is a natural phenomenon too and as such it must be amenable to some sort of mathematical description*. Even though analytical mathematics can be used to construct models of simple, mainly experimental tumor geometries such as tumor spheroids or nonsolid tumors, it does not seem particularly adequate for the description of realistic tumors in vivo with complex geometries, complex metabolic activity, and complex spatial proliferation distribution. On the contrary, discrete-state algorithmic descriptions of the system under consideration has been shown to be a quite efficient approach. Therefore, some sort of *philosophiae tumoralis principia algorithmica* (algorithmic principles of oncological philosophy) is to be expected to emerge in the near future. Evidently, experimental and clinical validation of such hypothetical principles in conjunction with the determination of their predictability limits would play a central role is such an approach.

Especially concerning radiation therapy, current treatment-planning algorithms are based on the concept of physical optimization of the dose distribution and rely on rather crude biological models of tumor and normal tissue response. Such algorithms practically ignore the highly complicated dynamic behavior of malignant cells and tissues. The introduction of advanced biosimulation methods based on cell proliferation mechanisms and also on information drawn from the cellular and molecular properties of each individual malignancy and each individual patient is expected to substantially improve the radiation therapy efficiency. This would be accomplished by using alternative fractionations, spatial dose distributions, and even combination with other therapeutic modalities such as chemotherapy, hyperthermia, and so on. Therefore, efficient modeling, simulation, and visualization of the biological phenomena taking place before, during, and after irradiation are of paramount importance. Discrete-time algorithmic descriptions (simulations) of the various phenomena offer the possibility of taking into account a large number of involved mechanisms and interactions. The same philosophy has already been extensively applied to purely technological problems, and the emerged numerical methods [e.g., the finite-difference time-domain (FDTD) technique] have proved to be very efficient and reliable. A further prominent characteristic of the biological phenomena under consideration is stochasticity. The fate of a single irradiated cell cannot be accurately predicted, for example. Only survival probabilities can be assigned to the cell based on the accumulated experimental and clinical observations made on large cell populations. Furthermore, the exact spatiotemporal distribution of the various cell cycle phases within the tumor volume is generally unknown, although some plausible macroscopic hypotheses can be made. Therefore, stochastic techniques such as the generic Monte Carlo method seem to be particularly appropriate for the prediction of tumor growth and response to radiation therapy.

The practical usefulness of such methods is both to improve understanding of the cancer behavior and to optimize the spatiotemporal treatment plan by performing in silico (on the computer) experiments before the actual delivery of radiation to the patient. In other words the clinician would be able to perform computer simulations of the likely tumor and adjacent normal tissue response to different irradiation scenarios based on the patient's individual imaging, histologic, and

genetic data. The simulation predictions would support him or her in selecting the most appropriate fighting strategy. To this end a substantial number of experimental and mathematical models have been developed. On the contrary, a rather small number of actual three-dimensional computer simulation models have appeared in the literature. Exploitation of the potential of current visualization techniques is even more limited.

This chapter begins with a brief literature review concerning experimental, mathematical, and computer simulation models of tumor growth, angiogenesis, and tumor and normal tissue response to radiation therapy. Reference to papers describing visualization algorithms used in oncologic simulations is also made. In a novel Monte Carlo simulation model developed by the *In Silico* Oncology Group of the National Technical University of Athens and including algorithms of in vivo tumor growth and response to irradiation, a specific application of the model to glioblastoma multiforme case and three-dimensional visualization of the predicted outcome is outlined. The chapter concludes with a critical evaluation of the presented paradigm, suggestions for further research, and a brief exposition of the future trends in in silico oncology.

5.2. BRIEF LITERATURE REVIEW

In the past four decades intensive efforts have been made in order to model tumor growth and tumor and normal tissue response to various therapeutic schemes such as radiation therapy. As the corresponding literature is particularly extended, only indicative examples of the modeling efforts are given in the following paragraphs.

Experimental models of tumor growth include two- and three-dimensional cell cultures (in vitro experimentation) and induction of tumors in laboratory animals (in vivo experimentation) [6–11]. Mathematical models of tumor growth attempt to analytically describe various aspects of the highly complex process, such as diffusion of oxygen and glucose [12, 13], control stability [14], competition between tumor and host [15], interdependence between structure and growth [16] and growth and stability [17, 18], temporal profile of tumor cell proliferation [19–21], tumor cell replication rules [22, 23], invasion [24], metastasis [25], cell cycle checkpoints [26], and angiogenesis [27–29]. The following approaches constitute representative examples of the modeling efforts. Adam and Maggelakis [12] analytically modeled the overall growth of a tumor spheroid using information about inhibitor production rates, oxygen consumption rates, volume loss and cell proliferation rates, and measures of the degree of nonuniformity of the various diffusion processes that take place. Casciari et al. [13] developed empirical correlations from experimental data to express mammary sarcoma of mouse (EMT6/Ro) tumor cell growth rates, oxygen consumption rates, and glucose consumption rates as functions of oxygen concentration, glucose concentration, and extracellular pH. Duechting [14] proposed a block diagram describing growth of normal cells as well as growth of benign and malignant tumors. He studied frequency and transition responses, locus diagrams, and stability conditions. Gatenby [15] developed a population ecology

mathematical model examining tumors as part of a dynamic society of interacting malignant and normal cells. Rizwan-Uddin and Saeed [16] presented predictions of a mathematical model of mass transfer in the development of a tumor, resulting in its eventual encapsulation and lobulation. Michelson and Leith [29] modeled the effect of the angiogenic signals of basic fibroblast growth factor (bFGF) and vascular endothelial growth factor (VEGF) on the adaptive tumor behavior.

Computer simulation models aim at three-dimensionally reconstructing a growing tumor based on the behavior of its constituent parts (either single cells or clusters of cells). Such models have been used in order to study, for example, the emergence of a spheroidal tumor in nutrient medium [30–38], the growth and behavior of a tumor in vivo [39–42], and the neovascularization (angiogenesis) process [31]. Duechting [31] developed a three-dimensional simulation model of tumor growth in vitro by combining systems analysis, control theory, and cellular automata. Wasserman and Acharya [39] developed a macroscopic tumor growth model mainly based on the mechanical properties of the tumor and the surrounding tissues. Kansal et al. [40, 41] proposed a three-dimensional cellular automaton model of brain tumor growth by using four parameters and introducing an adaptive grid lattice.

Experimental models of tumor response to radiation therapy primarily aim at determining the survival probability of the irradiated cells as a function of the absorbed dose (survival curves). The values of many other parameters of interest can also be estimated [43–45]. Mathematical models attempt to analytically describe the effect of ionizing radiation to tumor and normal tissue cells [5, 46–63]. Thames et al. [46] mathematically described the dissociation between acute and late radiation responses with changes in dose per fraction. Dale [51] extended the classical linear quadratic dose–effect relationship in order to examine the consequences of performing fractionated treatments for which there is insufficient time between fractions to allow complete damage repair. Fowler [53] reviewed the considerable progress achieved in fractionated radiotherapy due to the use of the linear quadratic model. Zaider and Minerbo [60] proposed a mathematical model of the progression of cells through the mitotic cycle under continuous low-dose-rate irradiation and applied it to studies of the effects of dose rate on HeLa cells. Jones and Dale [62] presented various modifications of the linear quadratic model that were used in order to optimize dose per fraction. Of special importance are the recent attempts to mathematically model the effect of specific genes (e.g., the *p53* status) to the radiation response of tumors [64]. Haas-Kogan et al. [64] modeled two distinct cellular responses to irradiation, *p53*-independent apoptosis, and *p53*-dependent G_1 arrest that characterize the radiation response of glioblastoma cells using the linear quadratic model. Mathematical modeling of chemotherapy and other treatment modalities that may be applied in parallel with radiation therapy has also been developed [65, 66].

Computer simulation models aim at three-dimensionally predicting and visualizing the response of a tumor [36, 67–86] or normal tissue [70] to various schemes of radiation therapy as a function of time. Nahum and Sanchez-Nieto [68] presented a computer model based on the concept of the tumor control probability (TCP)

and studied TCP as a function of the spatial dose distribution. Stamatakos et al. [71] developed a three-dimensional discrete radiation response model of an in vitro tumor spheroid and introduced high-performance computing and virtual reality techniques in order to visualize both the external surface and the internal structure of a dynamic tumor. Kocher et al. [76, 77] developed a simulation model of tumor response to radiosurgery (single-dose application) and studied the vascular effects.

Finally extensive work is being done on the combination of advanced visualization techniques, high-performance computing, and the World Wide Web capabilities in order to integrate and clinically apply the oncological simulation models [32–37, 79–81].

5.3. PARADIGM OF FOUR-DIMENSIONAL SIMULATION OF TUMOR GROWTH AND RESPONSE TO RADIATION THERAPY IN VIVO

5.3.1. Data Collection and Preprocessing

The imaging data [e.g., computed tomography (CT), magnetic resonance imaging (MRI), and positron emission tomography (PET)], the histopathologic (e.g., type of tumor) and genetic data (e.g., *p53* status, if available) of the patient are appropriately collected. The distribution of the absorbed dose in the region of interest at the end of the physical treatment planning procedure is also acquired.

The imaging data are introduced into a dedicated preprocessing software tool. If imaging data from diverse modalities are available, appropriate image registration techniques are used [85]. The clinician delineates the tumor and other structures of interest by using the dedicated software tool (Fig. 5.1). Each structure consists of a number of contours defined in successive tomographic slices. Subsequently, the imaging data, including the definition of the structures of interest, are adequately preprocessed, so as to be converted into the appropriate form, which will constitute the input for the simulation software. Preprocessing includes interpolation procedures in case of anisotropic data: gray-level interpolation for the imaging data and, most importantly, shape-based interpolation for the structures of interest [85]. The interpolation procedure is applied to every structure of interest and the results are combined in an image that constitutes the input of the simulation software. In this final image each structure is represented by its characteristic gray level (Fig. 5.2).

5.3.2. Data Visualization

The output of the above procedure is introduced into the visualization package AVS (Advanced Visualization Systems)/Express 4.2, which performs the visualization of the region of interest. AVS/Express is also used for the visualization of the simulation results. AVS/Express offers highly interactive three-dimensional data visualization capabilities, which facilitate the analysis and interpretation of the

FIGURE 5.1. The interface of the dedicated software tool for the delineation of anatomical structures of interest and the preprocessing procedure of the imaging data.

modeling results. It provides predefined components ("modules") for data acquisition and visualization with volume- or surface-rendering techniques. Predefined or user-defined modules can be combined to form complex "networks" of data manipulation. It offers modules for intersection of data in different cutting planes

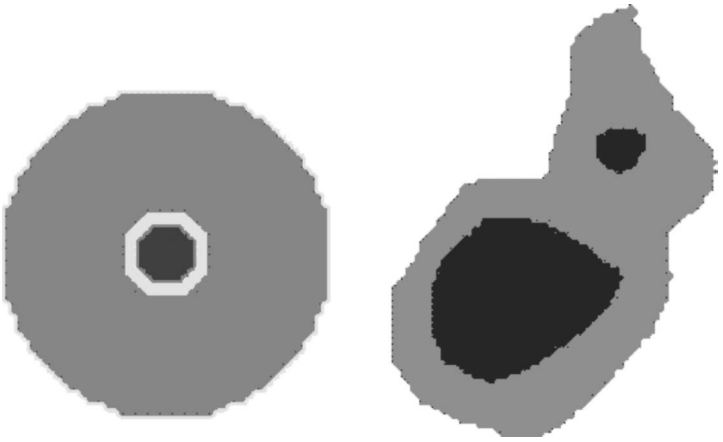


FIGURE 5.2. Indicative three-dimensional visualizations of the region of interest for a hypothetical tumor using AVS/Express 4.2.

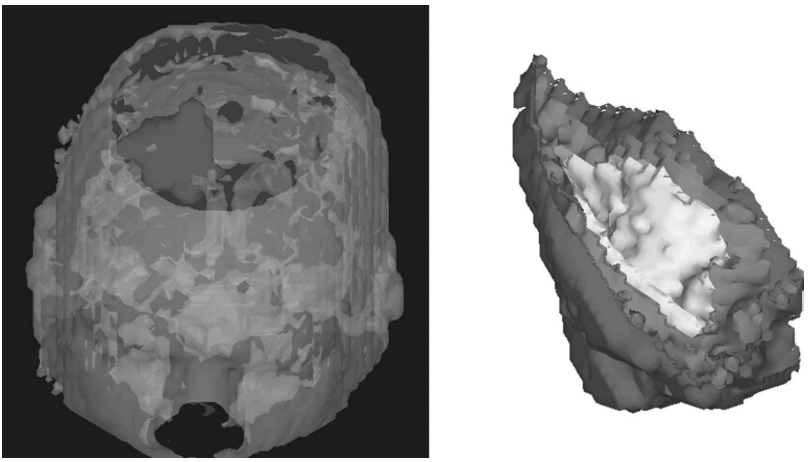


FIGURE 5.3. Two-dimensional equatorial slices from two indicative three-dimensional input data for the simulation software. Each structure is represented by its characteristic gray-level.

and orientations. In addition, the use of coloring and transparency features facilitates the visual inspection of complex topologies. For surface-rendering techniques the final representation can be exported from the package in VRML 1.0 or 2.0 format and become available for study in a machine-independent and interoperable way for local or remote examination through a local network or the Internet. Indicative three-dimensional visualizations performed with AVS/Express are presented in Figure 5.3.

5.3.3. Biology of Solid Tumor In Vivo: Algorithmic Expression

I. The cytokinetic model shown in Figure 5.4, based on the one introduced in [69, 70], is adopted. According to this model a tumor cell when cycling passes through the phases G_1 (gap 1), S (DNA synthesis), G_2 (gap 2), and M (mitosis). The corresponding maximum durations of these phases are designated TG_1, TS,

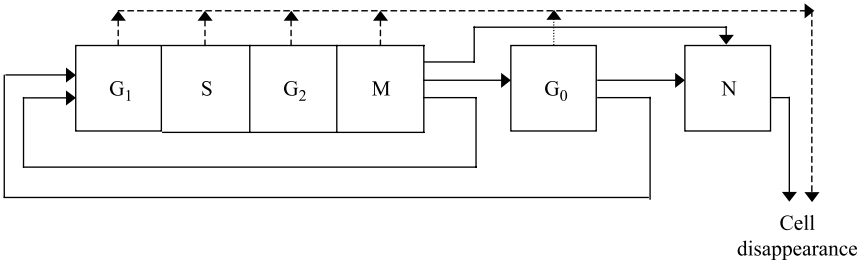


FIGURE 5.4. Cytokinetic model of a tumor cell. Symbol explanation: G_1: G_1 phase, S: DNA synthesis phase, G_2: G_2 phase, G_0: G_0 phase, N: necrosis, A: apoptosis.

TG_2, and TM. These durations, according to the literature, seem to follow the normal distribution. As a first approximation, we use the mean values of the duration of each cell cycle phase and neglect standard deviations.

After mitosis is completed, each one of the resulting cells reenters G_1 if oxygen and nutrient supply in its current position are adequate. Otherwise, it enters the G_0 resting phase in which, if oxygen and nutrient supply are inadequate, it can stay for a limited time (TG_0). Subsequently it enters the necrotic phase, unless the local environment of the cell becomes adequate before the expiration of TG_0. In the latter case the cell reenters G_1. In addition, there is also a probability that each cell residing in any phase other than necrosis or apoptosis dies and disappears from the tumor due to spontaneous apoptosis (dashed line in Fig. 5.4).

II. The description of the biological activity of the tumor is based on the introduction of the notion of the "geometric cell" (GC), the elementary cubic volume of a three-dimensional discretizing mesh covering the region of interest (Fig. 5.5). We assume that each GC of the mesh initially accommodates a number of biological cells (NBC). However, the maximum number of biological cells that can be accommodated in a GC is assumed to be NBC+NBC/2. Apparently NBC depends on the chosen size of the GC and determines the quantization error of the model. Biological cells are assumed to have a mass of 10^{-9} gr [3]; a typical cell density is therefore $10^{-6}\,\text{cells}/\text{mm}^3$. For example, in the case of a $1 \times 1 \times 1$-mm GC, NBC would be equal to 10^6.

III. Each GC of the mesh belonging to the tumor is assumed to contain biological cells distributed in a number of *equivalence classes* (compartments), each one characterized by the phase in which its cells are found (within or out of the cell cycle, i.e., G_1, S, G_2, M, G_0, necrosis, apoptosis).

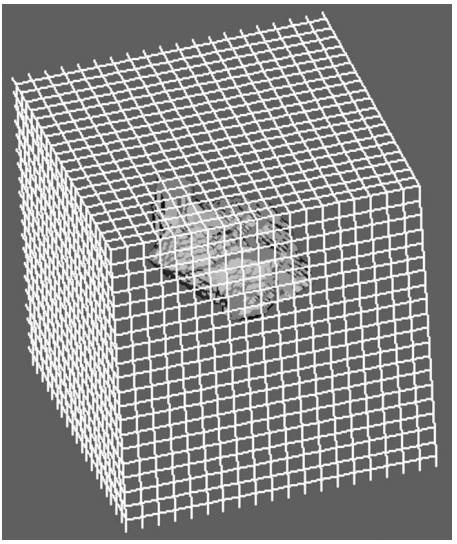


FIGURE 5.5. A three-dimensional discretizing mesh covers the region of interest.

IV. All biological cells of a given class within a GC are assumed to be synchronized, which means that, at a particular instant, the time they have already spent in the cell cycle phase characterizing the class under consideration is the same. Biological cells belonging to the same class but in different GCs are not considered synchronized.

V. The distribution of the initial NBC of a GC in each phase class is estimated according to the position of the corresponding GC within the tumor, namely based on the estimated metabolic activity in the local area [e.g., through PET/SPECT (single photon emission computed tomography) or functional MRI]. The information derived from the imaging data directs the division of the tumor region into subregions [86]. The GCs belonging to the same subregion are considered to be of roughly the same metabolic activity. Hence, subregions considered "proliferating cell regions," "resting-G_0 cell regions," or "dead cell regions" are defined. The determination of the relative fractions of proliferating, resting, and dead cells in each metabolic subregion depends on the histology of the tumor and on accumulated clinical experience. Such experience dictates, for example, that even a single alive clonogenic cell can repopulate a tumor mass).

VI. A "coloring criterion" must be formulated for the three-dimensional visualization of the simulation results. The coloring criterion "decides" on to which subregion a GC should be assigned at a specific instant. This rule is closely related to the definition of the subregions and to the current relative proportions of proliferating, resting, and dead cells within the GC. It also depends on the histopathological features of the tumor. As an example, in the case of glioblastoma multiforme tumors the following "98% coloring criterion" can be used:

```
For a GC of the discretizing mesh,
if the percentage of dead cells is lower than 98% then
        {if percentage of proliferating cells
                >percentage of G0 cells then
                  { paint the GC with the proliferating
                       cell region color}
         else { paint the GC with the G0 cell region
                color}}
else
         {paint the GC with the dead cell region color}
```

The main reason for intensifying the effect of the presence of proliferating and G_0 cells in the above criterion is that even a single alive clonogenic tumor cell (either cycling or in G_0) can lead to tumor recurrence.

VII. The initial distribution of the proliferating biological cells of a GC within each of the proliferating phases (G_1, S, G_2, M) is estimated using the duration of each cell cycle phase for the specific tumor.

5.3.4. Radiobiology of Solid Tumor in Vivo: Algorithmic Expression

The response of each cell to irradiation leading to absorbed dose D is described by the linear-quadratic (LQ) model, which is widely used in the pertinent literature

[3, 4, 69, 70]. According to this model, the survival probability S of a cell is given by the expression

$$S = \exp[-(\alpha D + \beta D^2)] \tag{5.1}$$

where D is the absorbed dose and α, β parameters characterize the initial slope and the curvature, respectively, of the survival curve. Equation (5.1) can be used for fractionated radiotherapy provided that there exists a sufficient time interval between fractions for sublethal damage repair to be completed [3].

The radiosensitivity of cells varies considerably as they pass through the subsequent phases of the cell cycle, with the S phase regarded as the most resistant [3, 44]. Cells in any proliferating phase (G_1, S, G_2, and M) are more radiosensitive than hypoxic cells residing in G_0. Furthermore, of particular notice is the fact that the parameters α and β of the LQ model constitute one possible way to incorporate the influence of genetic determinants, such as the *p53* gene status (mutations or expression of *p53* protein), into the simulation model [86].

5.3.5. Simulation Outline

Time is discretized and incremented. One hour has been adopted as the unit of time. In each time step the geometric mesh is scanned and the updated state of a given GC is determined as follows:

I. At the time instants that correspond to the delivery of a specific radiation dose D to the tumor, the number of cells killed in a particular GC is calculated based on the LQ model [Eq. (5.1)]. In a tumor growth simulation case, the dose D in Eq. (5.1) would be set to zero.

II. Lethally damaged cells following exposure to radiation undergo two mitotic divisions prior to death and disappearance from the tumor [4, p. 87].

III. At each time step the time registers of all GCs increase by 1 hour. All the necessary cell cycle phase transitions are computed.

IV. The possibilities of cell loss due to apoptosis and necrosis are computed by using the equations

$$\text{CBR} = \frac{\text{GF}}{T_C} \tag{5.2}$$

$$\text{CLF} = \frac{\text{CLR}}{\text{CBR}} \tag{5.3}$$

where CLF is the cell loss factor, CLR the cell loss rate, CBR the cell birth rate, GF the growth fraction, and T_C the duration of the cell cycle. The total cell loss factor is assumed to be the sum of the cell loss factor due to necrosis and the cell loss factor due to apoptosis and to remain constant throughout the simulation. Future versions of the model will investigate the assumption of a time-varying CLF. Possible variations of the cell cycle duration throughout the simulation are also under investigation.

The simulation of tumor expansion or shrinkage is based on the following rules: If the actual number of alive and dead (but still morphologically existing) tumor cells contained within a given GC is reduced to less than NBC/2, then a procedure which attempts to "unload" the remaining biological cells in the neighboring GCs takes place aimed at emptying the current GC. The basic criterion of the unloading procedure states that the unloading process proceeds in such a way that the biological cell density of the entire lattice is as uniform and as close to "NBC per GC" as possible. Therefore, the unloaded cells are preferentially placed within the neighboring GCs having the maximum available free space. If two or more of the neighboring GCs possess the same amount of free space, then a random-number generator is used for the selection. If at the end of the unloading procedure the given GC becomes empty, it disappears from the tumor. An appropriate shift of a chain of GCs, intended to fill the "vacuum," leads to differential tumor shrinkage. This can happen, for example, after a number of cells are killed due to irradiation.

On the other hand, if the number of alive and dead cells within a given GC exceeds NBC+NBC/2, then a similar procedure attempting to unload the excess cells in the surrounding GCs takes place. If the unloading procedure fails to reduce the number of cells to less than NBC+NBC/2, then a new GC emerges. Its position relative to the "mother" GC is determined using a random-number generator. An appropriate shifting of a chain of adjacent GCs leads to a differential expansion of the tumor. The "newborn" GC initially contains the excess number of biological cells, which are distributed in the various phase classes proportionally to the distribution in the mother GC. The impact of the definition of the upper and lower thresholds, which in the present case have been set to NBC + NBC/2 and NBC − NBC/2, respectively, on the uniformity of the distribution of the biological cells throughout the lattice is under investigation.

The differential tumor expansion and shrinkage algorithms are based on the use of random-number generators in conjunction with adequately formed morphological rules. These rules aim at tumor shrinkage or expansion conformal to the initial shape of the tumor. This is a logical assumption if the pressure in the normal tissues surrounding the tumor region is assumed to be uniform and the tumor is not in contact with practically undeformable tissues such as the bone.

More specifically, in the case of selection of shrinkage direction, the outermost tumor GC is detected along each one of six possible directions of shrinkage (Cartesian coordinate system *XYZ* centered at the current GC, each axis defining two possible directions of movement). Its "six-neighbor" GCs belonging to the tumor (NGCT) are counted. The direction corresponding to the maximum NGCT is finally selected out of the six possible directions as the direction along which the shifting of the GCs will take place (shifting direction). If more than one shifting direction has the same maximum NGCT, then the selection is based on the use of a random-number generator. A similar, though inverse, morphological-mechanical rule can be applied in the case of tumor expansion. An alternative algorithm, which will be used in future versions of the simulation model, performs shrinkage and expansion along a line of random angle. In this way, artifacts attributed to the movement of GCs along the axes of the Cartesian coordinate system will be avoided.

The need for the formulation of the above morphological rules for tumor shrinkage and expansion has arisen from the inspection of the macroscopic results of the simulation algorithms. A completely random selection of one out of the six possible shifting directions results in a premature extensive fragmentation of the tumor region in case of radiotherapy, which is usually incompatible with clinical experience. The general trend is a conformal shrinkage of most solid tumors (Fig. 1.4 in [4]).

The mechanical properties of the surrounding normal tissue are considered uniform around the tumor, with the exception of an absolute lack of deformability of the bone. As a first approximation, immunological reactions, invasion, and formation of metastases have been ignored.

5.3.6. Parametric Testing of Simulation Model: Case of Glioblastoma Multiforme Irradiated by Various Fractionation Schemes

A case of a glioblastoma multiforme (GBM) tumor recently irradiated has been selected. A specialized doctor has delineated the clinical boundary of the tumor and its necrotic area based on the corresponding MRI and PET data after irradiation (Fig. 5.6) (hysteron proteron for validation reasons). A three-dimensional mesh quantizing the anatomical region of interest has been considered. The dimensions of each GC are $1 \times 1 \times 1$ mm. Such a volume contains roughly 10^6 biological cells (NBC $= 10^6$). Figure 5.7 depicts a three-dimensional visualization of the tumor before the beginning of radiotherapy treatment.

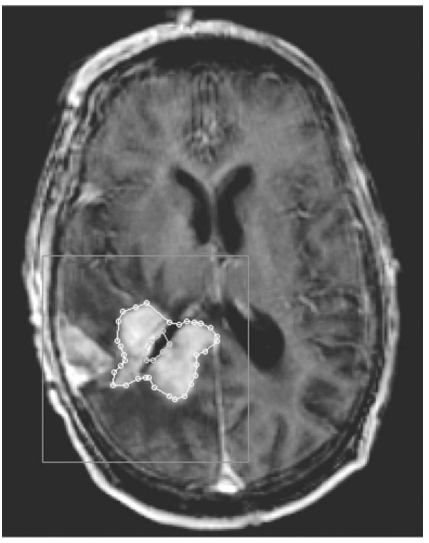


FIGURE 5.6. An MRI slice depicting a glioblastoma multiforme brain tumor recently irradiated. Both the clinical volume of the tumor and its central necrotic area have been delineated.

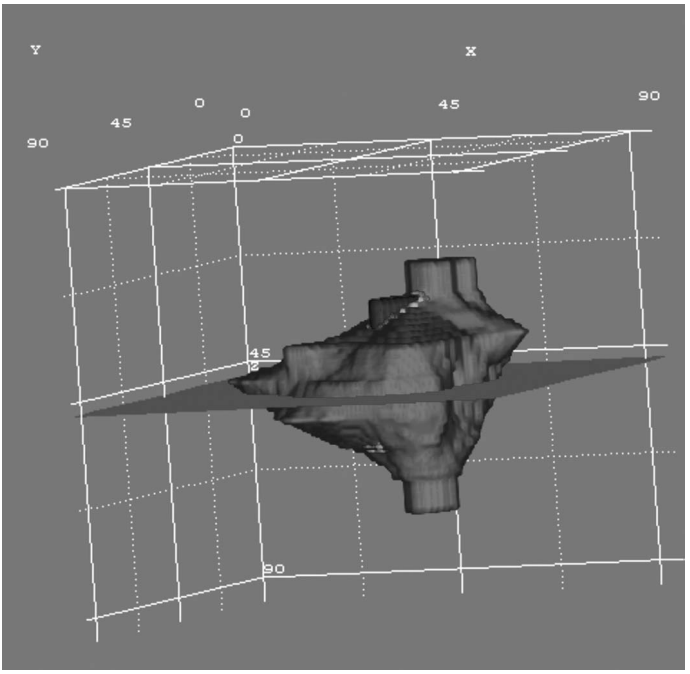


FIGURE 5.7. Three-dimensional visualization of the tumor before the beginning of the irradiation. Volume rendering produced with AVS/Express 4.2.

As no information about the metabolic activity (and therefore the density of the tumor neovasculature) prior to irradiation was available (e.g., through PET or functional MRI) for the particular case considered, the growth support criterion applied so far was the minimization of the distance from the outer surface of the tumor. This implies that biological cells residing in the outer layer of the tumor ("proliferating cell layer") can be adequately oxygenated and fed whereas the inner part of the tumor ("dead-cell layer") lacks efficient neovasculature and therefore oxygenation and nourishment. An intermediate layer containing a significant amount of G_0 cells has also been assumed ("G_0 cell layer") . Obviously, the above layered structure may not be the case in a large number of tumors. If, for example, the metabolic imaging data (e.g., PET, SPECT, functional MRI) prior to irradiation suggest that the metabolic activity of the tumor is rather uniform throughout its volume, the growth support criterion would become rather uniform too. For the specific type of tumor all nonclonogenic cells are considered to be necrotic (sterile cells are not taken into account). This is a logical first approximation, since GBM is generally considered a poorly differentiated type of tumor. A typical clonogenic cell density is 10^7 to 10^8 cells/cm^3 (10^4 to 10^5 cells/mm^3) [62, 68]. We assume a clonogenic cell density of 2×10^4 cells/mm^3 in the proliferating cell layer (a 6-mm-thick layer from the outer boundary of the tumor), 10^4 cells/mm^3 in the G_0 cell layer (a 1-mm-thick

layer surrounding the central necrotic region), and 0.2×10^4 cells/mm^3 in the dead-cell layer of the tumor. Within each geometric cell the initial distribution of the clonogenic cells through the cell cycle phases depends on the layer of the tumor in which the geometric cell belongs. More precisely, in the proliferating cell layer 70% of the clonogenic cells are assumed to be in the cycling phases and 30% in the G_0 phase. In the G_0 cell layer 30% of the clonogenic cells are in the cycling phases and 70% in the G_0 phase. Finally, in the dead-cell layer 10% of the clonogenic cells are in the cycling phases and 90% in the G_0 phase.

5.3.6.1. Constant Radiosensitivity Throughout Cell Cycle: High Cell Loss Factor In a first experiment the response of a hypothetical, radiosensitive GBM tumor to a standard fractionation scheme (2 Gy once a day, 5 days per week, 60 Gy total) has been simulated. The LQ model parameters of this hypothetical tumor have been assumed as follows: $\alpha = 0.6\,\text{Gy}^{-1}$, $\beta = 0.06\,\text{Gy}^{-2}$ [83]. They have also been assumed to remain constant throughout cell cycle. Other parameters of importance for this experiment were cell cycle duration $T_C = 30$ h; cell cycle phase durations $\text{TG}_1 = 11$ h, $\text{TS} = 13$ h, $\text{TG}_2 = 4$ h, $\text{TM} = 2$ h, and $\text{TG}_0 = 25$ h [87]; and cell loss factor taken equal to 0.9 [3]. Such a high cell loss factor has been selected in order to facilitate the demonstration of the ability of the model to simulate the shrinkage effect. We assume that the total cell loss factor is the sum of the cell loss factor due to necrosis (0.8) and the cell loss factor due to apoptosis (0.1).

The computer code has been developed in Microsoft Visual C++ 6 and Microsoft Visual Basic 6. As far as the computational demands are concerned, an execution of the radiation therapy simulation of 6 weeks ($96 \times 96 \times 96$ geometric cells, each one of dimension $1 \times 1 \times 1$ mm) on an AMD Athlon XP 1800 machine (786 MB RAM) takes about 10 min.

The testing predictions depicted in Figures 5.8 and 5.9 demonstrate the ability of the model to adequately simulate cell death and tumor shrinkage. In order to emphasize this potential of the model, in this explorative case the values of certain parameters (e.g., cell loss) have been deliberately exaggerated.

5.3.6.2. Influence of p53 Status The molecular basis of cell radiosensitivity has been extensively studied during the last decades. Representative efforts drawn from the extensive corresponding literature have been given in [86]. The roles of wild-type (wt) *p53* in modulating DNA repair, apoptosis, and the G_1 cell cycle arrest have each been implicated in the regulation of cellular response to ionizing radiation. A remarkable number of studies associate *p53* mutations with increased radioresistance and poor clinical outcome for patients with GBM.

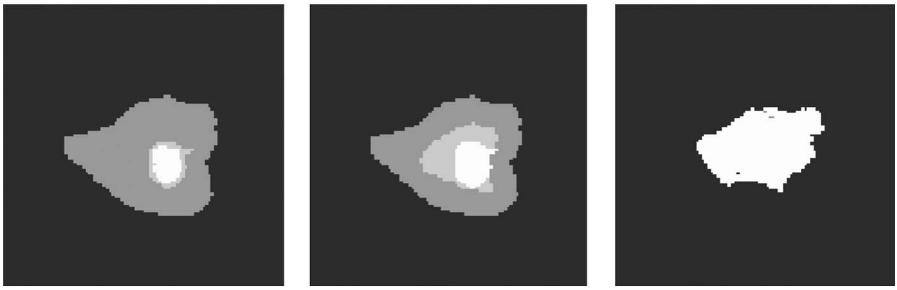
In this parametric study the results of Haas-Kogan et al. [64, 88] have been used. The authors in [64] investigated the influence of *p53* status on radiation-induced apoptosis and G_1 cell cycle arrest of GBM cells. They found that radiation-induced apoptosis of GBM cells occurred in a manner independent of wt *p53*, in contrast to G_1 cell cycle arrest, which was *p53* dependent. An increased radioresistance was observed in irradiated G_1 cells lacking functional wt *p53*, manifested

(*a*) (*b*) (*c*)

FIGURE 5.8. Irradiation according to the standard fractionation scheme (2 Gy once a day, 5 days per week, 60 Gy in total). Three-dimensional sections of the tumor using the cutting plane shown in figure 7. Surface rendering produced with AVS/Express 4.2. The cutting plane shown in figure 7 has been considered. (*a*) Before the beginning of irradiation, (*b*) 2 fictitious day after the beginning of irradiation, and (*c*) 3 fictitious days after the beginning of irradiation. Color code → red: proliferating cell layer, green: dormant cell layer (G_0), blue: dead cell layer. The values of certain parameters (e.g., cell loss) have been deliberately exaggerated in order to facilitate the demonstration of the ability of the model to simulate the shrinkage effect.

by a relatively lower α and α/β. Furthermore, in [88] they studied the influence of *p53* function on the effect of fractionated radiotherapy of GBM tumors and concluded that fractionated radiotherapy provides a selective advantage to GBM cells expressing mutant *p53* (mt *p53*).

Based on these results, we performed a parametric study of radiation response to an accelerated fractionation scheme (2 Gy twice a day, 5 days per week, 60 Gy in



(*a*) (*b*) (*c*)

FIGURE 5.9. (*a*) A centrally located slice of the tumor before the beginning of irradiation, (*b*) simulated response of the tumor to radiation therapy 2 fictitious days after the beginning of the radiotherapy course, and (*c*) simulated response of the tumor to radiation therapy 3 fictitious days after the beginning of the radiotherapy course. The cutting plane shown in figure 7 has been considered. Grey scale code → dark gray: proliferating cell layer, light gray: dormant cell layer (G0), white: dead cell layer. The values of certain parameters (e.g., cell loss) have been deliberately exaggerated in order to emphasize the ability of the model to simulate tumor shrinking as a response to radiation therapy.

total), assuming that the GBM tumor of Figures 5.6 and 5.7 was with (I) wt *p53* and (II) mt *p53*. Specifically, we assumed:

I. GBM tumor with intact wt *p53* function [64]:

$$\alpha_P = 0.61\ \text{Gy}^{-1} \qquad \alpha_S = 0.407\ \text{Gy}^{-1} \qquad \alpha_{G0} = 0.203\ \text{Gy}^{-1}$$
$$\beta_P = 0.02\ \text{Gy}^{-2} \qquad \beta_S = 0.02\ \text{Gy}^{-2} \qquad \beta_{G0} = 0.02\ \text{Gy}^{-2}$$

II. GBM tumor with mt *p53* [64]:

$$\alpha_P = 0.17\ \text{Gy}^{-1} \qquad \alpha_S = 0.113\ \text{Gy}^{-1} \qquad \alpha_{G0} = 0.057\ \text{Gy}^{-1}$$
$$\beta_P = 0.02\ \text{Gy}^{-2} \qquad \beta_S = 0.02\ \text{Gy}^{-2} \qquad \beta_{G0} = 0.02\ \text{Gy}^{-2}$$

The meanings of the symbols used are the following:

$\alpha_P,\ \beta_P$	LQ model parameters for all proliferative cell cycle phases except for DNA synthesis phase (S phase)
$\alpha_S,\ \beta_S$	LQ model parameters for S phase
$\alpha_{G0},\ \beta_{G0}$	LQ model parameters for resting G_0 phase

The mean values of α_S and α_{G0} have been assumed as perturbations of the α_P mean values, consistent with the findings of experimental radiobiology. Specifically, we assume $\alpha_S = 2\alpha_P/3$ and $\alpha_{G0} = \alpha_P/3$. These values for α_P and α_{G0} give an OER (oxygen enhancement ratio) equal to 3, consistent with the literature [4]. As far as the β behavior is concerned, we use $\beta_P = \beta_S = \beta_{G0}$, based on [64].

The cell cycle duration T_C is taken to be 24 h. This is the average of the cell cycle durations we have found in the literature for GBM cell lines [54, 88, 89]. In [90] the approximate percentage of the cell cycle time spent in each phase by a typical malignant cell is given as

$$\text{TG}_1 = \frac{40}{100}T_C \qquad \text{TS} = \frac{39}{100}T_C \qquad \text{TG}_2 = \frac{19}{100}T_C \qquad \text{TM} = \frac{2}{100}T_C$$

Based on the above distribution, for the considered cell cycle time of 24 h we get the following phase durations: $\text{TG}_1 = 9$ h, $\text{TS} = 8$ h, $\text{TG}_2 = 4$ h, and $\text{TM} = 1$ h. The duration of the G_0 phase is taken to be $\text{TG}_0 = 25$ h [87].

We assume a clonogenic cell density of 2×10^5 cells/mm^3 in the proliferating cell layer, 10^5 cells/mm^3 in the G_0 cell layer, and 0.2×10^5 cells/mm^3 in the dead-cell layer of the tumor [4, p. 84; 91]. The cell loss factor (CLF) has been taken equal to 0.3 [92]. In [93] the authors note that cell loss is mainly due to necrosis (CLFN) and apoptosis (CLFA) and that gliomas have a low CLF in general. We assume that the total CLF (0.3) is the sum of the CLFN (0.27) and CLFA (0.03). We

hypothesize low levels of apoptotic cells for GBM, as we have found that this is in general the case for gliomas [64, 93–95].

The simulation is assumed to begin ($t = 0$) on Monday 00:00 a.m. and, unless the tumor reaches earlier the boundaries of our cubic region of interest, to end on Sunday 24:00, 3 weeks later ($t = 504$ h). The delivery of irradiation takes place at 08:00 and 16:00 every day and the total duration of the accelerated fractionation scheme is 3 weeks. The interfraction interval (8 h) is considered sufficient for sublethal damage repair to be completed.

A typical simulation run of 6 weeks for a $96 \times 96 \times 96$ geometric mesh lasts about 15 min on an AMD Athlon XP 1800 machine (786 MB RAM). In order to ensure the numeric stability of the code, various executions have been performed in which different scanning directions and different initial seeds for the random-number generators have been used. The macroscopic features of the result of the simulation are not influenced by these variations.

Figure 5.10 depicts the number of alive cells (proliferating and G_0) as a function of time for the cases of a GBM with wt *p53* and mt *p53*. The delivery of the last dose fraction takes place at $t = 448$ h. At subsequent times, if the clonogenic cells in the tumor region have not been killed, they will begin to repopulate the tumor. The trend for reduction of the number of alive tumor cells during the radiotherapy scheme is clearly pronounced in the case of the tumor with a wt *p53*, which is considered to be more radiosensitive compared with the tumor with mt *p53*. In fact the tumor with mt *p53* is so radioresistant that radiation therapy fails to hinder clonogenic cells from rapidly proliferating during therapy. It should be noted that regions of potential microscopic disease have not been considered, and the accuracy of the simulation model cannot reach to the point of

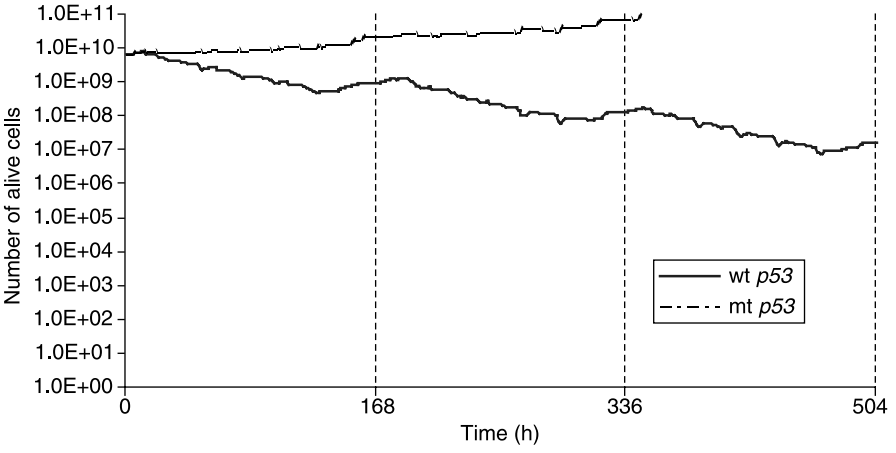
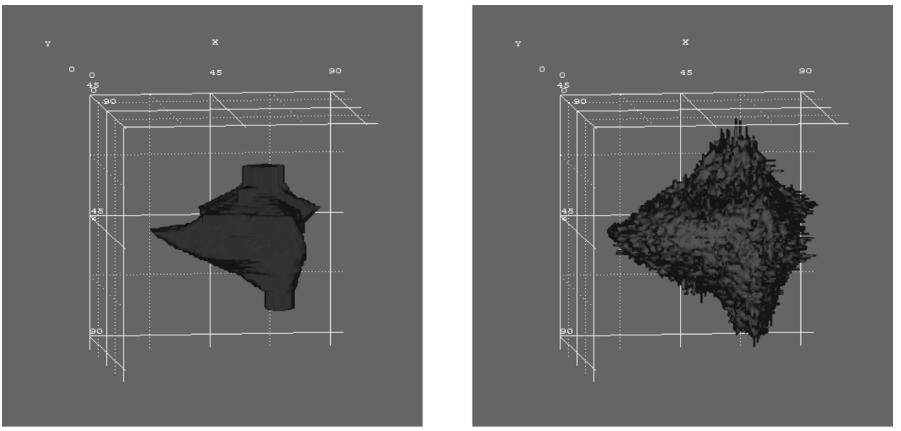


FIGURE 5.10. The number of alive cells (proliferating and resting) as a function of time for the tumors with wt and with mt *p53*.



wt *p53* mt *p53*

FIGURE 5.11. Three-dimensional visualization of the tumors with wt *p53* and mt *p53* two weeks after the beginning of the standard radiotherapy scheme (2 Gy once a day, 5 days a week, 60 Gy in total). Volume rendering produced with AVS/Express 4.2. Color code → red: proliferating cell layer, green: dormant cell layer (G0), blue: dead cell layer.

determining the fate of every single clonogenic cell. As a consequence, the "predictions" of the simulation model should always be interpreted with caution. In Figure 5.11 the tumors are three-dimensionally visualized two weeks after the start of the radiotherapy treatment. As expected, three-dimensional visualization offers improved insight into the macroscopic geometry and structure of the tumor.

It should also be stressed that apart from the α and β parameters of the LQ model, adjusted according to the results of [64, 88], so as to incorporate the influence of *p53* gene status, all other factors influencing the radiosensitivity of a tumor as a whole (e.g., hypoxic fraction, proportion of clonogenic cells, cell loss rate) have been kept unchanged during these comparative studies.

5.4. DISCUSSION

The in vivo simulation model presented in this chapter deals with a novel approach to the modeling of tumor growth and response to radiation therapy and is characterized by the unique combination of the following features:

(i) Possibility for the simulation of both untreated in vivo tumor growth and in vivo tumor response to radiotherapy
(ii) Consideration and use of the actual imaging, histopathologic, and genetic data for each particular clinical case
(iii) Incorporation of numerous biological mechanisms by means of an advanced algorithmic description

(iv) Introduction of the notion of the "geometric cell" and its constituent compartments, called "equivalence classes," corresponding to discrete phases within or outside the cell cycle

(v) Extensive use of random-number generators to simulate the stochastic nature of the various biological phenomena involved (Monte Carlo approach)

(vi) Discrete and modular character, which confers a high level of adaptability possibility for three-dimensional reconstruction and visualization of the results

A number of exploratory simulation tests have been performed for a clinical case of a GBM tumor. The influence of genetic determinants (such as the *p53* gene status) on tumor response to radiotherapy has been incorporated into the model by means of appropriately adjusting the LQ model parameters according to the GBM tumor literature.

The results of the simulation model have been semiquantitatively assessed. Comparison of the simulation results with the accumulated clinical experience demonstrates that the model has the potential of representing the clinical reality within acceptable reliability limits. Obviously, experimental and clinical feedback is always to be exploited in order to improve the model. To this end the software system is currently undergoing an extensive testing and adaptation procedure, basically by comparing the model "predictions" with clinical data before, during, and after radiotherapy courses. Eventual discrepancies will lead to a better estimation of specific model parameters such as the LQ α and β. Generic parameter estimation techniques such as the neural networks technique, taboo searching, and so on, will be used to this end. In parallel, advances in a vast range of the involved phenomena are constantly being translated into the algorithmic language in order to keep pace with the ever-accumulating knowledge in the corresponding scientific fields. The simulation model, being gradually refined, can also be used as a tool to study the relative importance of the mechanisms underlying tumor behaviour. Possible interrelationships between the parameters involved in tumor growth or tumor response to radiotherapy are currently being explored. Optimization of dose fractionation during radiation therapy by performing in silico experiments and individualization of treatment protocols constitute the long-term goals of this effort.

The simulation model presented so far may serve as a paradigm of an in silico approach to oncology. Although it is open to refinement and better adaptation through extensive clinical testing, it provides a comprehensive outline of the basic mechanisms taken into account and algorithmically expressed. Furthermore, as chemotherapy is frequently administered before, in parallel, or after radiation therapy, an analogous model simulating the special case of the GBM to the prodrug Temodal has already been simulated by our group. Simulation of the response of normal tissues to both radiation therapy and chemotherapy is also under development. It should nevertheless be pointed out that as there is a wide range of mechanisms of action of chemotherapy depending on the agent administered, substantial differences among the corresponding models are to be expected.

5.5. FUTURE TRENDS

Extensive combination of tumor behavior simulation models at the cellular and higher levels of biological complexity with advances in genomics and proteomics is expected to substantially strengthen the potential of the emerging discipline of in silico oncology and in particular of in silico radiation oncology. Integrated and highly automated decision support and treatment-planning systems combining microarray technology, image processing, and biosimulation software (including, e.g., radiotherapeutic and chemotherapeutic models) are becoming a necessary infrastructure for an analytical and rational approach to cancer diagnosis, prognosis, and eventual effective treatment. Therefore, special emphasis should be put on this heavily interdisciplinary combination process. Another point of outmost importance is the continuous update of the emerging integrated systems that should be based on the latest experimentally and clinically extracted knowledge as well as on the newest advances of computer science and technology.

ACKNOWLEDGMENTS

Dr. P. Asvestas, National Technical University of Athens, is duly acknowledged for his advice on specific image processing issues. The project is cofunded by the European Social Fund (75%) and National Resources (25%) under the scheme Operational Programme for Educational and Vocational Training (EPEAEK-PYTHAGORAS II).

REFERENCES

1. H. Lodish, D. Baltimore, A. Berk, S. Zipursky, P. Matsudaira, and J. Darnell, *Molecular Cell Biology*, Scientific American Books, New York, 1995, pp. 1247–1294.
2. J. Watson, N. Hopkins, J. Roberts, J. Steitz, and A. Weiner, *Molecular Biology of the Gene*, 4th ed., Benjamin/Cummings Publishing Company, Menlo Park, CA, 1987, pp. 1006–1096.
3. G. Steel (Ed.), *Basic Clinical Radiobiology*, Arnold, London, 1997, pp. 15, 47–48, 52–57, 123–131, 153, 161.
4. C. Perez and L. Brady, *Principles and Practice of Radiation Oncology*, Lippincott-Raven, Philadelphia, 1998, pp. 784–785.
5. J. F. Fowler, "Review of radiobiological models for improving cancer treatment," in K. Baier and D. Baltas (Eds.), *Modelling in Clinical Radiobiology*, Freiburg Oncology Series, Monograph No. 2, Albert-Ludwigs University, Freiburg, Germany, 1997, pp. 1–14.
6. M. Santini, G. Rainaldi and P. Indovina, "Multicellular tumor spheroids in radiation biology," *Int. J. Radiat. Biol*, 75(7): 787–799, 1999.
7. R. Jostes, M. Williams, M. Barcellos-Hoff, T. Hoshino, and D. Deen, "Growth delay in 9L rat brain tumor spheroids after irradiation with single and split doses of X rays," *Radiat. Res.*, 102(2): 182–189, 1985.
8. J. Casciari, S. Sotirchos, and R. Sutherland, "Variations in tumor cell growth rates and metabolism with oxygen concentration, glucose concentration, and extracellular pH," *J. Cell. Physiol.*, 151: 386–394, 1992.

9. M. Santini and G. Rainaldi, "Three-dimensional spheroid model in tumor biology," *Pathobiology*, 67: 148–157, 1999.
10. C. Nirmala, J. S. Rao, A. C. Ruifrock, L. A. Langford, and M. Obeyesekere, "Growth characteristics of glioblastoma spheroids," *Int. J. Oncol.*, 19: 1109–1115, 2001.
11. J. P. Freyer and R. M. Sutherland, "Regulation of growth and development of necrosis in EMT6/Ro multicellular spheroids by the glucose and oxygen supply," *Cancer Res.*, 46: 3504–3512, 1986.
12. J. A. Adam and S. A. Maggelakis, "Diffusion regulated growth characteristics of a spherical prevascular carcinoma," *Bull. Math. Biol.*, 52: 549–582, 1990.
13. J. J. Casciari, S. V. Sotirchos, and R. M. Sutherland, "Glucose diffusivity in multicellular tumor spheroids," *Cancer Res.*, 48: 3905–3909, 1988.
14. W. Duechting, "Krebs, ein instabiler Regelkreis.Versuch einer Systemanalyse," *Kybernetik*, 5(2): 70–77, 1968.
15. R. A. Gatenby, "Models of tumor-host interaction as competing populations: Implications for tumor biology and treatment," *J. Theor. Biol.*, 176: 447–455, 1995.
16. R. Uddin and I. M. Saeed, "Structure and growth of tumors: The effect of Cartesian, cylindrical, and spherical geometries," *Ann. Acad. Sci.*, 858: 127–136, 1998.
17. H. P. Greenspan, "On the growth and stability of cell cultures and solid tumors," *J. Theor. Biol.*, 56: 229–242, 1976.
18. H. Bremermann, "Reliability of proliferation controls. The Hayflick limit and its breakdown in cancer," *J. Theor. Biol.*, 97: 641–662, 1982.
19. R. Demicheli, R. Foroni, A. Ingrosso, G. Pratesi, C. Sorano, and M. Tortoreto, "An exponential-Gompertzian description of LoVo cell tumor growth from in vivo and in vitro data," *Cancer Res.*, 49: 6543–6546, 1989.
20. J. J. Terz, W. Lawrence, Jr., and B. Cox, "Analysis of the cycling and noncycling cell population of human solid tumors," *Cancer*, 40: 1462–1470, 1977.
21. G. Hejblum, D. Costagliola, A.-J. Valleron, and J.-Y. Mary, "Cell cycle models and mother-daughter correlation," *J. Theor. Biol.*, 131: 255–262, 1988.
22. K. A. Heichman and J. M. Roberts, "Rules to replicate by," *Cell*, 79: 557–562, 1994.
23. D. Wiarda and C. C. Travis, "Determinability of model parameters in a two-stage deterministic cancer model," *Math. Biosci.*, 146: 1–13, 1997.
24. R. A. Gatenby and E. T. Gawlinski, "A reaction-diffusion model of cancer invasion," *Cancer Res.*, 56: 5745–5753, 1996.
25. K. Iwata, K. Kawasaki, and N. Shigesada, "A dynamical model for the growth and size distribution of multiple metastatic tumors," *J. Theor. Biol.*, 201: 177–186, 2000.
26. A. D. Murray, "Creative blocks: Cell-cycle checkpoints and feedback controls," *Nature*, 359: 599–604, 1992.
27. I. I. H. Chen and R. L. Prewitt, "A mathematical representation for vessel network," *J. Theor. Biol.*, 97: 211–219, 1982.
28. D. Balding and D. L. S. Mc Elwain, "A mathematical model of tumor-induced capillary growth," *J. Theor. Biol.*, 114: 53–73, 1985.
29. S. Michelson and J. T. Leith, "Possible feedback and angiogenesis in tumor growth control," *Bull. Math. Biol.*, 59: 233–254, 1997.
30. W. Duechting and T. Vogelsaenger, "Three-dimensional pattern generation applied to spheroidal tumor growth in a nutrient medium," *Int. J. Biomed. Comput.*, 12(5): 377–392, 1981.

31. W. Duechting, "Tumor growth simulation," *Comput. Graphics*, 14: 505–508, 1990.
32. G. Stamatakos, N. Uzunoglu, K. Delibasis, M. Makropoulou, N. Mouravliansky, and A. Marsh, "A simplified simulation model and virtual reality visualization of tumor growth in vitro," *Future Generation Computer Systems*, 14: 79–89, 1998.
33. G. S. Stamatakos, N. K. Uzunoglu, K. Delibasis, M. Makropoulou, N. Mouravliansky, and A. Marsh, "Coupling parallel computing and the WWW to visualize a simplified simulation of tumor growth in vitro," in H. R. Arabnia (Ed.), *Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications, PDTA'98*, CSREA Press, Las Vegas, Nevada, 1998, pp. 526–533.
34. G. Stamatakos, N. Uzunoglu, K. Delibasis, N. Mouravliansky, M. Makropoulou, and A. Marsh, "Using VRML in a client-server architecture to visualize a simplified simulation model of tumor growth in vitro," in *Proceedings of the Twentieth Annual International Conference-IEEE/EMBS*, Hong Kong, 1998, pp. 2834–2837.
35. G. Stamatakos, E. Zacharaki, N. Mouravliansky, K. Delibasis, K. Nikita, N. Uzunoglu, and A. Marsh, "Using Web technologies and meta-computing to visualize a simplified simulation model of tumor growth in vitro," in P. Sloot, M. Bubak, A. Hoekstra, and B. Hertzberger (Eds.), *High-Performance Computing and Networking*, Lecture Notes in Computer Science, Vol. 1593, Springer, Berlin, 1999, pp. 973–982.
36. G. Stamatakos, E. Zacharaki, M. Makropoulou, N. Mouravliansky, K. Nikita, and N. Uzunoglu, "Tumour growth in vitro and tumor response to irradiation schemes: A simulation model and virtual reality visualization," *Radiother. Oncol.*, 56(1, Suppl.): 179–180, 2000.
37. G. S. Stamatakos, N. K. Uzunoglu, K. Delibasis, N. Mouravliansky, A. Marsh, and M. Makropoulou, "Tumor growth simulation and visualization: A review and a Web based paradigm," *Stud. Health Technol. Informatics*, 79: 255–274, 2000.
38. T. S. Deisboek, M. E. Berens, A. R. Kansal, S. Torquato, A. O. Stemmer-Rachmamimov, and E. A. Chiocca, "Pattern of self-organization in tumor systems: Complex growth dynamics in a novel brain tumor spheroid model," *Cell Prolif.*, 34: 115–134, 2001.
39. R. Wasserman and R. Acharya, "A patient-specific in vivo tumor model," *Math. Biosci.*, 136: 111–140, 1996.
40. A. R. Kansal, S. Torquato, G. R. Harsh, E. A. Chiocca, and T. S. Deisboeck, "Simulated brain tumor growth dynamics using a three-dimensional cellular automaton," *J. Theor. Biol.*, 203: 367–382, 2000.
41. A. R. Kansal, S. Torquato, G. R. Harsh IV, E. A. Chiocca, and T. S. Deisboeck, "Cellular automaton of idealized brain tumor growth dynamics," *BioSystems*, 55: 119–127, 2000.
42. G. Stamatakos, D. Dionysiou, N. Mouravliansky, K. Nikita, G. Pissakas, P. Georgolopoulou, and N. Uuznoglu, "Algorithmic description of the biological activity of a solid tumour in vivo," in *Proceedings of the EUROSIM 2001 Congress*, Delft, The Netherlands, June 26–29, 2001 (CD-ROM Edition).
43. L. Cohen, *Biophysical Models in Radiation Oncology*, CRC Press, Boca Raton, FL, 1983.
44. W. Sinclair and R. Morton, "X-ray and ultraviolet sensitivity of synchronized Chinese hamster cells at various stages of the cell cycle," *Biophys. J.*, 5: 1–25, 1965.
45. C. J. Gillespie, J. D. Chapman, A. P. Reuvers, and D. L. Dugle, "The inactivation of Chinese hamster cells by X rays: Synchronized and exponential cell populations," *Radiat. Res.*, 64: 353–364, 1975.

46. H. D. Thames, Jr., H. R. Withers, L. J. Peters, and G. H. Fletcher, "Changes in early and late radiation responses with altered dose fractionation: Implications for dose–survival relationships," *Int. J. Radiat. Oncol. Biol. Phys.*, 8: 219–226, 1982.
47. R. G. Dale, "The application of the linear-quadratic dose-effect equation to fractionated and protracted radiotherapy," *Br. J. Radiol.*, 58: 515–528, 1985.
48. H. D. Thames, "An 'incomplete-repair' model for survival after fractionated and continuous irradiations," *Int. J. Radiat. Biol.*, 47: 319–339, 1985.
49. H. D. Thames, M. E. Rozell, S. L. Tucker, K. K. Ang, D. R. Fisher, and E. L. Travis, "Direct analysis of quantal radiation response data," *Int. J. Radiat. Oncol. Biol. Phys.*, 49: 999–1009, 1986.
50. J. Denekamp, "Cell kinetics and radiation biology," *Int. J. Radiat. Biol.*, 49: 357–380, 1986.
51. R. Dale, "The application of the linear-quadratic model to fractionated radiotherapy when there is incomplete normal tissue recovery between fractions, and possible implications for treatments involving multiple fractions per day," *Br. J. Radiol.*, 59: 919–927, 1986.
52. E. L. Travis and S. L. Tucker, "Isoeffect models and fractionated radiation therapy," *Int. J. Radiat. Oncol. Biol. Phys.*, 13: 283–287, 1987.
53. J. F. Fowler, "The linear-quadratic formula and progress in fractionated radiotherapy," *Br. J. Radiol.*, 62: 679–694, 1989.
54. L. E. Dillehay, "A model of cell killing by low-dose-rate radiation including repair of sublethal damage, G2 block, and cell division," *Radiat. Res.*, 124: 201–207, 1990.
55. S. L. Tucker, H. D. Thames, and J. M. G. Taylor "How well is the probability of tumor cure after fractionated irradiation described by Poisson statistics?" *Radiat. Res.*, 124: 273–282, 1990.
56. D. J. Brenner, "Track structure, lesion development and cell survival," *Radiat. Res.*, 124: S29–S37, 1990.
57. H. D. Thames, S. M. Bentzen, I. Turesson, M. Overgaard, and W. Van de Bogaert, "Time-dose factors in radiotherapy: A review of the human data," *Radiother. Oncol.*, 19: 219–235, 1990.
58. J. Chen, J. Van de Geijn, and T. Goffman, "Extra lethal damage due to residual incompletely repaired sublethal damage in hyperfractionated and continuous radiation treatments," *Med. Phys.*, 18: 488–496, 1991.
59. H. D. Thames, T. E. Schultheiss, J. H. Hendry, S. L. Tucker, B. M. Dubray, and W. A. Brock, "Can modest escalations of dose be detected as increased tumor control?" *Int. J. Radiat. Oncol. Biol. Phys.*, 22: 241–246, 1992.
60. M. Zaider and G. N. Minerbo, "A mathematical model for cell cycle progression under continuous low-dose-rate irradiation," *Radiat. Res.*, 133: 20–26, 1993.
61. H. R. Withers, "Biologic basis of radiation therapy," in C. A. Perez and L. W. Brady (Eds.), *Principles and Practice of Radiation Oncology*, 2nd ed., J. P. Lippincott, New York, 1992, pp. 64–96.
62. B. Jones and R. Dale, "Mathematical models of tumor and normal tissue response," *Acta Oncol.*, 38: 883–893, 1999.
63. T. E. Wheldon, A. S. Michalowski, and J. Kirk, "The effect of irradiation on function in self-renewing normal tissues with differing proliferative organisation," *Br. J. Radiol.*, 55: 759–766, 1982.

64. D. A. Haas-Kogan, G. Yount, M. Haas, D. Levi, S. S. Kogan, L. Hu, C. Vidair, D. F. Deen, W. C. Dewey, and M. A. Israel, "p53-dependent G_1 arrest and p53 independent apoptosis influence the radiobiologic response of glioblastoma," *Int. J. Radiat. Oncol. Biol. Phys.*, 36(1): 95–103, 1996.

65. J. C. Panetta, "A mathematical model of periodically pulsed chemotherapy: Tumor recurrence and metastasis in a competitive environment," *Bull. Mat. Biol.*, 58: 425–447, 1996.

66. D. A. Cameron, W. M. Gregory, A. Bowman, and R. C. F. Leonard, "Mathematical modelling of tumor response in primary breast cancer," *Br. J. Cancer*, 73: 1409–1416, 1996.

67. T. Ginsberg, "Modellierung und Simulation der Proliferationsregulation und Strahlentherapie normaler und maligner Gewebe," Fortschritt-Berichte, VDI Verlag, Reihe 17:Biotechnik, Nr. 140, Dusseldorf, 1996.

68. A. Nahum and B. Sanchez-Nieto, "Tumour control probability modelling: Basic principles and applications in treatment planning," *Phys. Med.*, 17(xvii, Suppl. 2): 13–23, 2001.

69. W. Duechting, W. Ulmer, R. Lehrig, T. Ginsberg, and E. Dedeleit, "Computer simulation and modelling of tumor spheroid growth and their relevance for optimization of fractionated radiotherapy," *Strahlenther. Onkol.*, 168(6): 354–360, 1992.

70. W. Duechting, T. Ginsberg, and W. Ulmer, "Modeling of radiogenic responses induced by fractionated irradiation in malignant and normal tissue," *Stem Cells*, 13(Suppl. 1): 301–306, 1995.

71. G. Stamatakos, E. Zacharaki, M. Makropoulou, N. Mouravliansky, A. Marsh, K. Nikita, and N. Uzunoglu, "Modeling tumor growth and irradiation response in vitro—a combination of high-performance computing and web based technologies including VRML visualization," *IEEE Trans. Inform. Technol. Biomed.*, 5(4): 279–289, 2001.

72. G. Stamatakos, E. I. Zacharaki, N. K. Uzunoglu, and K. S. Nikita, "Tumor growth and response to irradiation in vitro: A technologically advanced simulation model," *Int. J. Radiat. Oncol. Biol. Phys.*, 51(Suppl. 1): 240–241, 2001.

73. G. Stamatakos, D. Dionysiou, K. Nikita, N. Zamboglou, D. Baltas, G. Pissakas, and N. K. Uzunoglu, "In vivo tumor growth and response to radiation therapy: A novel algorithmic description," *Int. J. Radiat. Oncol. Biol. Phys.*, 51(3, Suppl. 1): 240, 2001.

74. D. Dionysiou, G. Stamatakos, N. Uzunoglu, and K. Nikita, "Simulation of solid tumor growth in vivo and tumor response to radiation therapy," Abstracts of the Fifth International Workshop on Mathematical Methods in Scattering Theory and Biomedical Technology, Corfu, Greece, October 18–19, 2001.

75. G. Stamatakos, D. Dionysiou, E. Zacharaki, N. Mouravliansky, K. Nikita, and N. Uzunoglu, "*In silico* radiation oncology: Combining novel simulation algorithms with current visualization techniques," *Proc. IEEE*, 90(11): 1764–1777, Special Issue on "Bioinformatics: Advances and Challenges," November 2002.

76. M. Kocher and H. Treuer, "Reoxygenation of hypoxic cells by tumor shrinkage during irradiation. A computer simulation," *Strahlenther. Onkol.*, 171: 219–230, 1995.

77. M. Kocher, H. Treuer, J. Voges, M. Hoevels, V. Sturm, and R. P. Mueller, "Computer simulation of cytotoxic and vascular effects of radiosurgery in solid and necrotic brain metastases," *Radiother. Oncol.*, 54: 149–156, 2000.

78. W. Duechting, R. Lehring, G. Rademacher, and W. Ulmer, "Computer simulation of clinical irradiation schemes applied to in vitro tumor spheroids," *Strahlenther. Onkol.*, 165: 873–878, 1989.

79. B. Grant, "Virtual reality gives medicine a powerful new tool," *Biophoton. Int.*, November/December 1997, pp. 40–45.

80. W. Lorensen and H. Cline, "Marching cubes: High resolution 3D surface construction algorithm," *Comput. Graph.*, 21(3): 163–169, 1987.

81. B. S. Kuszyk, D. R. Ney, and E. K. Fishman, "The current state of the art in three dimensional oncologic imaging: An overview," *Int. J. Radiat. Oncol. Biol. Phys.*, 33: 1029–1039, 1995.

82. W. Duechting, "Computermodelle zur Optimierung der Strahlentherapie," *Automatisierungstechnik*, 46(11): 546–552, 1998.

83. B. Jones and C. Bleasdale, "Influence of tumour regression and clonogen repopulation on tumour control by brachytherapy," in K. Baier and D. Baltas (Eds.), *Modelling in Clinical Radiobiology*, Freiburg Oncology Series, Monograph No. 2, Albert-Ludwigs University, Freiburg, Germany, 1997, pp. 116–126.

84. T. Williams and R. Bjerknes, "Stochastic model for abnormal clone spread through epithelial basal layer," *Nature*, 236: 19–21, 1972.

85. D. D. Dionysiou, N. A. Mouravliansky, G. S. Stamatakos, N. K. Uzunoglu, and N. K. Nikita, "Coupling in silico biology with image processing: New perspectives in tumor growth simulation," in M. H. Hamza, *Proceedings of the Third IAESTED International Conference on Visualization, Imaging, and Image Processing*, Acta Press, Anaheim, Calgary, Zurich, 2003, pp. 488–493.

86. D. D. Dionysiou, G. S. Stamatakos, N. K. Uzunoglu, K. S. Nikita, and A. Maroli, "After dimensional simulation model of tumor response to radiotherapy involves parametric validation considering radiosensitivity gene profile," *J. Theor. Biol.*, 23: 1–22, 2004.

87. W. Duechting, W. Ulmer, and T. Ginsberg, "Modelling of tumor growth and irradiation," in *Proceedings of the Tenth International Conference on the Use of Computers in Radiation Therapy*, Manchester, U.K., March 20–24, 1994.

88. D. A. Haas-Kogan, S. S. Kogan, G. L. Yount, J. Hsu, J. M. Haas, D. F. Deen, and M. A. Israel, "p53 function influences the effect of fractionated radiotherapy on glioblastoma tumors," *Int. J. Radiat. Oncol. Biol. Phys.*, 43: 399–403, 1999.

89. B. Hegedues, A. Czirok, I. Fazekas, T. Babel, E. Madarasz, and T. Viscsek, "Locomotion and proliferation of glioblastoma cells in vitro: Statistical evaluation of videomicroscopic observations," *J. Neurosurg.*, 92: 428–434, 2000.

90. B. G. Katzung, *Basic and Clinical Pharmacology*, 8th ed., Lange Medical Books/McGraw-Hill, Las Vegas, Nevada, USA, 2001.

91. F. Giangaspero, C. Doglioni, M. T. Rivano, S. Pileri, J. Gerdes, and H. Stein, "Growth fraction in human brain tumors defoned by the monoclonal antibody Ki-67," *Acta Neuropathol.*, 74(2): 179–182, 1987.

92. P. Huang, A. Allam, L. Perez, A. Taghian, J. Freeman, and H. Suit, "The effect of combining recombinant human tumor necrosis factor-alpha with local radiation on tumor control probability of a human glioblastoma multiforme xenograft in nude mice," *Int. J. Radiat. Oncol. Biol. Phys.*, 32(1): 93–98, 1995.

93. M. Nakajima, S. Nakasu, S. Morikawa, and T. Inubushi, "Estimation of volume doubling time and cell loss in an experimental rat glioma model in vivo," *Acta Neurochir.*, 140: 607–613, 1998.

94. S. Tribius, A. Pidel, and D. Casper, "ATM protein expression correlates with radioresistance in primary glioblastoma cells in culture," *Int. J. Radiat. Oncol. Biol. Phys*., 50: 511–523, 2001.

95. G. Shu, M. Kim, P. Chen, F. Furman, C. Julin, and M. Israel, "The intrinsic radioresistance of glioblastoma-derived cell lines is associated with a failure of p53 to induce $p21^{BAX}$ expression," *Proceedings of the National Academy of Sciences*, (*PNAS*), 95(24): 14453–14458, 1998.

CHAPTER 6

Genomewide Motif Identification Using a Dictionary Model

CHIARA SABATTI and KENNETH LANGE

6.1. INTRODUCTION

Computational genomics has many different goals and profits from many different scientific perspectives. One obvious goal is to find all of the genes within a genome and how they operate. This task is complicated by the segmentation of genes into exons and introns. After a gene is transcribed into mRNA, its introns are spliced out of the message. Many genes display alternative splicing patterns that eliminate some of the underlying exons as well. Regulatory regions upstream of a gene determine when and in what tissues a gene is transcribed. A second goal of genomics is to use the amino acid content of each message to deduce the structure and function of the encoded protein. A third goal is to understand how genes and gene products interact in space and time. Each of these goals benefits from the pattern recognition principles widely used in computer science and statistics. At the same time, the peculiarities of genetics demand special techniques in addition to general methods. Because the information housed in a genome is written in a distinct language, it is tempting to transfer ideas from mathematical linguistics to genomics. In our view, such a transfer is apt to be more successful for semantics than for grammar. This chapter surveys and develops a dictionary model for locating binding sites in regulatory regions. In the dictionary model, a DNA sequence is viewed as a random concatenation of words with alternative spellings.

6.1.1. Biological Problem

Deoxyribonucleic acid, the molecule that encodes genetic information, is a long polymer whose structure can be effectively be described by a sequence of letters of four types—A, C, G, and T—corresponding to the four nucleotides (or bases) adenine, cytosine, guanine, and thymine. The vast majority of human DNA is

Genomics and Proteomics Engineering in Medicine and Biology. Edited by Metin Akay

organized into 46 linear chromosomes stored in the cell nucleus. Except for the X and Y sex chromosomes, the remaining 44 chromosomes come in 22 pairs of nearly identical homologous chromosomes. The total length of the 22 consensus autosomes and the two sex chromosomes is approximately three billion bases. By comparison, the genome of the bacterium *Escherichia coli* consists of a single circular strand five and a half million bases long. In the past decade, the complete genomes of hundreds of organisms have been sequenced, and last year a rough draft of the human genome was announced [1, 2]. These remarkable achievements make it possible to undertake whole-genome analysis and compare genomes of different species.

In eukaryotes, the higher organisms with a cell nucleus, genes occupy only a small fraction of the total genome. For example, in humans, recent estimates suggest that coding DNA amounts to only 1.5% of the genome. The function of the remaining portion of DNA is not entirely understood, but it is clear that it plays an important role in evolution and in the regulation of gene expression. In this chapter, we focus on noncoding DNA, in particular on regions immediately upstream of genes. These regions are often involved in regulation of transcription, the process of copying genes in preparation for their translation into proteins. In order for the transcription machinery to operate on a given gene at a given time, regulatory proteins typically must bind or unbind to specific locations upstream of the gene. Most organisms possess multiple interacting regulatory proteins, and each regulatory protein typically influences the expression of many genes. Thus, one can expect to find far fewer regulatory proteins than genes. For example, *E. coli* has about 4200 genes and only about 100 major regulatory proteins.

In this conceptual framework, each regulatory protein recognizes and binds to a series of DNA locations. These locations share a common sequence pattern that is specific to the protein. Because of the variation in different realizations of the same pattern, geneticists have adopted the term *motif* rather than *pattern*. This is consistent with usage in the visual arts, where motif refers to a virtual archetype that can be rendered in a variety of different ways. Figure 6.1 presents some experimentally identified binding sites for CRP, a regulatory protein of major importance

```
attcgtgatagctgtcgtaaag
ttttgttacctgcctctaactt
aagtgtgacgccgtgcaaataa
tgccgtgattatagacactttt
atttgcgatgcgtcgcgcattt
taatgagattcagatcacatat
taatgtgacgtcctttgcatac
gaaggcgacctgggtcatgctg
aggtgttaaattgatcacgttt
cgatgcgaggcggatcgaaaaa
aaattcaatattcatcacactt
```

FIGURE 6.1. Experimentally identified binding sites for CRP mentioned at the website http://arep.med.harvard.edu/ecoli_matrices/. Each row represents one binding site of length 22.

in *E. coli*. This example clearly illustrates both the constancy and variation among realizations of the same DNA motif. All realizations span 22 bases. Although experimentation is the definitive way of identifying and characterizing binding site motifs, geneticists are keenly interested in less labor intensive methods. For that reason, bioinformatics approaches have blossomed. These are the themes of this chapter.

6.1.2. Previous Methods of Motif Recognition

As promised, we now briefly review three different approaches for identifying binding sites in DNA. Although this overview is hardly exhaustive, it does demonstrate the steady evolution of the models toward greater complexity and biological realism.

In 1990 Lawrence and Reilly [3] proposed a successful motif model in which the binding sites for a regulatory protein are assumed to have a constant length k. While this assumption is not always true, it is the rule because the usual lock-and-key argument of molecular biology requires all binding domains to fit into the same physical portion of the regulatory protein. At each motif position i, any of the four letters A, C, G, and T may occur. The relative frequencies of occurrence are described by a distribution $\ell_i = (\ell_{iA}, \ell_{iC}, \ell_{iG}, \ell_{iT})$ specific to position i. The letters appearing at different positions are independent. In statistical language, a motif is distributed as a product of multinomials. Motifs are contrasted to "background" sequence, where letters are chosen independently from a common distribution $\ell_0 = (\ell_{0A}, \ell_{0C}, \ell_{0G}, \ell_{0T})$. In a typical data set, each observed upstream sequence is assumed to harbor a single instance of the motif, but its exact location is unknown. Lawrence and Reilly [3] turned this missing-data feature to their advantage and devised an expectation–minimization (EM) algorithm for estimating both the parameter vectors $\ell_i, i = 1, \ldots, k$, and the locations of the motif within each upstream sequence. Later Lawrence et al. [4] elaborated a Bayesian version of the model and applied Gibbs sampling to estimate parameters and motif locations. Their Gibbs algorithm can be run on the Internet at the site http://www.bayesbio.html.

A different type of input data motivated the research of Robison et al. [5]. Instead of starting with a small set of sequences known to harbor the same unknown motif, they considered the entire genome of *E. coli* relative to a collection of experimentally identified binding sites involving 55 regulatory proteins. Their goal was to identify all of the other binding sites for these proteins. The computational strategy in [5] is nonparametric and heuristic. A scoring function is defined for each motif. The mean m and variance v of the score values from a set of experimentally certain binding sites are recorded. The scoring function is then evaluated at each genome position, and the locations that lead to a score higher than $m - 2\sqrt{v}$ are considered putative binding sites for the protein under study. Results of this study can be viewed at http://arep.med.harvard.edu/ecoli_matrices/. The most appealing feature of the Robison et al. approach is its genomewide nature. One of its least appealing features is its relatively uninformative description of the binding site.

Bussemaker et al. [6] propose a third and very different approach to motif recognition. In their model, DNA sequence data are viewed as a concatenation of different words, each word randomly selected from a dictionary with specified probabilities. Words of length 1 play substantially the same role as background sequence in [3]. Longer words may represent binding sites. Bussemaker et al. [6, 7] describe algorithms that estimate the probabilities of all of the words in a fixed dictionary and sequentially build a dictionary from data. Their algorithms have been tested on the first 10 chapters of the novel *Moby Dick* with all punctuation signs and blanks between words removed. The results are encouraging, though occasionally identified words are concatenations of two English words. A similar approach can be applied to DNA to identify regulatory sites. One defect of the model is its dubious assumption that each word has a unique spelling. If we take misspellings into account, then constructing a dictionary from scratch appears overly ambitious, particularly with a four-letter alphabet.

In the rest of this chapter, we develop a model that borrows some elements from all the above approaches: (a) our description of a motif substantially coincides with that in [3]; (b) in common with [5], we seek to identify the binding sites of a predetermined set of regulatory proteins for which some experimental evidence exists; and (c) we use a likelihood description for DNA similar to that in [6]. Note that databases such as the TRANSFAC database at http://transfac.gbf.de/TRANSFAC/ warehouse sequence information on experimentally identified binding sites for a variety of proteins across many organisms.

6.2. UNIFIED MODEL

The model we propose describes a DNA sequence as a concatenation of words, each independently selected from a dictionary according to a specific probability distribution. For us, a word is simply an irreducible semantic unit or, in the genetic context, a motif. Each word may have more than one spelling. Thus, in English, *theater* and *theatre* represent the same word. Two different words may share a spelling. For instance, *pot* may refer either to a cooking utensil or something to smoke.

In our model, a word w always has the same number of letters $|w|$. Hence, alternative spellings such as *night* and *nite* with different number of letters are disallowed. For reasons that will soon be apparent, it is convenient to group words according to their lengths and to impose a maximum word length $k_{\max}$ on our dictionary. It may be that no words of a given length $k \leq k_{\max}$ exist. For example, in the Lawrence et al. model [3] for the CPR binding site, only words of length 1 and length 22 appear. A random sequence S is constructed from left to right by concatenating random words, with each word and each spelling selected independently. The letters of a word are independently sampled from different multinomial distributions. This is known as product multinomial sampling.

In summary, our DNA model requires a static dictionary with a list of alternative spellings and probability distributions determining which words and spellings are selected. The parameters describing the model are as follows:

1. The probability of choosing a word of length k is q_k. Here k ranges from 1 to $k_{\max}$, and $\sum_{k=1}^{k_{\max}} q_k = 1$. If there are no words of length k, then $q_k = 0$.
2. Conditional on choosing a word of length k, a particular word w with $|w| = k$ is selected with probability r_w. Hence, $\sum_{|w|=k} r_w = 1$.
3. The letters of a word w follow a product multinomial distribution with success probabilities

$$\ell_{wi} = (\ell_{wiA}, \ell_{wiC}, \ell_{wiG}, \ell_{wiT})$$

for the letters A, C, T, and G at position i of w.

A randomly chosen word of length k exhibits the spelling $s = (s_1, \ldots, s_k)$ with probability

$$p(s) = \sum_{|w|=k} r_w \prod_{i=1}^{k} \ell_{wis_i} \tag{6.1}$$

If some letters are missing, for instance when sequencing quality is poor, then formula (6.1) fails. To force its validity in the presence of missing data, we represent missing letters by question marks and introduce the additional letter probability $\ell_{wi?} = 1$ for each word w and position i within w. This missing-letter convention will be used later to describe the probability of partially observed words that overlap the edges of a sequence.

An observed sequence generally contains more than one word, with unknown boundaries separating the words. Missing-word boundaries are more vexing than missing letters. We will call the portion of a sequence between two consecutive word boundaries a *segment* and the set of word boundaries dividing a sequence an *ordered partition* of the sequence. For theoretical purposes, the probability of a sequence is best evaluated by conditioning on its ordered partition and then averaging the resulting conditional probability over all partitions. In numerical practice, we implement this strategy recursively via forward and backward algorithms similar to those used with hidden Markov chains.

We consider two stochastic models for generating a random sequence S by concatenating words. These models differ in how they treat edge effects. The model proposed by Bussemaker et al. [6], which we will call as full-text model, assumes that a sequence starts and ends with full words. This is reasonable if the sequence represents a DNA strand in its entirety or the sequence coincides with a well-delimited and biologically meaningful region such as an exon. We propose an alternative model, which we call the equilibrium model, in which the first (or last) letter of an observed sequence need not be the first (or last) letter of a word.

In this model we observe a random fragment of text from an infinitely long sequence. The equilibrium model is more realistic for randomly selected DNA sequences of predetermined length.

To describe the probability of an observed sequence s under these two models, we now introduce some necessary index notation. A vector of consecutive indices

$$\sigma = (i,\ i+1, \ldots,\ j-1,\ j) = (i:j)$$

is called a compatible block if its length $|\sigma| = j - i + 1$ does not exceed the maximum word length $k_{\max}$. An ordered partition π of a sequence s divides the indices of s into a vector of compatible blocks $\pi = (\pi_1, \ldots, \pi_{|\pi|})$ subject to two conditions. Condition 1 applies to both models and says that if block π_i ends with index j, then block π_{i+1} begins with index $j+1$. Condition 2a applies only to the full-text model and requires the first block π_1 to begin with index 1 and the last block $\pi_{|\pi|}$ to end with the last index $|s|$ of s. Condition 2b applies only to the equilibrium model and requires the first block π_1 merely to contain index 1 and the last block $\pi_{|\pi|}$ merely to contain the last index $|s|$ of s. Each block π_i of π determines a segment $s[\pi_i]$ of s.

For instance, the ordered partition π composed of the blocks $\pi_1 = (1, 2)$, $\pi_2 = (3, 4, 5)$, and $\pi_3 = (6)$ divides the sequence $(s_1, \ldots, s_6)$ into the three segments

$$\begin{aligned} s[\pi_1] &= (s_1, s_2) \\ s[\pi_2] &= (s_3, s_4, s_5) \\ s[\pi_3] &= (s_6) \end{aligned}$$

This particular partition is consistent with both models. The collection $\mathcal{F}$ of partitions compatible with the full-text model is smaller than the collection $\mathcal{E}$ of partitions compatible with the equilibrium model. For example, the ordered partition $\pi \in \mathcal{E} \setminus \mathcal{F}$ with blocks $\pi_1 = (-1,\ 0,\ 1,\ 2)$, $\pi_2 = (3,\ 4,\ 5)$, and $\pi_3 = (6,\ 7)$ divides the sequence $(s_1, \ldots, s_6)$ into the three segments

$$\begin{aligned} s[\pi_1] &= (s_{-1}, s_0, s_1, s_2) = (?,\ ?,\ s_1,\ s_2) \\ s[\pi_2] &= (s_3,\ s_4,\ s_5) \\ s[\pi_3] &= (s_6,\ s_7) = (s_6,\ ?) \end{aligned}$$

Here we have padded s with missing letters on its left and right ends. In general, the constraints $\sum_{i=1}^{|\pi|} |\pi_i| = |s|$ for $\pi \in \mathcal{F}$ and $\sum_{i=1}^{|\pi|} |\pi_i| \geq |s|$ for $\pi \in \mathcal{E}$ must be imposed on the sum of the segment lengths.

We now derive the likelihood of a sequence s under the full-text model. Let F be the event that randomly concatenating words give a sequence with a word boundary at position $|s|$. Because the probability of a partition $\pi \in \mathcal{F}$ is proportional to the product of the probabilities of the lengths of the segments

constituting it, we have

$$\Pr(\pi \mid F) = \frac{\prod_{i=1}^{|\pi|} q_{|\pi_i|}}{\sum_{\pi \in \mathcal{F}} \prod_{i=1}^{|\pi|} q_{|\pi_i|}}$$

The normalizing constant here is difficult to evaluate analytically, but it can be rewritten as

$$\Pr(F) = \sum_{\pi \in \mathcal{F}} \prod_{i=1}^{|\pi|} q_{|\pi_i|} = \sum_{m \in \mathcal{M}} \binom{m_1 + \cdots + m_{k_{\max}}}{m_1, \ldots, m_{k_{\max}}} \prod_{k=1}^{k_{\max}} q_k^{m_k}$$

where $\mathcal{M}$ denotes the set of vectors $m = (m_1, \ldots, m_{k_{\max}})$ of nonnegative integers such that $\sum_{k=1}^{k_{\max}} k m_k = |s|$. Here m_k is the number of blocks of length k. The likelihood of the sequence under the full-text model boils down to

$$\begin{aligned}
\mathcal{L}_F(s) &= \Pr(S = s \mid F) \\
&= \frac{\sum_{\pi \in \mathcal{F}} \prod_{i=1}^{|\pi|} q_{|\pi_i|} \Pr(s[\pi_i] \mid \pi)}{\sum_{\pi \in \mathcal{F}} \prod_{i=1}^{|\pi|} q_{|\pi_i|}} \\
&= \frac{\sum_{\pi \in \mathcal{F}} \prod_{i=1}^{|\pi|} q_{|\pi_i|} p(s[\pi_i])}{\sum_{\pi \in \mathcal{F}} \prod_{i=1}^{|\pi|} q_{|\pi_i|}}
\end{aligned}$$

Bussemaker et al. [7] give an algorithm for computing the numerator of this likelihood but none for computing the denominator $\Pr(F)$. They assert that it is sufficiently close to 1 for practical purposes. While this may be true in their specific context, we have observed substantial variation in $\Pr(F)$ as a function of $q = (q_1, \ldots, q_{k_{\max}})$. For example, for a dictionary containing only words of length 1 and 10 and a sequence of 800 bases, $\Pr(F)$ varies between 1 and 0.02. This makes us uncomfortable in equating it to 1. Later we will derive an efficient algorithm for computing the value of $\Pr(F)$.

Over the enormous stretches of DNA seen in all genomes, it is reasonable to suppose that the process of concatenating words has reached equilibrium at the start of any small sequence s. The equilibrium model makes it possible to assign a probability to the first segment generated by a partition $\pi \in \mathcal{E}$ covering s. Indeed, the probability that a randomly chosen position along the genome is covered by a word of length j is the ratio $jq_j/\bar{q}$, where $\bar{q} = \sum_{k=1}^{k_{\max}} k q_k$ denotes the length of an average word. In particular, the probability $jq_j/\bar{q}$ applies to position 1 of s. The conditional probability that position 1 of s coincides with a particular position of a covering word of length j is $1/j$. It follows that the jth index π_{1j} of π_1 covers position 1 of s with probability $q_{|\pi_1|}/\bar{q}$. Similar considerations apply to the last block of π if we consider concatenating words from right to left rather than from left to right. In either case, we can express the probability of $\pi \in \mathcal{E}$ under

the event E of equilibrium as

$$\Pr(\pi|E) = \frac{\prod_{i=1}^{|\pi|} q_{|\pi_i|}}{\overline{q}}$$

It is a relatively simple exercise to check that $\sum_{\pi\in\mathcal{E}} \Pr(\pi \mid E) = 1$.

For readers dissatisfied with this intuitive explanation of equilibrium, it may help to consider a Markov chain on an infinite sequence of letters constructed by randomly concatenating words. The state of the chain X_n at position n of the sequence is a pair of integers (i, j) with $1 \le i \le j \le k_{\max}$. The integer j gives the length of the word covering position n, and the integer i gives the position of n within that word. The actual letter at n is irrelevant. It is easy to prove that this finite-state chain is irreducible and, provided there is at least one single-letter word, aperiodic. Let λ_{nij} be the probability that the chain occupies state (i, j) at position n. Elementary reasoning yields the one-step recurrence

$$\lambda_{nij} = 1_{\{i>1\}}\lambda_{n-1,i-1,j} + 1_{\{i=1\}}\sum_{k=1}^{k_{\max}} \lambda_{n-1,kk}q_j$$

and standard theory for a Markov chain says that the limits $\lim_{n\to\infty} \lambda_{nij} = \lambda_{ij}$ exist and do not depend on the initial distribution of the chain. Because the probability distribution $\lambda_{ij} = q_j/\overline{q}$ obviously satisfies the one-step recurrence, this validates our claimed equilibrium model.

By allowing missing letters and partitions that straddle the ends of s, we can write the likelihood of s under the equilibrium model as

$$\begin{aligned}\mathcal{L}_E(s) &= \Pr(S = s \mid E)\\ &= \frac{1}{\overline{q}}\sum_{\pi\in\mathcal{E}}\prod_{i=1}^{|\pi|} q_{|\pi_i|}p(s[\pi_i])\end{aligned}$$

Again, this formula is ill adapted to computing. It is noteworthy, however, that the normalizing constant is vastly simpler. Furthermore, the likelihood under the full-text model can be viewed as a conditional probability in the equilibrium model in the sense that $\mathcal{L}_F(s) = \Pr(S = s \mid E,\ F)$.

6.3. ALGORITHMS FOR LIKELIHOOD EVALUATION

Our likelihood algorithms resemble Baum's forward and backward algorithms from the theory of hidden Markov chains [8, 9]. For the sake of simplicity, we first consider the full-text likelihood of s. Let B_i be the event that a word ends at position i. The forward algorithm updates the joint probabilities

$$f_i = \Pr(S[1\colon i] = s[1\colon i],\ B_i)$$

and the backward algorithm updates the conditional probabilities

$$b_i = \Pr(S[i:n] = s[i:n] | B_{i-1})$$

for $n = |s|$.

The forward algorithm initializes $f_0 = 1$ and iterates according to

$$f_i = \sum_{k=1}^{\min\{k_{\max}, i\}} f_{i-k}\, q_k p(s[i-k+1:i])$$

in the order $i = 1, \ldots, n$. At the last step, f_n equals the numerator of $\mathcal{L}_F(s)$, that is, $\sum_{\pi \in \mathcal{F}} \prod_{i=1}^{|\pi|} q_{|\pi_i|} \Pr(s[\pi_i] \,|\, \pi)$. The forward algorithm for computing the denominator is similar except that it iterates via

$$f_i = \sum_{k=1}^{\min\{k_{\max}, i\}} f_{i-k} q_k$$

ignoring the letter content of the sequence. The backward algorithm begins with $b_{n+1} = 1$ and updates

$$b_i = \sum_{k=1}^{\min\{k_{\max}, n+1-i\}} b_{i+k} q_k p(s[i:\, i+k-1])$$

in the reverse order $i = n, \ldots, 1$. At the last step, we recover the numerator of $\mathcal{L}_F(s)$ as b_1. Finally, the backward algorithm for the denominator iterates via

$$b_i = \sum_{k=1}^{\min\{k_{\max}, n+1-i\}} b_{i+k} q_k$$

To derive these updates, we simply concatenate an additional segment to one of the current partial sequences, assuming that the entire sequence starts and ends with full words. Bussemaker et al. [6, 7] give the backward and forward algorithms for the numerator but omit the algorithms for the denominator of $\mathcal{L}_F(s)$.

The forward and backward algorithms for the equilibrium likelihood are similar but more complicated. The forward algorithm commences with $f_i = 1/\bar{q}$ for $i = 1 - k_{\max}, \ldots, 0$. This expanded set of initial values reflects the variety of starting points for segments containing position 1. The remaining joint probabilities are determined by

$$f_i = \sum_{k=\max\{1,\, i+1-n\}}^{k_{\max}} f_{i-k} q_k p(s[i-k+1{:}i])$$

for $i = 1, \ldots, n + k_{\max} - 1$. This is precisely the update used for the numerator of the full-text likelihood when $i \leq n$. When $i > n$, the requirement that the last word must contain position n limits the range of summation of k to $i - k < n$. The sum $\mathcal{L}_E(s) = f_n + \cdots + f_{n+k_{\max}-1}$ defines the equilibrium likelihood. The backward algorithm begins with $b_i = 1$ for $i = n+1, \ldots, n+k_{\max}$ and iterates according to

$$b_i = \sum_{k=\max\{1,\,2-i\}}^{k_{\max}} b_{i+k} q_k p(s[i:i+k-1])$$

for $i = n, \ldots, 2 - k_{\max}$. In this case, the equilibrium likelihood $\mathcal{L}_E(s) = (b_{2-k_{\max}} + \cdots + b_1)/\bar{q}$.

As a trivial example, consider $s = (s_1)$ and $k_{\max} = 2$. Then the updates

$$\begin{aligned}
f_1 &= f_{-1} q_2 \sum_{|w|=2} r_w \ell_{w2s_1} + f_0 q_1 \sum_{|w|=1} r_w \ell_{w1s_1} \\
f_2 &= f_0 q_2 \sum_{|w|=2} r_w \ell_{w1s_1} \\
b_1 &= b_2 q_1 \sum_{|w|=1} r_w \ell_{w1s_1} + b_3 q_2 \sum_{|w|=2} r_w \ell_{w1s_1} \\
b_0 &= b_2 q_2 \sum_{|w|=2} r_2 \ell_{w2s_1}
\end{aligned}$$

both lead to the equilibrium likelihood

$$\mathcal{L}_E(s) = \frac{1}{q_1 + 2q_2} \left[q_1 \sum_{|w|=1} r_w \ell_{w1s_1} + q_2 \sum_{|w|=2} r_w (\ell_{w1s_1} + \ell_{w2s_1}) \right]$$

For long sequences, one has to rescale to prevent underflows. Rescaling is a general device that applies to linear iteration. Suppose x^i is a vector sequence generated by the recurrence $x^{i+1} = M^i x^i$ for matrices M^i. In rescaling we replace this sequence by another sequence y^i starting with $y^0 = x^0$ and satisfying $y^{i+1} = c_i^{-1} M^i y^i$. The positive constant c_i is typically taken to be $\|y^i\|$ for some norm. One can easily show by induction that $x^i = (\prod_{j=0}^{i-1} c_j) y^i$. If we want the logarithm of some positive inner product $v^* x^i$, then we compute the logarithm of the positive inner product $v^* y^i$ and add the compensating sum $\sum_{j=0}^{i-1} \ln c_j$. Readers can supply the details of how this applies to computing loglikelihoods under the forward and backward algorithms.

Intermediate values from the forward and backward algorithms are stored for a variety of reasons. For instance, under the equilibrium model, we may want the conditional probability that the sequence s contains a segment extending from

index i to index j. This probability can be expressed as

$$\kappa_{ij} = \frac{f_{i-1} q_{j-i+1} p(s[i:j])\, b_{j+1}}{\mathcal{L}_E(s)} \tag{6.2}$$

The restriction that a particular word w fills this segment has conditional probability

$$\rho_{ij}(w) = \frac{f_{i-1} q_{j-i+1} r_w \prod_{k=1}^{j-i+1} \ell_{wks_{i+k-1}} b_{j+1}}{\mathcal{L}_E(s)} \tag{6.3}$$

These particular conditional probabilities are pertinent to estimation of the parameter vectors q, r, and ℓ describing the model.

6.4. PARAMETER ESTIMATION VIA MINORIZATION–MAXIMIZATION ALGORITHM

A Bayesian approach to parameter estimation is attractive because it allows the incorporation of prior information on experimentally identified binding sites. The application of a 0–1 loss function in similar classification problems suggests that we maximize the posterior density. This is proportional to the product of the prior density and the likelihood. There is no harm in selecting the prior density from a convenient functional family provided we match its parameters to available prior data. Since the presence of the prior adds little complexity to optimization of the likelihood itself, we will first discuss maximum-likelihood estimation and then indicate how it can be modified to accommodate a prior.

To maximize the complicated likelihood function $\mathcal{L}_E(s \mid q, r, \ell)$, we resort to a minorization–maximization (MM) algorithm [10]. This iterative optimization principle maximizes a target function $f(x)$ by taking a current iterate x^m and constructing a minorizing function $g(x \mid x^m)$ in the sense that $g(x \mid x^m) \leq f(x)$ for all x and $g(x^m \mid x^m) = f(x^m)$. The next iterate x^{m+1} is chosen to maximize $g(x \mid x^m)$. This choice of x^{m+1} guarantees that $f(x^{m+1}) \geq f(x^m)$. For the MM strategy to be successful, maximization of $g(x \mid x^m)$ should be easy.

The best known class of MM algorithms consists of the EM algorithms. All EM algorithms revolve around the notion of missing data. In the current setting, the missing data are the partition π segmenting the sequence and the words assigned to the different segments of s generated by π. In the expectation step of the EM algorithm, one constructs a minorizing function to the loglikelihood by taking the conditional expectation of the complete data loglikelihood with respect to the observed data. For the equilibrium model, the complete data likelihood is

$$\frac{1}{q} \prod_{i=1}^{|\pi|} q_{|\pi_i|} r_{w_i} \prod_{j=1}^{|w_i|} \ell_{w_i j s_{\pi_{ij}}}$$

where segment $s[\pi_i]$ is assigned word $w_{\bar{i}}$, and π_{ij} denotes the jth index of π_i. Let M_k be the number of segments of length k, N_w be the number of appearances of word w, and L_{wjt} be the number of letters of type t occurring at position j of the segments assigned word w. In this notation, the complete data loglikelihood is expressed as

$$\sum_{k=1}^{k_{\max}} M_k \ln q_k + \sum_w N_w \ln r_w + \sum_{w,i,j} L_{wij} \ln \ell_{wij} - \ln \overline{q}$$

The conditional expectations of the counts M_k, N_w, and L_{wij} given $S = s$ are readily evaluated as

$$E(M_k \mid S = s,\ q,\ r,\ \ell) = \sum_{i=-k+2}^{|s|} \kappa_{i,i+k-1}$$

$$E(N_w \mid S = s,\ q,\ r,\ \ell) = \sum_{i=-|w|+2}^{|s|} \rho_{i,i+|w|-1}(w)$$

$$E(L_{wjt} \mid S = s,\ q,\ r,\ \ell) = \sum_{i=-|w|+2}^{|s|} 1_{\{s_{i+j-1}=t_j\}} \rho_{i,i+|w|-1}(w)$$

using Eqs. (6.2) and (6.3).

The EM algorithm for hidden multinomial trials updates a success probability by equating it to the ratio of the expected number of successes to the expected number of trials given the observed data and the current parameter values [11]. This recipe translates into the iterates

$$r_w^{m+1} = \frac{E(N_w \mid S = s,\ q^m,\ r^m,\ \ell^m)}{E(M_{|w|} \mid S = s,\ q^m,\ r^m,\ \ell^m)}$$

$$\ell_{wjt}^{m+1} = \frac{E(L_{wjt} \mid S = s,\ q^m,\ r^m,\ \ell^m)}{E(N_w \mid S = s,\ q^m,\ r^m,\ \ell^m)}$$

Updating the segment probabilities q_k is more problematic. Because the surrogate function created by the expectation step separates the q_k parameters from the remaining parameters, it suffices to maximize the function

$$g(q|q^m) = \sum_{k=1}^{k_{\max}} E(M_k \mid S = s,\ q^m,\ r^m,\ \ell^m) \ln q_k - \ln\left(\sum_{k=1}^{k_{\max}} k q_k\right)$$

subject to the constraints $q_k \geq 0$ and $\sum_{k=1}^{k_{\max}} q_k = 1$. To our knowledge, this problem cannot be solved in closed form. It is therefore convenient to undertake a second minorization exploiting the inequality $\ln x \leq \ln y + x/y - 1$. Application of this

inequality produces the minorizing function

$$h(q \mid q^m) = \sum_{k=1}^{k_{\max}} E(M_k \mid S = s, q^m, r^m, \ell^m) \ln q_k - \ln\left(\sum_{k=1}^{k_{\max}} k q_k^m\right) - c^m \sum_{k=1}^{k_{\max}} k q_k + 1$$

with $c^m = 1/(\sum_{k=1}^{k_{\max}} k q_k^m)$.

The function $h(q \mid q^m)$ still resists exact maximization, but at least it separates the different q_k. To maximize $h(q \mid q^m)$, we employ the method of Lagrange multipliers. This entails finding a stationary point of the Lagrangian

$$h(q \mid q^m) + \lambda\left(\sum_{k=1}^{k_{\max}} q_k - 1\right)$$

Differentiating the Lagrangian with respect to q_k yields the equation

$$0 = \frac{e_k^m}{q_k} - c^m k + \lambda$$

where

$$e_k^m = E(M_k \mid S = s, q^m, r^m, \ell^m)$$

The components

$$q_k = \frac{e_k^m}{c^m k - \lambda}$$

of the stationary point involve the unknown Lagrange multiplier λ. Fortunately, λ is determined by the constraint

$$1 = \sum_{k=1}^{k_{\max}} q_k = \sum_{k=1}^{k_{\max}} \frac{e_k^m}{c^m k - \lambda}$$

The sum on the right-hand side of the second of these two equations is strictly monotone in λ, on the interval $(-\infty, c^m)$ leading to positive solutions for all q_k's. Hence, it equals 1 at exactly one point. Any of a variety of numerical methods will yield this point. In practice, we use bisection, which is easy to program and highly reliable. Its relatively slow rate of convergence is hardly noticeable amid the other more computationally intensive tasks.

We now briefly describe how slight modifications of these algorithms permit maximization of the posterior density. The general idea is to put independent priors on q, r, and ℓ. Because Dirichlet densities are conjugate priors for multinomial densities, it is convenient to choose Dirichlet priors. Therefore, consider a

Dirichlet prior

$$\frac{\Gamma(\sum_{k=1}^{k_{\max}} \alpha_k)}{\prod_{k=1}^{k_{\max}} \Gamma(\alpha_k)} \prod_{k=1}^{k} q_k^{\alpha_k - 1}$$

for q, say. In selecting the prior parameters $\alpha_1, \ldots, \alpha_{k_{\max}}$, it is helpful to imagine a prior experiment and interpret $\alpha_k - 1$ as the number of successes of type k in that experiment. In this imaginary setting, there is nothing wrong with specifying a fractional number of successes. The sum $\sum_{k=1}^{k_{\max}} \alpha_k - k_{\max}$ gives the number of trials in the prior experiment and hence determines the strength of the prior. If little or no prior information is available, then one can set all $\alpha_k = 1$. This yields a posterior density that coincides with the likelihood. Setting all $\alpha_k = 2$ regularizes estimation and deters estimates of q_k from approaching the boundary value 0.

In summary, adding a Dirichlet prior to a multinomial likelihood corresponds to adding $\alpha_k - 1$ pseudocounts to category k of the observed data. Hence, if we focus on estimating q, then in the MM algorithm just described we replace M_k by $M_k + \alpha_k - 1$. Everything else about the algorithm remains the same. Similar considerations apply to estimation of the parameter vectors r and ℓ except we deal with product multinomials rather than multinomials. This distinction entails substituting products of independent Dirichlet priors for a single Dirichlet prior.

6.5. EXAMPLES

We now consider two illustrative examples. In the first we used the data of Lawrence et al. [4] on 18 short microbial sequences to reconstruct the binding site for camp receptor protein (Crp), an important regulatory protein. This particular data set has served as a benchmark for testing many motif-finding algorithms. Each sequence is 105 bp long and is expected to harbor at least one binding site. Our goal was to estimate the word and letter probabilities for a dictionary consisting of just two words—a one-letter word representing background and a 22-letter word representing the Crp binding site. Given that the sequences are short compared to the length of the motif and that the motif occurs frequently across the sequences, we used noninformative word and letter priors and a high 0.8 cutoff posterior probability for declaring a motif. To avoid getting trapped at a local mode, we started our Fortran 95 implementation of the MM algorithm from multiple random points. Among 10 runs of the algorithm, the one with the highest logposterior shows 19 of the 24 previously noted sites [4] and an additional 23 putative sites. Our reconstructed spelling matrix corresponds well with the known matrix for Crp, even if the motif appears to be slightly less conserved than in the experimentally identified sites. Although too many false positives explain this phenomenon, we may also have detected binding sites that have lower affinity and hence are selected less frequently in reality. A depiction of the identified motif can be found in Figure 6.2 where each stack of letters corresponds to one position in the sequence. The height of a stack is proportional to the conservation of that position, and the height of the letters within a

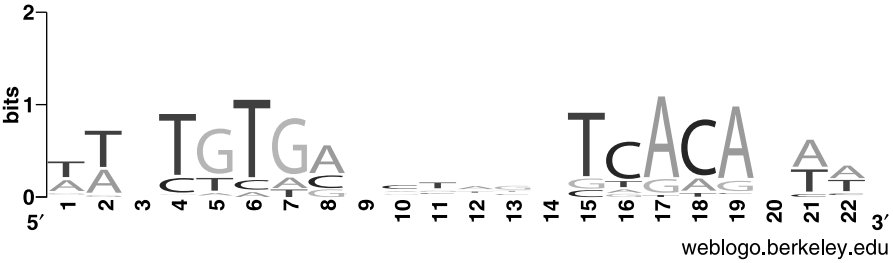


FIGURE 6.2. Profile of binding site for Crp as reconstructed starting from 18 microbial sequences. The graphics software used is available at http://weblogo.berkeley.edu/ and is based on the sequence logo described in [12].

stack is proportional to their relative frequency. The motif is clearly palindromic and positions 4–5–6–7–8 are more conserved than others, corresponding to the experimental data presented in Figure 6.1.

This is an easy example because it deals with a single motif. In our follow-up paper [13], we describe genomewide localization of the binding sites for 41 regulatory proteins in *E. coli*. Given the sparse experimental information, it impossible to check our motif predictions directly. However, comparison of the motif predictions with the results of gene expression experiments suggests that the dictionary model is working well.

6.6. DISCUSSION AND CONCLUSION

In this chapter, we have explored some of the conceptual issues involved in applying the dictionary model to motif finding. A clearer understanding of these issues is crucial in formulating algorithms that make statistical sense.

The model of Lawrence and Reilly [3] and its extensions [4] successfully identify sites in short sequences locally enriched for a given binding site. As the whole genomes of more species become available, there is a growing interest in global methods of motif identification. The work of Robison et al. [5] and Bussemaker et al. [6] is motivated by this aim. Our current synthesis points in the same direction. In the long run, comparison of homologous sequences from related species is apt to provide the best leverage in identifying binding sites [14]. Adaptation of the dictionary model to this purpose is a natural research goal.

Other more modest theoretical extensions come to mind. For example, one could search for protein motifs by substituting amino acids for bases. In noncoding regions of DNA, it might be useful to model binding site motifs that are palindromes. This puts constraints on the parameters in the product multinomial distributions for letters within a give word. The independent choice of letters in a word is also suspect. A Markov chain model might be more appropriate in some circumstances. Finally, our model assumes that consecutive words are selected independently. However, it is reasonable to posit that multiple proteins interact in regulating expression.

This assumption translates into the co-occurrence of binding sites. Co-occurrence can be investigated within the framework of the unified model by monitoring the posterior probabilities of binding sites and checking whether these tend to be cross correlated as a function of position along a sequence.

We have assumed a static dictionary. Bussemaker et al. [7, 6] tackle the problem of dictionary construction. Although their methods are elegant, it is unclear how well they will perform in the presence of alternative spellings. One of the virtues of the unified model is that it encourages exploration of alternative spellings and estimation of letter frequencies within words.

REFERENCES

1. International Human Genome Sequencing Consortium, "Initial sequencing and analysis of the human genome," *Nature*, 409: 860–921, 2001.
2. J. C.Vender et al. "The sequence of the human genome," *Science*, 291: 1304–1351, 2001.
3. C. E. Lawrence and A. A. Reilly, "An expectation maximization (em) algorithm for the identification and characterization of common sites in unaligned biopolymer sequences," *Proteins*, 7: 41–51, 1990.
4. C. E. Lawrence, S. F. Altschul, M. S. Bogouski, J. S. Liu, A. F. Neuwald, and J. C. Wooten, "Detecting subtle sequence signals: A Gibbs sampling strategy for multiple alignment," *Science*, 262: 208–214, 1993.
5. K. Robison, A. M. McGuire, and G. M. Church, "A comprehensive library of DNA-binding site matrices for 55 proteins applied to the complete *Escherichia coli* K12 genome," *J. Mol. Biol.*, 284: 241–254, 1998.
6. H. J. Bussemaker, H. Li, and E. D. Siggia, "Building a dictionary for genomes: Identification of presumptive regulatory sites by statistical analysis," *Proc. Natl. Acad. Sci.*, 97: 10096–10100, 2000.
7. H. J. Bussemaker, H. Li, and E. D. Siggia, "Regulatory element detection using a probabilistic segmentation model," *Proc. Int. Conf. Intell. Syst. Mol. Biol.*, 8: 67–74, 2000.
8. L. E. Baum, "An inequality and associated maximization technique in statistical estimation for probabilistic functions of Markov processes," *Inequalities*, 3: 1–8, 1972.
9. P. A. Devijver, "Baum's forward-backward algorithm revisited," *Pattern Recognition Lett.*, 3: 369–373, 1985.
10. K. Lange, D. R. Hunter, and I. Yang, "Optimization transfer using surrogate objective functions (with discussion)," *J. Computat. Graph. Stat.*, 9: 1–59, 2000.
11. K. Lange, *Mathematical and Statistical Methods for Genetic Analysis*, 2nd ed., Springer-Verlag, New York, 2002.
12. T. D. Schneider and R. M. Stephens, "Sequence logos: A new way to display consensus sequences," *Nucleic Acids Res.*, 18: 6097–6100, 1990.
13. C. Sabatti, L. Rohlin, K. Lange, and J. Liao, "Vocabulon: A dictionary model approach for reconstruction and localization of transcription factor binding sites," UCLA Statistics Department Preprint No. 369.
14. L. A. McCue, W. Thompson, C. S. Carmack, M. P. Ryan, J. S. Liu, V. Derbyshire, and C. E. Lawrence, "Phylogenetic footprinting of transcription factor binding sites in proteobacterial genomes," *Nucleic Acids Res.*, 29: 774–782, 2001.

CHAPTER 7

Error Control Codes and the Genome

ELEBEOBA E. MAY

In the latter part of the 1980s the increase in genomic data spurred a renewed interest in the use of information theory in the study of genomics [1–3]. Information measures, based on the Shannon entropy [4], have been used in recognition of DNA patterns, classification of genetic sequences, and other computational studies of genetic processes [1–18]. Informational analysis of genetic sequences has provided significant insight into parallels between the genetic process and information-processing systems used in the field of communication engineering [18, 19–21]. This chapter explores the application of error control coding (ECC) theory to the analysis of genomic sequences and provides an overview of coding theoretic frameworks for modeling information processing in living systems.

7.1. ERROR CONTROL AND COMMUNICATION: A REVIEW

The need for coding theory and its techniques stems from the need for error control mechanisms in a communication system. Error control mechanisms can be categorized as forward error correction and retransmission error control. Forward error correction assumes that errors will occur and provides a mechanism that, when applied to the received message, is able to correct the errors. Retransmission error control techniques detect the errors in the received message and request retransmission of the message [22]. The system in Figure 7.1 illustrates how ECC is incorporated into a typical engineering communication system. Digitized information is compressed by the source encoder and then redundant symbols are added by the channel encoder in preparation for transmission. The error-control-encoded message is transmitted via a potentially noisy channel where the transmitted information may be corrupted in a random fashion. At the receiver, the received message is decoded by the channel decoder. The decoding process involves recognition and removal of errors introduced during transmission and removal of the

Genomics and Proteomics Engineering in Medicine and Biology. Edited by Metin Akay

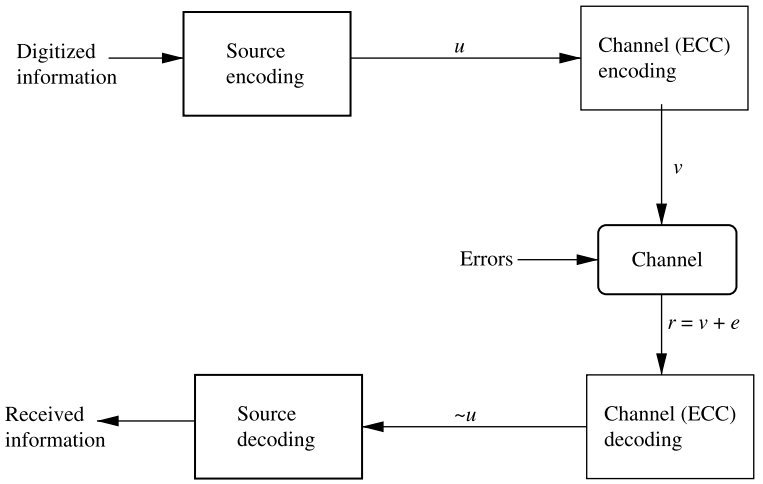


FIGURE 7.1. Communication system that incorporates coding.

redundant symbols used for error protection. The decoding mechanism can only cope with errors that do not exceed its error correction capability. The source decoder uncompresses the transmitted information producing the original digitized information.

7.1.1. Error-Correcting Codes

The elements we will focus on in this system are the encoder, the channel, and the decoder, elements responsible for the error control and correction aspects of communication.

7.1.1.1. Encoder The encoder encodes the digitized information frame by frame. An input frame consists of a fixed number k of symbols that are presented to the encoder. The output frame, the frame to be transmitted, consists of n (also fixed) output symbols, where n is larger than k. Since the number of output symbols is greater than the number of input symbols, redundancy has been introduced [22]. The coding rate

$$R = \frac{k}{n} \tag{7.1}$$

is the the ratio of the number of input symbols in a frame to the number of output symbols in a frame. The lower the coding rate, the greater the degree of redundancy [22].

The encoder combines the k input symbols with $n - k$ symbols usually based on a deterministic algorithm, although random encoding methods, as illustrated by Shannon, can be used [4, 23]. Encoding results in a mapping of input frames into

a set of output frames known as codewords. There must be a codeword for every possible information sequence. For a q-ary alphabet, the encoder will produce q^k codewords. As an example, for a binary code ($q = 2$) with $k = 2$, there are 2^2, or four, possible information sequences and therefore four codewords. The set of q^k codewords comprises the codebook. Because encoding adds redundant bits, there are a number of n-bit sequences (exactly $q^n - q^k$ such sequences) which are not codewords. This allows error detection and correction. If a transmitted n-bit sequence does not map to a codeword, we assume one or more bits have been corrupted. The decoding task is to find the most likely changes in a transmitted n-bit sequence that will result in a valid codeword. The type of output produced is determined by the number of input frames used in the encoding process. Block coding uses only the current input frame. Convolutional coding uses the current frame plus m previous input frames [22, 24]. Error control codes can be referred to as (n, k) codes or (n, k, m) codes in the case of convolutional codes, where m is the memory length (a more detailed discussion of encoder memory is presented in Section 7.1.3).

7.1.1.2. Communication Channel The communication channel is the medium through which information is transmitted to the receiver. The channel can corrupt the transmitted message through attenuation, distortion, interference, and addition of noise. The way in which transmitted binary symbols (0 or 1) are corrupted depends on various characteristics of the communication channel [22]:

- If the channel is a *memoryless channel*, the probability of binary symbol error is statistically independent of the error history of the preceding symbols.
- If the channel is a *symmetric channel*, for binary symbols 0 and 1, the probability of 0 being received instead of 1, due to transmission errors, is the same as the probability of 1 being received instead of 0.
- If the channel is an *additive white Gaussian noise* (*AWGN*) *channel*—a memoryless channel—this adds wideband, normally distributed noise to the amplitude-modulated transmitted signal.
- If the channel is a *bursty channel*, there are periods of high symbol error rates separated by periods of low, or zero, symbol error rates.
- If the channel is a *compound channel*, the errors are a mix of bursty errors and random errors.

7.1.1.3. Decoder The method of decoding used by the channel decoder is dependent on the method of encoding. The aim of a coding system is to attempt to detect and correct the most likely errors. The decoder receives a series of frames that, given no errors in the transmitted sequence, should be composed only of codewords. If the received sequence has been corrupted during transmission, there will be sequences which do not map uniquely to any codewords. This is used to detect the presence of errors. Different mechanisms are then used to decide what the original codeword was and thus correct the error. When the error rate exceeds the

correction capacity of the code, two things can occur: (1) The decoder can detect the error but cannot find a unique solution and thus correct the error or (2) the decoder cannot detect the error because the corruption has mapped one legal codeword into another legal codeword. Errors that exceed the error-correcting capabilities of the code may not be handled correctly.

7.1.2. Basics of Linear Block Codes

For block and convolution codes, the mathematics is carried out in a finite field also referred to as a Galois field [22, 23]. A q-ary finite field GF(q) is a Galois field with q elements that consists of a finite set of symbols, a set of two operations, and the inverses of those operations. The operations and their inverses, when applied to the set of symbols, can only yield values within that set. As an example, the binary field GF(2) consists of

- as finite set of symbols $0, 1$;
- the operations modulo 2 addition ($+$) and modulo 2 multiplication ($*$); and
- corresponding inverse operations.

7.1.2.1. Encoding Methodology A linear block code is a code defined such that the sum of any two codewords results in another valid codeword in the codebook set. There are several ways for a block encoder to produce codewords from a k-bit information sequence [25]. One method, systematic encoding, produces codewords which contain the k information bits at the beginning of the codeword. The information bits are then followed by $n - k$ parity bits. All nonsystematic linear block codes can be reduced to an equivalent systematic code [23]. The value of these $n - k$ bits is determined by the encoding algorithm contained in the generator matrix G. The generator matrix is used to encode the k-bit information vector u and form the n-bit transmitted codeword vector v. The relationship between u, v, and G is

$$v = uG \tag{7.2}$$

(Note: Throughout this chapter ab denotes a times b.)
The generator matrix G is $k \times n$, u is $1 \times k$, and v is $1 \times n$; this yields the following matrix representation of the above equation:

$$[v_1 \quad v_2 \quad \cdots \quad v_n] = [u_1 \quad u_2 \quad \ldots \quad u_k] \begin{bmatrix} g_{11} & g_{12} & \cdots & g_{1n} \\ \vdots & \vdots & \ddots & \vdots \\ g_{k1} & g_{k2} & \cdots & g_{kn} \end{bmatrix} \tag{7.3}$$

The codeword v is produced by the modulo q addition of basis codewords [22]. The basis codewords are the k linearly independent codewords that form the generator matrix. Linearly independent codewords are the set of k vectors that cannot be produced by linear combinations of two or more codewords in the codebook set.

TABLE 7.1 Data to Parity Mapping for Simple (3,2) Linear Block Code

u	$v = uG$
00	000
01	011
10	101
11	110

When the generator matrix is in systematic form, G is of the form

$$G = [I_k; P] \tag{7.4}$$

where I_k is the $k \times k$ identity matrix and P is a $k \times (n - k)$ matrix [23]. Equation [7.5] and Table 7.1 show the generator and corresponding data to parity mapping for a simple (3,2) linear block code:

$$G = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix} \tag{7.5}$$

From Table 7.1 we note that the codebook set is $S_C = (000, 011, 101, 110)$.

7.1.2.2. Decoding Methodology Decoding involves two steps. First the decoder must check whether the sequence corresponds to a codeword. Second, if the decoder is an error-correcting decoder, then it must identify the error pattern. There are various decoding methods. One method, minimum-distance decoding, is a maximum-likelihood approach based on comparing Hamming distance values between a received sequence and codewords in the codebook. The Hamming distance between two sequences, $d(a, b)$, is the number of differences between sequence a and sequence b [22]. For a received sequence r, the minimum distance $d_{\min}$ of r is the minimum of $d(r, S_c)$, where S_c is the set of all codewords v in the codebook. In minimum-distance decoding, we decode r to the codeword for which $d(r, S_c)$ is the least. If the minimum-distance computation results in the same distance value for more than one codeword, although an error is detected, it is not correctable because of the degeneracy of the mapping. Minimizing the distance is the optimum decoding approach for a channel in which symbol errors are independent (memoryless channel) [22].

Another decoding technique, syndrome decoding, is based on the relationship between r, the received sequence (a potentially noisy version of v), and the $(n - k) \times n$ parity-check matrix H. The H matrix is the generator for the dual code space with respect to the code space generated by G [23]. The parity-check matrix has the form

$$H = [P^{\mathrm{T}}; I_{n-k}] \tag{7.6}$$

where P^{T} is the transpose of the P matrix of G [see Eq. (7.4)] and I_{n-k} is the $(n-k) \times (n-k)$ identity matrix [23]. The relationship between H and G is

$$GH^{\mathrm{T}} = 0 \tag{7.7}$$

For every valid codeword v in the coding space of G

$$vH^{\mathrm{T}} = 0 \tag{7.8}$$

If we represent the n-symbol received vector r as $r = v + e$, where e represents the error vector introduced by the channel, we can define the syndrome of r as

$$s = rH^{\mathrm{T}} = (v + e)H^{\mathrm{T}} = 0 + eH^{\mathrm{T}} \tag{7.9}$$

The syndrome is the error pattern present in the received information sequence. In the absence of detectable errors, $s = 0$. The syndrome pattern can be used to correct and decode the received information sequence. Using the simple (3,2) code in Eq. (7.5), the corresponding H matrix is

$$H = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix} \tag{7.10}$$

Given two received messages $r_1 = [011]$ and $r_2 = [010]$, we can calculate the syndrome values for each, potentially noisy sequence:

$$s_1 = r_1 H^{\mathrm{T}} = \begin{bmatrix} 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} = [0] \tag{7.11}$$

$$s_2 = r_2 H^{\mathrm{T}} = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} = [1] \tag{7.12}$$

From this simple illustration, we note that the nonzero s_2 syndrome value accurately indicates the presence of an error in r_2 while the zero s_1 value indicates the absence of errors in the received r_1 sequence. May et al. theorize that this syndrome-checking framework can be paralleled to the behavior of various macromolecules, such as the ribosome, that operate on genetic messages [26].

7.1.3. Basics of Convolutional Codes

Block codes produce encoded blocks from the present information block at time i. In contrast, convolutional coding produces encoded blocks based on present and past information bits or blocks. Convolutional coding, like block coding, is carried out over a finite field using a set of discrete source symbols. For now, we consider the binary field, consisting of [0, 1] and the operations modulo 2 addition and

modulo 2 multiplication. In convolutional encoding, an n-bit encoded block at time i depends on the k-bit information block at time i and on m previous information blocks [24]. Hence, a convolutional encoder requires memory. Convolutional codes are referred to as (n, k, m) codes.

7.1.3.1. Encoding Methodology A convolutional encoder is a mechanism with a k-bit input vector u_i, n-bit output vector v_i, and m memory elements. Figure 7.2 illustrates a $(2, 1, 2)$ convolutional encoder, where the blocks indicate memory [24]. This is a $k = 1$, $n = 2$, or $\frac{1}{2}$ rate encoding scheme where a block is equal to one bit. That is, for every input bit encoding produces two parity bits. The general encoding procedure is as follows [22, 24]:

- A k-bit input block at time i, u_i, is modulo 2 added to the previous m input bits to form the n-bit output vector v_i.
- The most recent k input bit is shifted into the memory register and the rest of the bits in the register are shifted to the right.
- The new input block is then modulo 2 added to the contents of the memory register to produce a new output vector.
- The process is repeated until all input data have been encoded.

A set of n generator vectors completely specify the encoder. The generators are $m + 1$ bits long and indicate which elements are modulo 2 added to produce each bit in the output vector. For the encoder illustrated in Figure 7.2, the generator vectors are

$$g_1 = [1 \quad 0 \quad 1] \tag{7.13}$$

$$g_2 = [1 \quad 1 \quad 1] \tag{7.14}$$

The generator vectors can also be represented as generator polynomials:

$$g_1(x) = 1 + x^2 \tag{7.15}$$

$$g_2(x) = 1 + x + x^2 \tag{7.16}$$

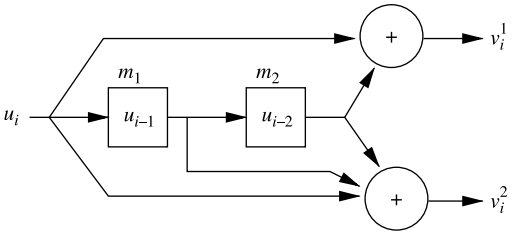


FIGURE 7.2. A $(2, 1, 2)$ convolutional encoder.

For x^D, D represents the number of delay units. Each generator vector or polynomial is associated with one of the n output bits in the output vector v. The encoding process depends not only on the present input but also on the previous m inputs. This forms an interdependence among the transmitted data bits. Given the information stream

$$u(i) = [0 \quad 0 \quad 0 \quad 1 \quad 0 \quad 0] \qquad i = 0, \ldots, 5 \tag{7.17}$$

we can use the convolution code specified by Eqs. (7.13) and (7.14) to produce the corresponding codeword sequence:

$$v(i) = [00 \quad 11 \quad 01 \quad 11] \qquad i = 2, \ldots, 5 \tag{7.18}$$

In the above example, note that the first two valid outputs for v occur at $i = 2$.

7.1.3.2. Decoding Methodology There are various approaches for decoding convolutionally encoded data. Similar to block decoding, the maximum-likelihood decoding approach compares the received sequence with every possible code sequence the encoding system could have produced. Given a received sequence and the state diagram of the encoding system, maximum-likelihood decoding produces the most likely estimate of the transmitted vector v. The Viterbi decoding algorithm [22, 24] is a maximum-likelihood decoding algorithm which uses a code trellis to estimate the transmitted vector given a received vector.

Table-based decoding, another decoding approach, uses syndrome decoding methods and a decoding window which consists of $m + 1$ frames [22, 27, 28]. The received sequence is treated like a block code and a syndrome value is generated for each received block. As in block codes, the value of the syndrome indicates the presence or absence of an error in the received sequence. Although not a maximum-likelihood method, syndrome-based decoding of convolutional codes is more computationally efficient and this decoding model has been used in constructing an ECC framework for modeling translation initiation system [26, 29].

7.2. CENTRAL DOGMA AS COMMUNICATION SYSTEM

To determine the algorithm used by living systems to transmit vital genetic information, several researchers have explored the parallel between the flow of genetic information in biological systems and the flow of information in engineering communication systems, reexamining the central dogma of genetics from an information transmission viewpoint [1, 19, 20, 30, 31]. The central premise of genetics is that genes are perpetuated in the form of nucleic acid sequences but once expressed function as proteins [32]. Investigators have developed models that attempt to capture different information-theoretic aspects of the genetic system [1, 19, 20, 33, 34]. Three of these models are reviewed in the sections that follow.

7.2.1. Gatlin's Communication Model

One of the earliest work on the information-theoretic properties of biological systems is presented by Gatlin [19]. In the opening chapter of her work, Gatlin [19] asserts that "life may be defined operationally as an information processing system . . . that has acquired through evolution the ability to store and process the information necessary for its own accurate reproduction." In Gatlin's interpretation of the biological information-processing system DNA base sequences are the encoded message generated by a source, an error control encoder. The encoded DNA goes through a channel (defined in Gatlin's model by transcription and translation) that Gatlin defines as all the mechanics for protein production. The amino acid sequence of the protein is the received message. It is unclear where DNA replication fits or whether Gatlin considers the replication process as part of the encoder. However, she does suggest that extra bases in DNA may be used for error control and correction purposes.

In addition to an information-theoretic view of genetic processes, Gatlin also parallels the genetic sequence to a computer program. She proposes that the genetic code can be viewed as "part of an informational hierarchy" where the redundant DNA regions and the noncoding DNA have important programmatic control functions. It is well known that non-protein-coding regions of DNA, such as promoters and the 5′ untranslated leader of messenger RNA (mRNA), have regulatory functions in the protein synthesis process, lending plausibility to her early ideas [32, 35].

7.2.2. Yockey's Communication Model

Yockey performs a fundamental investigation of biological information theory and lays the foundations for developing theoretical biology from the mathematical principles of information and coding theory [20]. Yockey's biological information framework is based on a data storage model, where the behavior of the genetic information system is compared to the logic of a Turing machine. The DNA is paralleled to the input tape where the genetic message is the bit string recorded on the tape. The computer program or internal states of the Turing machine are the RNA molecules and molecular machines that implement the protein synthesis process. The output tape, similar to Gatlin's model, is the protein families produced from the recorded message in DNA.

Error-correcting codes are used in data storage media to ensure data fidelity and hence Yockey's model incorporates ECC. A simplified version of Yockey's DNA–mRNA–protein communication system is re-created in Figure 7.3 [20]. In Yockey's DNA–mRNA–protein communication system, the source code is the genetic message in DNA and is stored on the DNA tape. Transcription is the encoder, transferring DNA code into mRNA code. Messenger RNA is the channel by which the genetic message is communicated to the ribosome, which is the decoder. Translation represents the decoding step where the information in the mRNA code is decoded into the protein message or the protein tape. Genetic

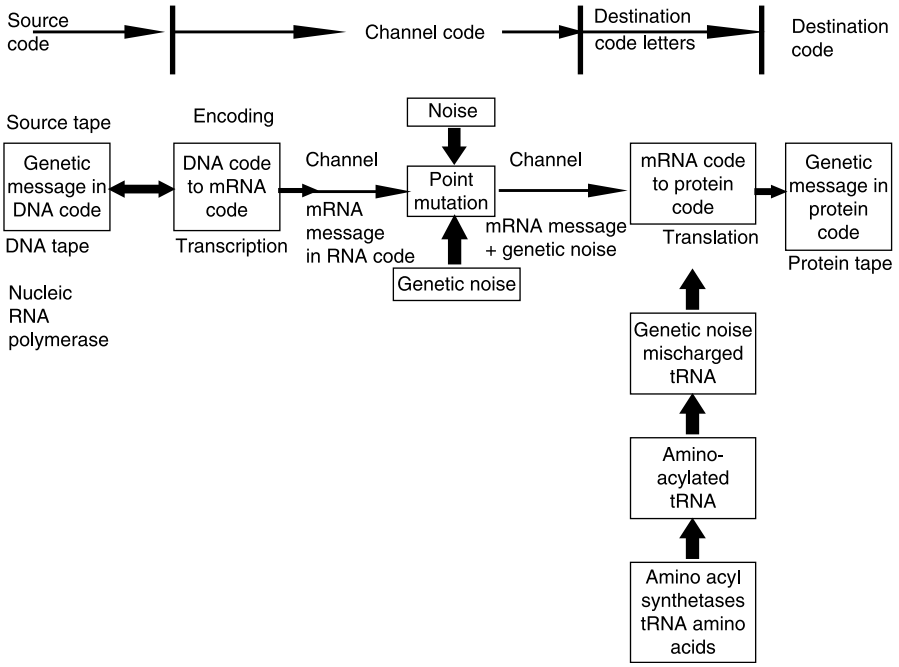


FIGURE 7.3. Yockey's DNA–mRNA–protein communication system.

noise is introduced by events such as point mutations. Yockey states that while genetic noise can occur throughout the system, all of the noise is represented in the mRNA channel.

In Yockey's model, the genetic code (the codon-to-amino-acid mapping) is the decoding process and is referred to as a block code. He suggests that the redundancy in the codon-to-amino-acid mapping is used as part of the error protection mechanism. Therefore we can assume that the transcription step would be the error control encoding step in Yockey's model even though there is not an apparent increase in redundancy during the transcription process.

7.2.3. May et al.'s Communication Model

Drawing on Battail [30] and Eigen's [21] work, May et al.'s [26] communication view of the genetic system assumes (1) a nested genetic encoding process and (2) that the replication process represents the error-introducing channel. As a direct consequence of their first assumption, the genetic decoding process is separated into three phases: transcription, translation initiation, and translation elongation plus termination. Figure 7.4 depicts this view of information transmission in genetic systems. In the genetic communication system, the unreplicated DNA sequence is the output of an error control genetic encoder that adds redundancy to inherently

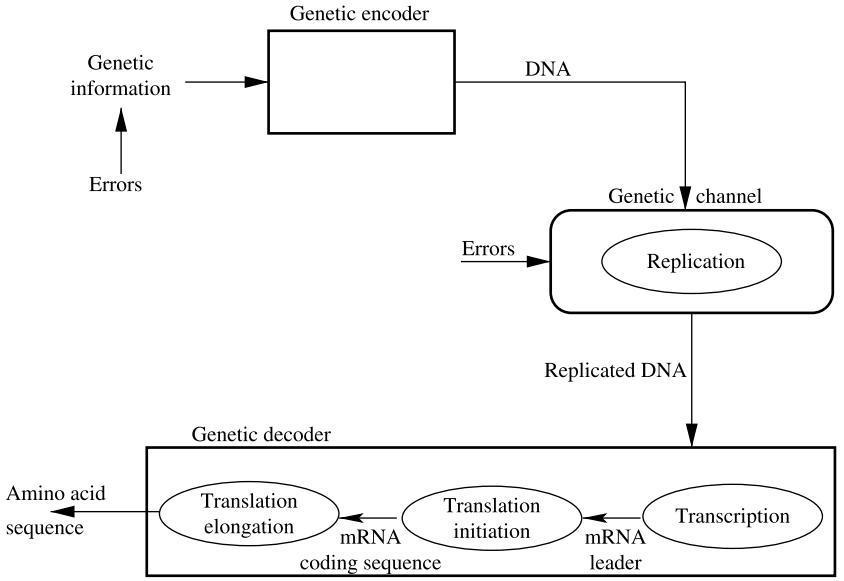


FIGURE 7.4. May et al.'s coding-theoretic view of the central dogma of genetics.

noisy genetic information. The noise in the source can be thought of as mutations transferred from parent to offspring. Defining the genetic encoder in May et al.'s model may require addressing questions similar in scope to those surrounding the genetic code's origins. As Yockey [20] states in reference to this issue, "the reason for the difficulty in speculating on the origin of the genetic code is that there seems to be no place to begin. There is no trace in physics or chemistry of the control of chemical reactions by a sequence of any sort or of a code between sequences." Additional insight into potential biological functions corresponding to the encoder may emerge as researchers continue to investigate the origin and evolution of the genetic code [36, 37].

Unlike the May et al. model, neither Gatlin's nor Yockey's model explicitly addresses replication. Both frameworks represent the noise-introducing channel as the genetic mechanism responsible for protein synthesis, namely transcription and translation in Gatlin's framework and the mRNA itself in Yockey's framework. In contrast, May et al. define the genetic channel as the DNA replication process during which errors are introduced into the nucleotide sequence. Similar to the Yockey model, May et al. parallel the ribosome to an error control decoder. Given the similarities between the translation and transcription mechanisms, transcription is represented as a decoding step and the RNA polymerase is viewed as an error control decoder. While Gatlin's work addressed the potential function of non-protein-coding regions, it does not specifically highlight these regions in the communication model of genetics. May et al.'s model distinguishes between the error control decoding of protein-coding regions and the decoding of

non-protein-coding, regulatory regions that control translation initiation, which is typically the rate-limiting aspect of the protein production process [32, 38, 39]. Given the importance of regulating protein synthesis and the redundancy present in regulatory regions such as the ribosome binding site (RBS), it seems plausible that regulatory information is error control encoded [40].

7.3. REVERSE ENGINEERING THE GENETIC ERROR CONTROL SYSTEM

The information produced by genome projects is key to understanding how an organism functions from genetic- to cellular-level behavior. Identifying gene locations and regulatory regions is a fundamental step in this process. It is not feasible to experimentally annotate all of an organism's regulatory regions—hence the need for computational tools for accurately deciphering the information contained in genetic sequences. The majority of gene annotation techniques rely on patterns and statistical characteristics of the genome for model construction. While these methods yield viable results, they do not offer insight into the underlying mechanics of the genetic process.

Coding theory algorithms can serve as powerful pattern recognizers for annotating biologically active sites of a genome and also as pattern generators that can mathematically represent the genetic process and macromolecules that operate on a genomic sequence of interest. The mathematical representation of a convolutional code is also the mathematical model for the digital system that produces that signal (or pattern) and all other signals associated with that system.

Development of coding-theoretic frameworks for molecular biology is an ongoing endeavor. Although the existence of redundancy in genetic sequences is accepted and the possibility of that redundancy for error correction and control is being explored and exploited, mathematically determining the encoding algorithm, particularly for regulatory regions, remains a major research challenge. Devising a method for reconstructing the error control code of a received, noisy signal is a challenge that if met will provide a way to construct mathematical models of molecular machines (macromolecules such as ribosome, RNA polymerase, and initiation factors) involved in the regulation of genetic processes. Reverse engineering the genetic ECC system requires thorough investigation of several issues, including:

1. Is there plausible and potentially quantitative evidence that ECC exists in genomic sequences.
2. What are the ECC characteristics of the genetic system (characteristics that parallel traditional communication systems such as channel capacity and coding rate)?
3. Assuming the existence of some type of genetic ECC, how can we computationally invert the system using potentially noisy sequence information?

7.3.1. Making a Case for Existence of Error Control in Genomic Sequence

Several researchers have moved beyond the qualitative models of biological communication and attempted to determine the existence of error control codes for genomic sequences [20, 31, 34, 41, 42]. Liebovitch et al. [41] and Rosen and Moore [34] both developed techniques to determine the existence of an error control code for genomic sequences. Neither found evidence of error control codes for the sequences tested. Given the computational limitations of the study, Liebovitch et al. suggest that a more comprehensive examination would be required. Both methods investigate a subset of linear block codes and do not consider convolutional coding properties or account for the inherent noise in genomic sequences. Extending beyond specific genomic regions and sequences, MacDonaill develops an ECC model for nucleic acid sequences in general [42]. He has proposed a four-bit, binary, parity-check error control code for genetic sequences based on chemical properties of the nucleotide bases.

Battail presents a more qualitative argument supporting the potential existence of a genetic error control system [30]. He argues, similar to Eigen [21], that for Dawkins's model of evolution to be tractable, error correction or ECC must be present in the genetic replication process. According to Battail, proofreading, a result of the error avoidance mechanism suggested by genome replication literature, does not correct errors present in the original genetic message. Only a genetic error correction mechanism can guarantee reliable message regeneration in the presence of errors or mutations due to thermal noise, radioactivity, and cosmic rays [30].

Battail further asserts that the need for error protection becomes obvious when one considers that the number of errors in a k-symbol message that has been replicated r times is comparable to the number of errors in an unreplicated $(r \times k)$-symbol message. For a given error rate, the number of times an organism undergoes replication approaches an infinite number. Hence for a message to remain reliable within an organism's life cycle (not to mention evolutionary information transmission which occurs over thousands of years), the message must have strong error protection. Battail points out that if there exists a minimum Hamming distance d between codewords, then almost errorless communication is possible if and only if the following holds:

$$p \times n < \tfrac{1}{2}d \tag{7.19}$$

where p is the error probability for the channel and n is the length of the codewords. If we take n to be the length of the gene or a portion of the gene, minimum-distance decoding may be used to produce a near errorless rule [30]. Eukaryotes' tendency to evolve toward increasing complexity may parallel the connection between increasing word length and increasing reliability, which is stated in the fundamental theorem of channel coding [30]. The fundamental theorem of channel coding states that coding rates that are below the channel capacity result in arbitrarily small probabilities of error ($\lambda^n \to 0$) for sufficiently large blocks lengths n [43].

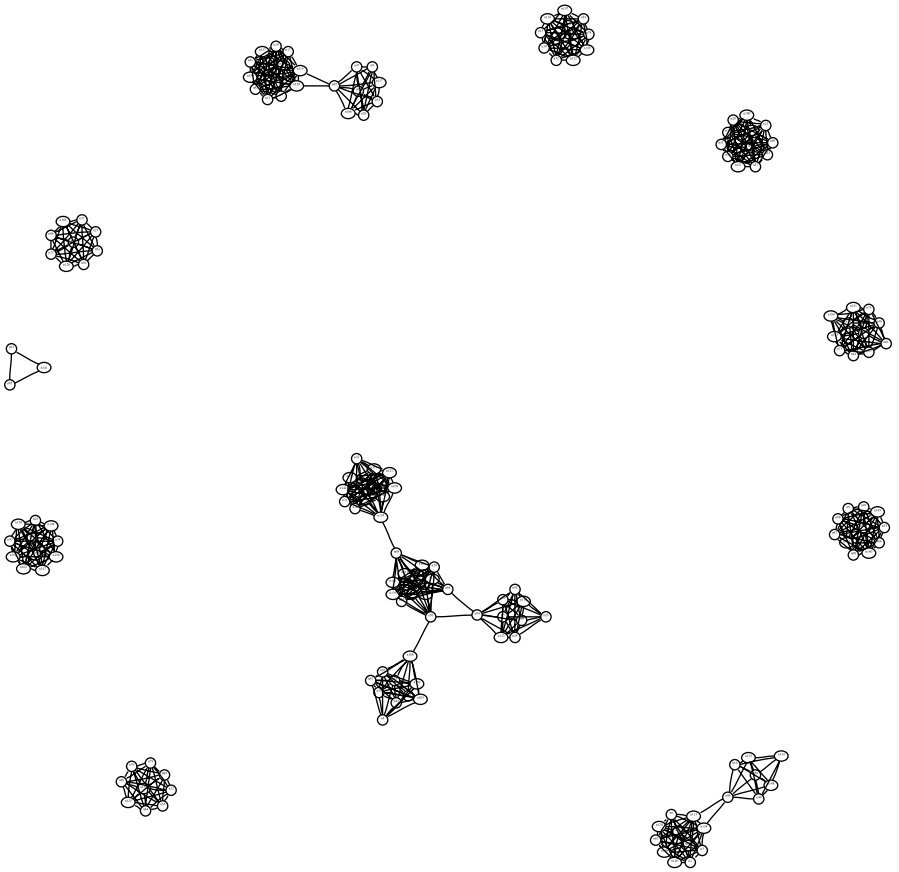


FIGURE 7.5. Binary (7, 4) block code.

Taking a unique approach to the question of error control codes in genomic sequences, Schmidt and May [44] exploit graph-theoretic methods in their investigation of ECC properties of *Escherichia coli* K-12 translation initiation sequences. They discover that unlike binary random sequences, binary block codes form distinctive cluster graphs (see Figs. 7.5 and 7.6). Applying this graph-based method to a subset of *E. coli* K-12 initiation sites, they observe that while noninitiation sites fail to cluster into distinct groups (Fig. 7.7), there is evidence of cluster formation in valid initiation sequences (Fig. 7.8), suggesting the possibility of ECC-type characteristics for *E. coli* translation initiation sites.

7.3.2. System Characteristics

The capacity of the communication channel is a key system characteristic that governs the type of ECC used in transmission. The genetic communication system depicted in Figure 7.4 represents the error-introducing transmission

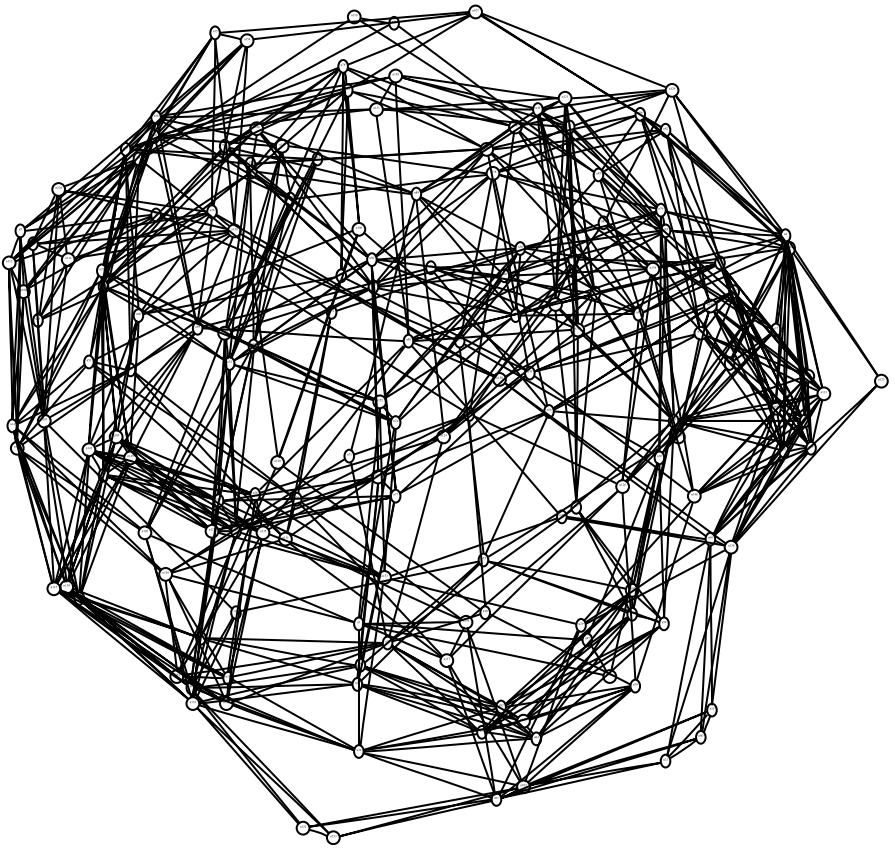


FIGURE 7.6. Binary random sequence.

channel as the replication process. Shannon's channel coding theorem asserts that there exists a channel code with rate $R = k/n$ such that the probability of decoding error becomes arbitrarily small as n increases [4, 23, 43]. The capacity of a transmission channel (the maximum data transmission rate) is dependent on the error rate of the channel $p_{i,j}$, the probability of the channel transforming symbol i into symbol j for $i \neq j$. In order to determine appropriate ECC parameters for genetic regulatory sequences, we must characterize the replication channel and the error or mutation rates associated with replication. Mutation-derived capacity values can suggest R and from that plausible n and k values for genetic systems.

Mutations are replication errors that remain or are missed by genetic proofreading mechanisms. Drake et al. [45–47] have performed extensive research and analysis of mutation rates in prokaryotic and eukaryotic organisms. Based on mutagenesis studies, they note that mutation rate in RNA viruses range from 1 per genome per replication for lytic viruses to 0.1 per genome per replication for retroviruses and retrotransposons. The DNA microbes, more complex and

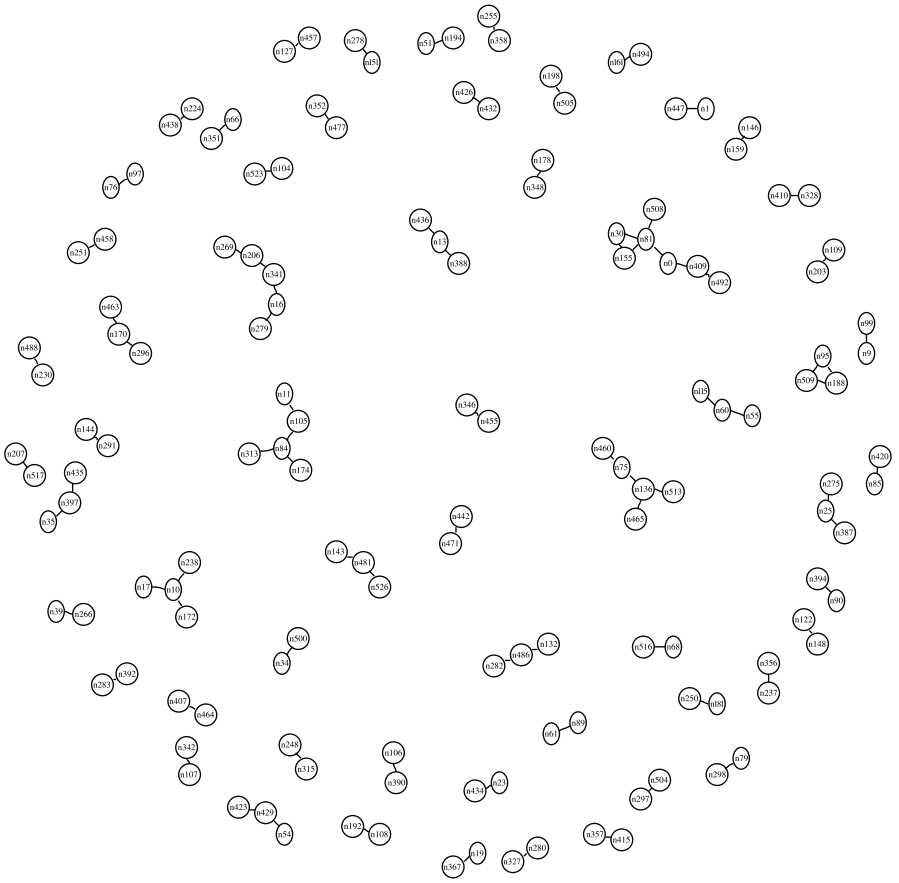


FIGURE 7.7. Graphical representation of ECC properties of *E. coli* K-12 noninitiating intergenic regions.

typically larger than RNA viruses, have mutation rates of $\frac{1}{300}$ per genome per replication. Moving higher still to the larger, more complex eukaryotic organism, higher eukaryotes have mutation rates ranging from 0.1 to 100 per genome per sexual generation and a mutation rate of $\frac{1}{300}$ per cell division per effective genome. The effective genome is the portion of the genome where mutations are most lethal (i.e., genes or exons) [45]. In general, while RNA viruses have significantly higher mutation or channel error rates, DNA microbes have error rates relatively similar to the mutation rate in the effective genome of higher eukaryotes. The question arises whether and how organism complexity (which we can loosely approximate using genome size) is related to replication channel fidelity. Drake investigates this for DNA microbes by analyzing the log-log plot of base mutation rates as a function of genome size [46]. These plots are reproduced using the base mutation and genome size data from Drake et al. [45] for both the DNA microbes and the higher eukaryotes.

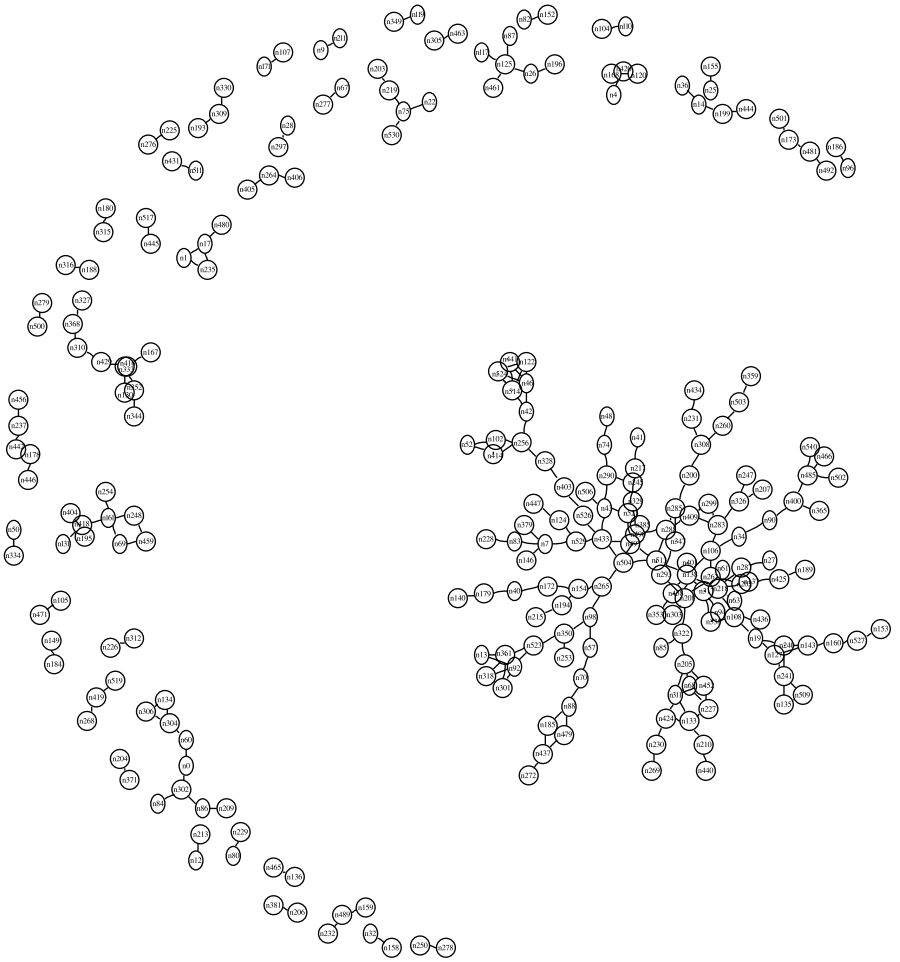


FIGURE 7.8. Graphical representation of ECC properties of *E. coli* K-12 translation initiation sites, position −10 to −3.

Figures 7.9 and 7.10 show the log-log plots of genome size as a function of base mutation for DNA microbes and eukaryotic organisms, respectively. The log-log plots for the DNA microbes are equivalent to Drake et al.'s results, as would be expected. The relationship between the DNA microbes' mutation rates and genome size exhibits power law behavior. Higher eukaryotes do not appear to exhibit similar behavior, although the eukaryotic data set contained a relativly small number of organisms. As concluded by Drake et al. and illustrated in Figure 7.9, there is an inverse relationship between genome size G and an organism's base mutation rate μ_b. This inverse relationship is evident for the higher eukaryotes as well.

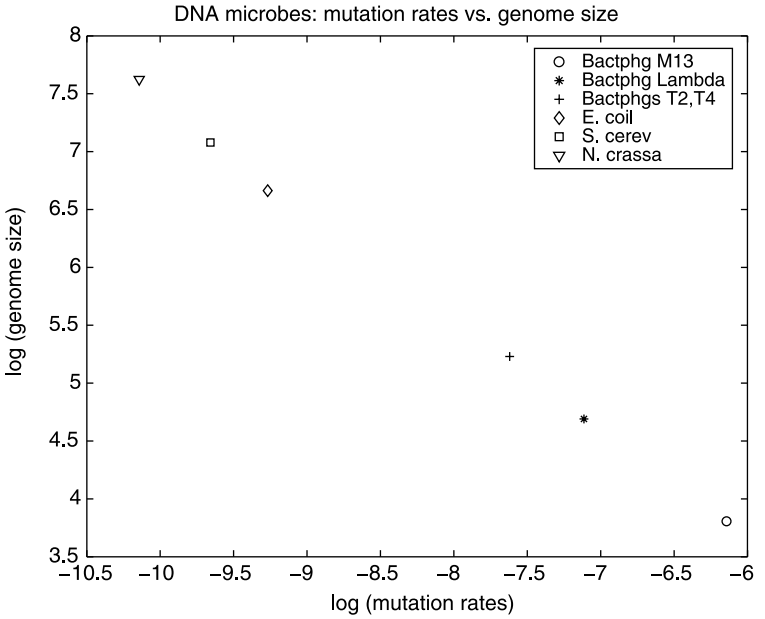


FIGURE 7.9. Comparison of microbial genome mutation rate to genome size.

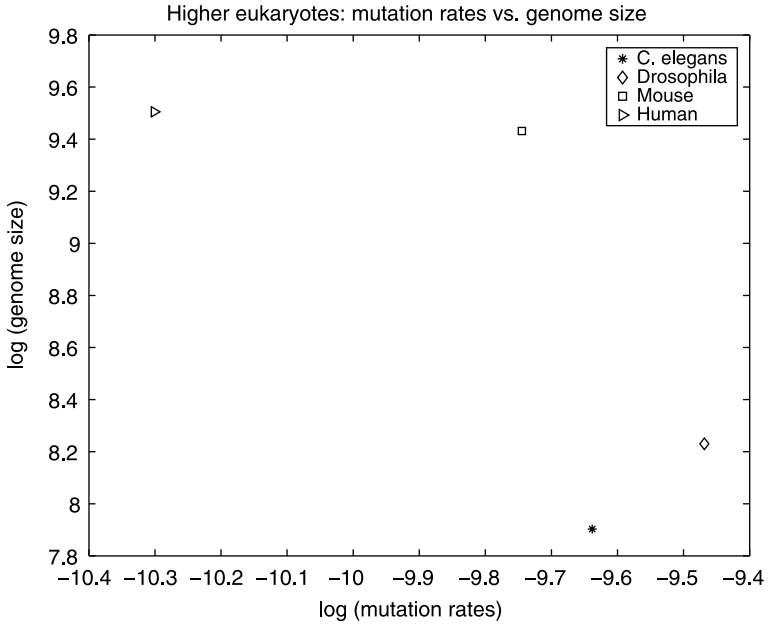


FIGURE 7.10. Comparison of eukaryotic genome mutation rate to genome size.

TABLE 7.2 Channel Transition Probability Assuming p(transition mutation) = p(transversion mutation)

	A	G	C	T
A	$1-\mu_b$	$\frac{1}{3}\mu_b$	$\frac{1}{3}\mu_b$	$\frac{1}{3}\mu_b$
G	$\frac{1}{3}\mu_b$	$1-\mu_b$	$\frac{1}{3}\mu_b$	$\frac{1}{3}\mu_b$
C	$\frac{1}{3}\mu_b$	$\frac{1}{3}\mu_b$	$1-\mu_b$	$\frac{1}{3}\mu_b$
T	$\frac{1}{3}\mu_b$	$\frac{1}{3}\mu_b$	$\frac{1}{3}\mu_b$	$1-\mu_b$

The genetic channel capacity is calculated using mutation rates reported in Drake et al. [45]. Assuming a discrete memoryless channel (DMC), the capacity of the channel, C, is the maximum reduction in uncertainty of the input X given knowledge of Y [43]:

$$C = \sup_X I(X, Y) \tag{7.20}$$

where

$$I(X, Y) = H(X) - H(X \mid Y) = H(Y) - H(Y \mid X) \tag{7.21}$$

The Shannon entropy $H(X)$ and $H(Y \mid X)$ are defined as

$$H(X) = -\sum_i p(x_i) \log_2 p(x_i) \tag{7.22}$$

$$H(Y \mid X) = -\sum_k \sum_j p(x_k, y_j) \log_2 p(y_j \mid x_k) \tag{7.23}$$

The probability $p(y_j \mid x_k)$ is the channel error probability. If $p(y \mid x)$ is specified by the mutation error rate μ_b, then $p(y_j \mid x_k) = \mu_b \ \forall y \neq x$ and $p(y_j \mid x_k) = 1 - \mu_b \, \forall y = x$ (where μ_b is the mutation rate per base per replication). The channel transition matrix (Table 7.2) assumes all base mutations are equal; hence a transition mutation [purine to purine, adenine(A) $\longleftrightarrow$ guanine(G), and pyrimidine to pyrimidine, cytosine(C) $\longleftrightarrow$ thymine(T)] and a transversion mutation [purine to pyrimidine, (A, G) $\rightarrow$ (C, T), and pyrimidine to purine, (C, T) $\rightarrow$ (A, G)] are equally probable. Additional capacity calculations are being performed using transition probability matrices where the probability of a transition mutation is greater than the probability of a transversion mutation, which is consistent with the biological evidence. Figures 7.11 and 7.12 show the replication channel capacity of the organism as a function of the log of the organism's genome size for DNA microbes and higher eukaryotes, respectively, using μ_b values from Drake et al. [45] and channel transition probabilities from Table 7.2. The prokaryotic organisms have larger channel capacity than the higher eukaryotes. This suggests that for DNA microbes the coding rate R is closer to $(n-1)/n$, leaving few bases for ECC. In contrast, the channel capacity values for higher eukaryotes implies a distinctly smaller value for R. This implies that the eukaryotic genome has more bases available for ECC.

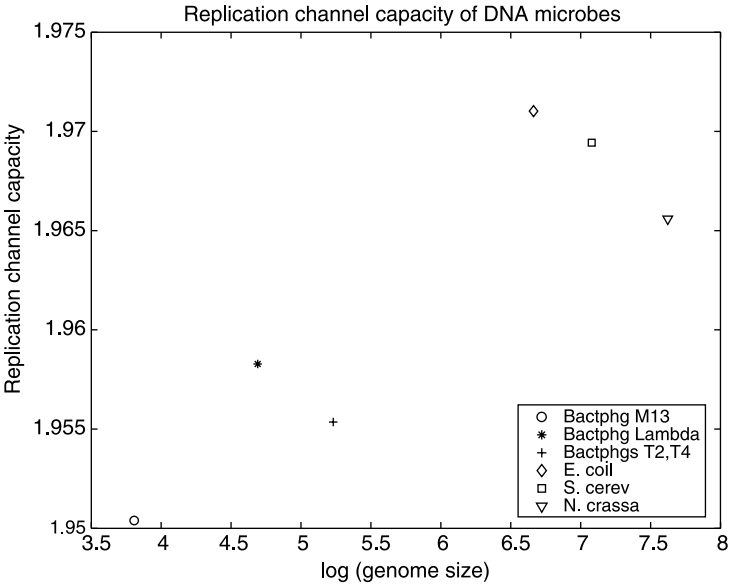


FIGURE 7.11. Capacity of prokaryotic replication channels.

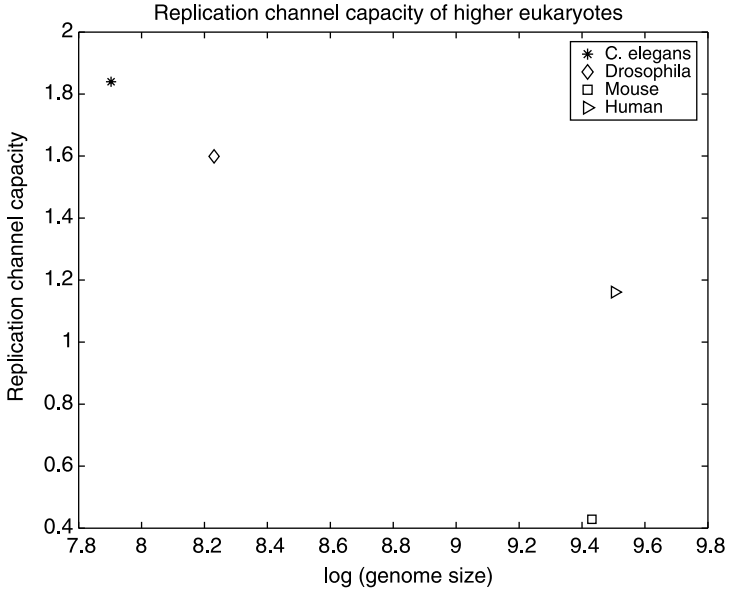


FIGURE 7.12. Capacity of eukaryotic replication channels.

7.3.3. Inverse Error Control Coding Models

If the encoding algorithm for a received error-control-encoded sequence is unknown or part of the data are missing, designing a viable decoder for the received transmission is a significant but rarely addressed computational challenge. Communication engineers forward engineer ECC systems and do not encounter this situation often. In order to determine the ECC properties of genetic systems and the algorithm used by living systems to transmit vital genetic information, researchers are developing quantitative approaches to reverse engineering error-control-encoded data streams and genetic sequences.

7.3.3.1. Inverse ECC Model I: Ribosome as Block Decoder May et al. [48, 49] modeled mRNA as a noisy, systematic zero-parity encoded signal and the ribosome as an (n, k) minimum-distance block decoder (where the 16S ribosomal RNA is used as a template for generating all valid n-length codewords). The model was able to distinguish between translated sequence groups and nontranslated sequence groups from *E. coli* K-12 genome. When applied to mRNA leader regions of other prokaryotic organisms (*Salmonella typhimurium* LT2, *Bacillus subtilis*, and *Staphylococcus aureus* Mu50), similar results were observed.

The original block code model was developed based on the last 13 bases of the 3′ end of 16S ribosomal RNA [32] and consisted of a set of 33 and 26 codewords for the (5,2) and (8,2) codes, respectively. The codewords were constructed using heuristics based on RNA/DNA base-pairing principles and common features of bacterial ribosomal binding sites (such as the existence and location of the Shine–Dalgarno sequence). Although the original model can be used to distinguish between valid and invalid leader sequences, a deterministic representation would provide a quantitative model of the translation initiation system that can be used to algorithmically correct errors in the system and generate plausible leader sequences.

Toward this end, we revisit the algorithmic structure of linear block codes, previously described in Section 7.1.2. Each codeword v in an (n, k) linear block code's codebook can be produced using a generator matrix G, which encodes the information vector u in a deterministic manner [25]. Recall the relationship between u, v, and G can be expressed as

$$v = uG \tag{7.24}$$

where G is $k \times n$, u is $1 \times k$, and v is $1 \times n$. The parity-check matrix H is a $(n - k) \times n$ matrix and relates to the generators as follows [23, 25]:

$$GH^{\mathrm{T}} = 0 \tag{7.25}$$

where H^{T} is the transpose of the parity-check matrix. The parity-check matrix is used to check for transmission errors in the received sequence, $r = v + e$ as previously discussed. In the absence of errors, $e = 0$, the syndrome vector s will be an

all-zero vector:

$$s = rH^{\mathrm{T}} = (v + e)H^{\mathrm{T}} = vH^{\mathrm{T}} = 0 \tag{7.26}$$

If $C_{n,k}$ represents the codebook (i.e., contains all codewords v) for a linear (n,k) block code, then based on Eq. (7.26) we can state the following:

$$C_{n,k}H^{\mathrm{T}} = Z \tag{7.27}$$

where Z is the all-zero matrix. Therefore, given a set of codewords produced using a linear block code, it is feasible to determine the dual code H and ultimately the corresponding generator G for the codebook. This is the rationale used in constructing a generator for the systematic block code model for translation initiation in prokaryotes.

For systematic $(n,\ k)$ codes, the model assumed by May et al. [49], G and H are of the form

$$G = [I_k; P] \tag{7.28}$$

$$H = [P^{\mathrm{T}}; I_{n-k}] \tag{7.29}$$

where P is a $k \times (n-k)$ matrix and I represents the $k \times k$ [or $(n-k) \times (n-k)$] identity matrix [23, 25]. Assuming a systematic code reduces the number of unknowns in the H matrix by $(n-k)^2$, the systematic form also simplifies conversion from H back to G.

For a given codebook set $C_{n,k}$ corresponds to a linear, systematic block code. We can find the optimal solution for H by interrogating all possible solutions for P (except $P = Z$). The optimal solution produces an H that optimizes a cost function of the form

$$\text{Fitness}(H \mid P) = R_S \frac{|\text{zeros in } S|}{|S|} + R_P \frac{|\text{nonzeros in } P|}{|P|} \tag{7.30}$$

where S represents the syndrome matrix (each row in S corresponds to the syndrome of a codeword in $C_{n,k}$) and $R_S + R_P = 1.0$.

The methodology was tested using the (7,4) Hamming code's codebook, $C_{\text{Hamming}_{(7,4)}}$ [25]. The algorithm successfully recovered the generator matrix for the (7,4) Hamming code. The verification test produces a code with a fitness value of 1; this is expected since $C_{\text{Hamming}_{(7,4)}}$ is a complete, error-free representation of the code.

Given the positive results of the original block code model [49], the codebook, $C_{\text{Original}_{(5,2)}}$, for the systematic parity check code is used as an initial estimate of the set of valid codewords for the translation initiation system. Generators were also constructed using two additional codebook sets: $C_{\text{Orig}D\text{min}_{(5,2)}}$ and $C_{16\text{S}_{(5,2)}}$. Here, $C_{\text{Orig}D\text{min}_{(5,2)}}$ is a reduced subset of $C_{\text{Original}_{(5,2)}}$, constructed by selecting a minimum number of codewords from $C_{\text{Original}_{(5,2)}}$ such that each two-base

information sequence is represented and the minimum-distance value for the codebook set is maximized. The codewords in the $C_{16S_{(5,2)}}$ codebook are the five-base subsets formed from contiguous bases of the 16S rRNA, known to interact with the mRNA leader region during initiation.

Equation (7.26) is used to evaluate the performance of the optimal generator produced. For each optimal generator matrix G, the corresponding parity-check matrix H is used to calculate the syndrome values for mRNA subsequences that are valid leaders or invalid leaders (intergenic regions). Figure 7.13 shows the results for $C_{16S_{(5,2)}}$, where the horizontal axis is position relative to the first base of the initiation codon and the vertical axis is the average syndrome value (the syndrome is either 0 for actual zero values or 1 for all nonzero syndrome values).

The sets $G_{\text{Original}_{(5,2)}}$ and $G_{\text{Orig}D\text{min}_{(5,2)}}$ did not produce syndrome patterns with regions of distinction resembling the minimum-distance plots produced in May et al.'s original block code model [49]. But, the generator derived from strict subsets of the 16S rRNA (Fig. 7.13) more closely resembles the minimum-distance results of the May et al. model. The $(-12:-6)$ region exhibits the greatest difference between the valid and invalid sequence groups. As expected, in this region the valid leader sequences have the relatively lower average syndrome value. The $G_{16S_{(5,2)}}$ model also produces strong synchronization patterns for the valid sequence group. The synchronization patterns appear to exhibit a frequency of 3, suggesting that the pattern can be useful for reading frame identification.

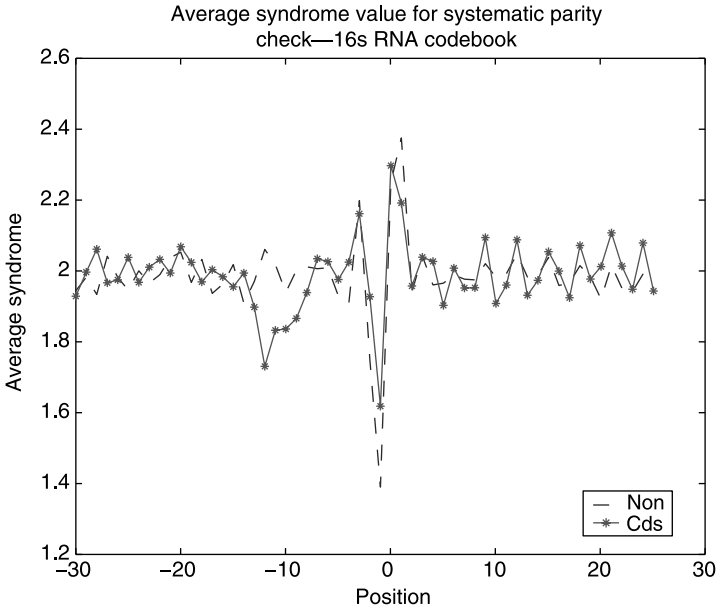


FIGURE 7.13. Average syndrome value for generator $G_{16S_{(5,2)}}$.

The linear block code approach provides a framework for constructing inverse quantitative models for genetic regulatory processes. Optimization and algebraic approaches to solving Eq. (7.27) are being actively explored.

7.3.3.2. Inverse ECC Model II: Functional Inversion In addition to sequence-based models and analysis methods, translation initiation models can be constructed by analyzing possible binding patterns between mRNA leader sequences and the exposed portion of the 16S rRNA. Although binding is related to higher level interactions influenced by mRNA structure, rRNA structure, and ribosomal and protein interactions, it is hypothesized that translation initiation can be viewed from a binary perspective. Studies of prokaryotic translation initiation sites reveal that ribosomal binding sites appear to evolve to functional requirements rather than to genetic sequences that produce the strongest binding site [17]. Several factors influence translation of mRNA sequences, including initiation codon, presence and location of the Shine–Dalgarno sequence, spacing between the initiation codon, and the Shine–Dalgarno domain, the second codon following the initiator codon, and possibly other nucleotides in the -20 to $+13$ region of the mRNA leader region [38]. These factors influence how the small subunit of the ribosome interacts with and binds to the mRNA leader region such that conditions are favorable for successful translation initiation. The binding pattern formed between the 16S rRNA and the mRNA leader region directly affects translation initiation.

Functional Definition of mRNA Leader and Ribosomal Interaction A leader sequence with perfect complementary base pairing to the 16S rRNA may not be the most viable sequence from an evolutionary viewpoint. However, it is plausible to assume that increased affinity to the 16S rRNA increases initiation potential. Not only must a leader sequence contain nucleotides that bind to the 16S, the binding must occur within a reasonable proximity to the initiation codon [38]. Since translation initiation is influenced by positional binding, the biological process of translation initiation can be mapped to a functional, binary domain. Sequence information and the last 13 bases of the 16S rRNA,

$$3'\,\mathrm{A\,U\,U\;\;C\,C\,U\;\;C\,C\,A\;\;C\,U\,A\;\;G}\ldots 5' \tag{7.31}$$

are used to map mRNA leader regions to their positional binding representations.

After mapping the mRNA sequence into binary binding vectors, each vector is categorized based on their (M_1, M_2, M_3) binding pattern values. Given a 13-base binary binding pattern, the value M_1 is the greatest number of consecutive base pairings (1's), M_2 is the second greatest, and M_3 is the third greatest. The expectation is that binding patterns with large M_1 values, within an acceptable distance from the initiation codon, will favor translation initiation. Sequences with smaller M_1 values would be expected to have significant M_2 and M_3 values to increase the probability of ribosome binding.

Each positional binary binding pattern is classified based on their (M_1, M_2, M_3) value. Different binary binding patterns can belong to the same (M_1, M_2, M_3) class. Each (M_1, M_2, M_3) class was assigned a number between 1 ($M_1 = 13$, $M_2 = 0$, $M_3 = 0$) and 91 ($M_1 = 0$, $M_2 = 0$, $M_3 = 0$). For example, given the following two binary binding vectors:

$$\text{BinaryBindingVec}_A = 1\ 1\ 1\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0 \tag{7.32}$$

$$\text{BinaryBindingVec}_B = 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 1\ 0 \tag{7.33}$$

Both vectors would be classified as ($M_1 = 3$, $M_2 = 1$, $M_3 = 0$), or with the classification number 80. The probability of each classification number occurring (based on all possible 13-base binding vectors) is calculated and classification thresholds were tabulated.

To test the assumption that valid initiation regions fall within a given (M_1, M_2, M_3) pattern threshold that differs from the average for the nonleader and random sequence groups, binding analysis is performed on 531 *E. coli* leader sequences, 1000 *E. coli* intergenic, nonleader sequences, and 1000 randomly generated sequences. The nonleader and random sequences all had AUG initiation sites in the center of the candidate sequence. Each sequence contained 60 nucleotide bases.

The mRNA leader sequences were mapped to their corresponding binary binding vectors and classified based on their (M_1, M_2, M_3) values as previously described. The positional (M_1, M_2, M_3) vectors were evaluated for each sequence group. Figure 7.14 shows the percent of sequences with ($M_1 = 4$, $M_2 = 0$, $M_3 = 0$) or

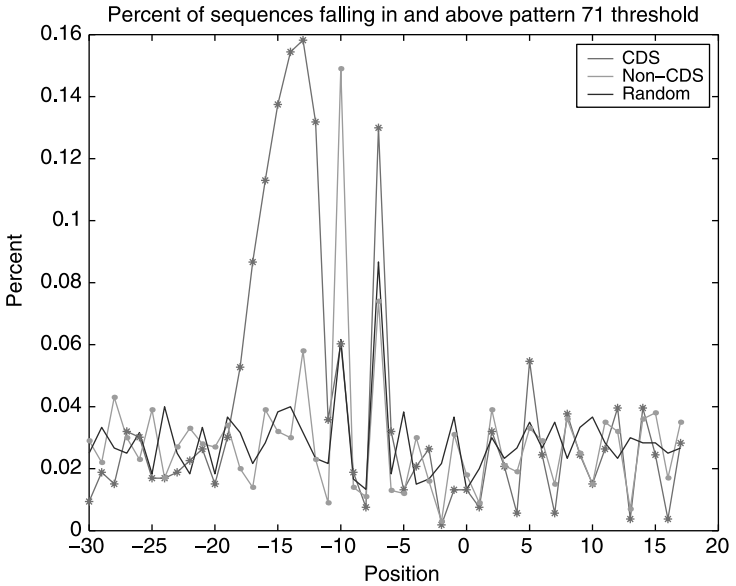


FIGURE 7.14. Percent of sequences with binding pattern of 4,0,0 and above.

stronger binding pattern. The horizontal axis represents position and the vertical axis represents the percent of sequences in each sequence group with a binding pattern of $(M_1 = 4,\ M_2 = 0,\ M_3 = 0)$ or greater. From Figure 7.14 we note the following:

- The region between -18 and -9 has the greatest distinction between the translated sequence group and the nontranslated and random sequence groups. This is consistent with regions of distinction found in previous work [50].
- Inside the coding region for translated sequences (position 0 and greater), there is a clear synchronization pattern which repeats every three bases. This pattern is not as consistent in the nontranslated or random sequence groups.

Using the same sequence information, the best binding pattern was selected for each sequence in each group. The strongest binding pattern (per sequence) was recorded for positions -18 to -12. Table 7.3 shows the distribution of strongest binding patterns in each of the $(M_1,\ M_2,\ M_3)$ binding classification groups. As the results in Table 7.3 indicate, a binding pattern threshold of 71 [i.e., $(M_1 = 4,\ M_2 = 0,\ M_3 = 0)$ or stronger] captures a large amount of translated sequences while excluding a significant number of nontranslated and random sequences.

Thus far, the functional binding statistics have characterized the binding behavior over specific position ranges. The key to defining the binary binding model (and ultimately the convolutional coding model) for translation initiation lies in the ability to capture positional binding information. Ribosomal recognition of a translation initiation site depends on more than one position in the leader sequence. Positional binding information of translated and nontranslated sequences was compared using the joint probability ratio:

$$p = \frac{P(\text{bind, position} \mid \text{translated})}{P(\text{bind, position} \mid \text{nontranslated})} \tag{7.34}$$

The probability p was calculated for positions -18 to -3 and the results (by binding pattern classification groups) are shown in Figure 7.15, where the horizontal axis is position relative to the first base of the initiation codon and the vertical axis is

TABLE 7.3 Distribution of Strongest Binding Patterns for Translated, Nontranslated, and Random Sequence Groups

(M_1, M_2, M_3)	Translated (%)	Nontranslated (%)	Random (%)
13,0,0 to 6,0,0	14.12	1.50	1.50
5,x,x	22.41	3.80	3.67
4,x,x	30.51	14.00	14.50
3,x,x	27.68	39.20	40.67
2,x,x to 0,0,0	5.27	41.50	39.67

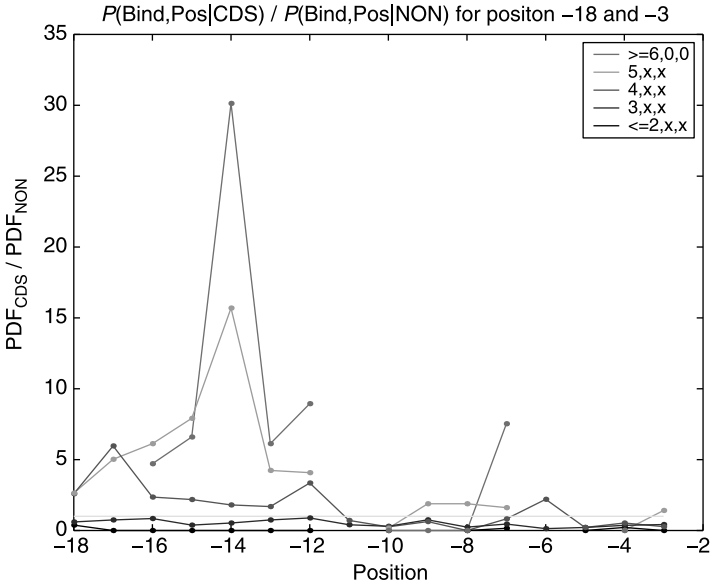


FIGURE 7.15. Positional binding ratio of translated sequence group to nontranslated sequence group.

the ratio defined in Eq. (7.34). Discontinuities are a result of dividing by zero. Ratios greater than 1 indicate positions where translated sequence binding dominates nontranslated. Ratios less than 1 indicate the opposite occurrence. Table 7.4 summarizes the key positions for each binding classification group in Figure 7.15 that achieved ratios greater than 1. Positional ratio values are used to determine weighting coefficients for horizontal motif-based convolutional codes for binary binding vectors (also used to develop horizontal motif-based base 5 convolution codes in related work [26]).

From Binding Vectors to Codewords and Inverse ECC Systems Each binding vector pattern can be considered a codeword for that position. The question becomes what coding system produced the binding vector codewords and whether

TABLE 7.4 Location of Largest Translated to Nontranslated Positional Binding Ratio Value

(M_1, M_2, M_3)	Position
13,0,0 to 6,0,0	−14
5,x,x	−14
4,x,x	−17
3,x,x	Ratio <1
2,x,x to 0,0,0	Ratio <1

the coding system follows a horizontal encoder/decoder or a vertical coding scheme. The 13-bit binding patterns present in translated sequences are viewed as codewords generated by a candidate convolutional encoder. We developed a genetic algorithm (GA) and used it to construct convolutional code models that best describe the binary codewords (i.e., the functional aspects of the translation initiation process).

Similar to previous convolution code construction methods, we use GAs to search for the optimal code (thereby inverting the ECC system) based on a fitness criterion [51]. An optimal code is defined as a code that recognizes the binary binding patterns which describe the interaction between the ribosome and the mRNA. The effectiveness of each candidate code model is evaluated using the $n - k$ decoding gmasks constructed from the candidate code (see [27, 29, 52] for a description of binary table-based coding and gmask construction). The GAs search space included all possible (n, k, m) binary convolutional codes. The population, fitness evaluation method, and genetic operators are defined based on the following objective: Locate an $(n = 3,\ k = 1,\ m = 4)$ binary convolutional code that has the greatest probability of producing the binary binding vector for each *E. coli* leader sequence in the training set. The fitness of each individual in the population (a set of potential solutions) is based on the syndrome values produced when the code's gmasks are applied to the mRNA binary binding vector sequence. A syndrome value of zero indicates that no errors within the code's error detection capability occurred. Random selection and target sampling rates are used to select highly fit individuals for reproduction. New populations are created using parameterized uniform crossover. Mutation is used to preserve population diversity and elitism ensures that the most fit solution is not discarded. The GA searches for the optimal horizontal equal weight (equal error protection) and motif-based, unequal error protection (UEP) codes for each sequence. To construct vertical-code models, the GA searches for the optimal convolutional code in each position of the leader regions' binding vectors for the entire data set.

Messenger RNA leader sequences from *E. coli* K-12 strain MG1655 (downloaded from the National Institutes of Health site ncbi.nlm.nih.gov and parsed by Rosnick [53]) are used as training sequences for constructing the best candidate code model. The syndrome distance vector for each code model is calculated and indicates how well the associated decoder recognizes the subsequence at hand. If the GA found the perfect code, the convolutional coding system that produced the exact sequence, then the syndrome distance vector would be the all-zero vector and the fitness value would be 1.

Figure 7.16 shows the average syndrome distance value for the optimal codes discovered using the *E. coli* model set. The horizontal axis is position relative to the first base in the initiation codon and the vertical axis is the average syndrome distance value. The resulting ECC binary models performed comparable to one another. The motif-based method captured the functional behavior of the ribosome binding site better than the other two models. Unlike the base 5 models [26] where all-zero parity sequences (i.e., binary binding vectors) do not occur, the binary code models are affected by all-zero binding patterns. The effects of the

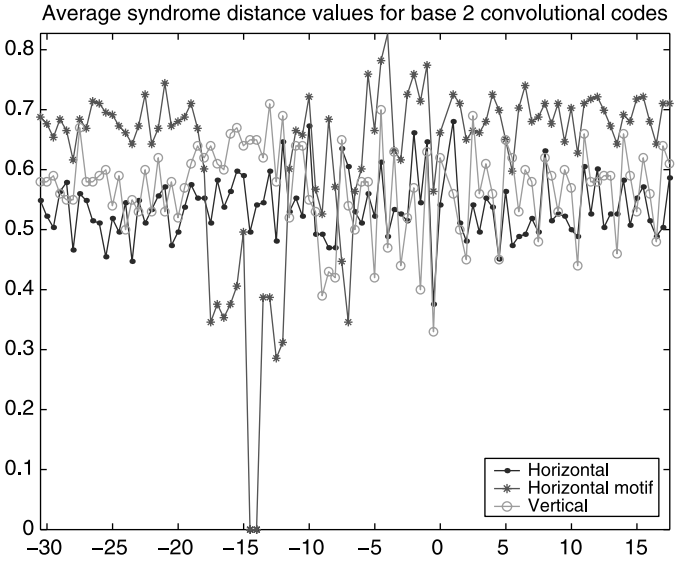


FIGURE 7.16. Average syndrome distance from all-zero syndrome for binary table-based convolutional code models for translation initiation.

all-zero parity can be minimized by using motif-based fitness measures over regions in the mRNA with greater binding affinity to the exposed portions of the 16S rRNA. Use of fitness penalties for all-zero parity sequences may improve the models. The resulting gmasks were affected, as in the base 5 case, by the table-based gmask construction method. Investigating other decoding methods and increasing the memory length of the code should improve the resulting models.

The gmask coefficients were analyzed to determine which binding regions and to what degree binding relationships between the mRNA leader sequence training set and the exposed portion of the 16S rRNA are captured by the code models. Figure 7.17 shows the average gmask values for the code models. The horizontal axis indicates bit position in the gmask vector. The vertical axis is the average value of the gmask coefficients over all codes in the model set.

For each 12-bit binary binding subpattern, the 9-bit gmask shifts twice. Each shift corresponds to binding with a different region of the last 13 bases of the 16S rRNA:

$$\text{Shift}_1 = (\ldots \text{A}\ \ \text{U}\ \ \text{U}\ \ \text{C}\ \ \text{C}\ \ \text{U}\ \ \text{C}\ \ \text{C}\ \ \text{A}\ \ldots)$$
$$\text{Shift}_2 = (\ldots \text{C}\ \ \text{C}\ \ \text{U}\ \ \text{C}\ \ \text{C}\ \ \text{A}\ \ \text{C}\ \ \text{U}\ \ \text{A}\ \ldots)$$

For the 9-bit gmask, coefficient values of 1 indicate a position on shift sequence 1 or 2 with which the mRNA leader must form a hydrogen bond. In Figure 7.17, position 7 on both gmasks and, to a slightly lesser degree, position 4 are the results of the gmask construction method used in table-based codes. Figure 7.17 indicates that the gmasks for vertical codes contain a relatively large number of zeros in many

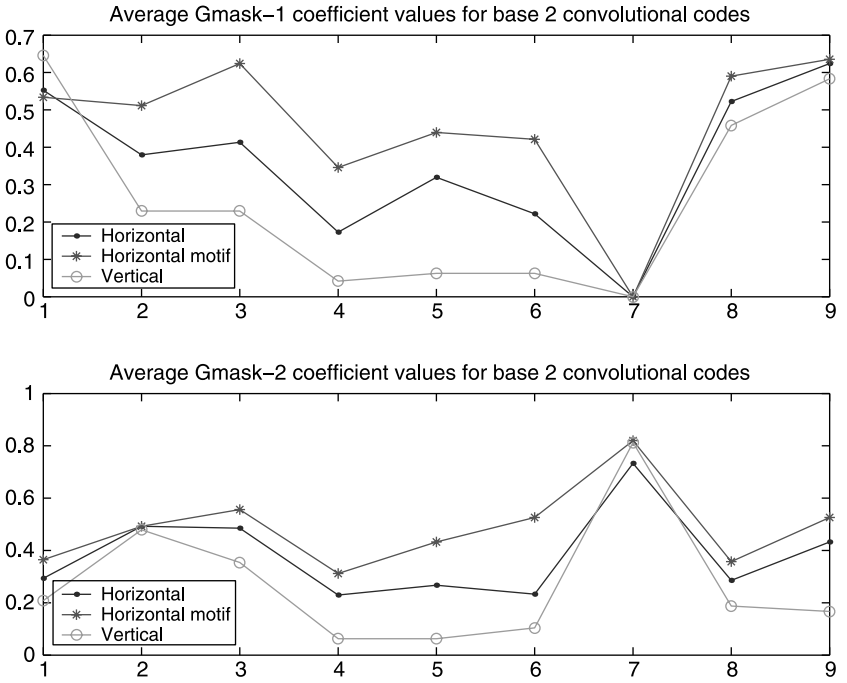


FIGURE 7.17. Average gmask values for binary table-based convolutional code models for translation initiation.

positions. Large number of zeros in the gmask of the vertical-code model probably inflated the syndrome distance performance results of the vertical-code model. Although the equal-weight horizontal code's gmasks contained fewer zeros than the vertical code, its gmasks still contained more zeros than the motif-based horizontal code's gmasks. For gmask 1, positions 1 to 3 had relatively high average coefficient values for the motif-based codes. This corresponds to the first three codons in shift 1 or 2: (A U U) or (C C U), codons complementary to regions of the Shine–Dalgarno sequence. The last two positions of gmask 2 (motif-based model) also indicated high binding, corresponding to binding with the last two bases in shift 1 or 2: (C A) or (U A). The high binding areas for gmask 2 of the motif-based horizontal-code model are positions 2 and 3 and positions 6 and 7. These positions correspond to (U U) and (U C) or (C U) and (A C).

Functional code models for protein translation initiation aid in understanding the system and can help define the binding behavior that is necessary for translation initiation. The accuracy of the model can be increased by incorporation of more specific binding information such as the number of hydrogen bonds formed per binding event. Inverse functional models can lead to improvements in the sequence-based coding models and aid in the development of algorithms for designing and improving the efficiency of transgenic leader sequences.

7.4. APPLICATIONS OF BIOLOGICAL CODING THEORY

As more researchers explore the ECC properties of genetic sequences and apply these methods to computational biology and molecular computing problems, the information- and coding-theoretic properties of genetic systems can be further understood and potentially exploited for bioengineering applications.

7.4.1. Coding Theory and Molecular Biology

Coding-theoretic methods have been used to analyze genetic sequences for various classification purposes. Arques and Michel statistically analyzed the results of 12,288 autocorrelation functions of protein-coding sequences [54]. Based on the results of the autocorrelation analysis, they identified three sets of circular codes X_0, X_1, X_2 that can be used to distinguish the three possible reading frames in a protein-coding sequence [54]. A set of codons X is a circular code, or a code without commas, if the code is able to be read in only one frame without a designated initiation signal. Crick et al. originally introduced the concept of codes without commas in the alphabet A, C, G, T. It was later successfully addressed and extracted over the alphabet R, Y, N [54]. Arques and Michel define a circular code over the A, C, G, T alphabet. They were able to use the three sets of circular codes to retrieve the correct reading frame for a given protein sequence in a 13-base window. They have used their coding-based model to analyze Kozak's scanning mechanism for eukaryotic translation initiation and other models of translation [54].

Stambuk also explored circular coding properties of nucleic acid sequences [55, 56]. His approach was based on the combinatorial necklace model, which asks: "How many different necklaces of length m can be made from a bead of q given colors" [55, 57]. Using $q = [\text{A, C, G, T}]$ and $q = [R = \text{purine}, \quad Y = \text{pyrimidine}, \quad N = R \text{ or } Y]$, Stambuk applied the necklace model to genetic sequence analysis [55]. Although Stambuk did not use ECC in his analysis, his work demonstrated the use of coding theory arithmetic in the analysis of the genetic code.

Researchers have applied source coding to genetic sequences [16, 58]. In the engineering communication system, source encoding, or data compression, occurs prior to channel coding (ECC). Source encoding removes the redundancy in the information stream to reduce the amount of symbols transmitted over the channel. The compression algorithm assigns the most frequent patterns shorter descriptions and the most infrequent patterns are assigned longer descriptions [43]. Loewenstern et al. apply source-coding methods to genomic sequences for the purpose of motif identification [16, 59, 60]. Powell et al. implemented compression schemes for finding biologically interesting sites in genomic sequences [58]. Delgrange et al. used data compression methods to locate approximate tandem repeat regions within DNA sequences [61].

7.4.2. Coding Theory and DNA Computing

The field of DNA computing was launched when Adleman solved an instance of the Hamiltonian path problem using DNA strands to encode the problem and biological

processes (annealing, ligation, etc.) to compute a solution [62]. Researchers have proposed several applications using the DNA computing framework including algorithms for breaking the data encryption standard, DNA encryption methods, and techniques to investigate nature's cellular computing processes [62–65]. In DNA computing, the information storage capability of DNA is combined with laboratory techniques that manipulate the DNA to perform computations [65]. A major challenge in DNA computing is the problem of encoding algorithms into the DNA medium. Kari and colleagues apply circular coding methods to the forward-encoding problem for DNA computing applications [65], the forward problem being how can one encode an algorithm using DNA such that one avoids undesirable folding. Kari et al. use coding theory to define heuristics for constructing codewords for DNA computing applications. The codewords cannot form undesirable bonds with itself or other codewords used or produced during the computational process. Error control and correction in DNA computing are also being investigated [66, 67].

7.4.3. Error Control Coding Methods for Genomic Sequence and System Analysis

Application of coding theory to genetic data dates back to the late 1950s [68, 69] with the deciphering of the genetic code. Since then, ECC methods have been applied to genetic sequence analysis and classification, biological chip design, as well as analysis of genetic regulatory processes. Sengupta and Tompa approach the problem of oligo array design from a combinatorial design framework and use ECC methods to increase the fidelity of oligo array construction [70]. Reif and LaBean propose ECC-based methods for the development of error correction strands for repairing errors in DNA chips [71].

Based on the discriminating behavior of their ECC model, May et al. constructed four Bayesian classifiers to distinguish between valid and invalid ribosome binding sites [49, 72]. Their classification system uses an 11-base classification window, which is a relatively small decision window compared to other classification systems [73–75]. Similar to the biological model, the error-control-based classifiers use the redundancy, or extra information, present in the mRNA leader sequence to locate valid translation initiation sites. May et al.'s optimal classification system has a correct classification rate of 73.81% with true positive and true negative rates of 67.90% and 79.72%, respectively. Their results suggest that it is highly possible to implement an ECC-based classification system for detecting and possibly designing prokaryotic translation initiation sites. Elucidating how genetic systems incorporate and use redundancy and understanding the functional significance of genetic errors from a coding theory perspective will help provide insight into the fundamental rules which govern genetic regulatory systems.

ACKNOWLEDGMENTS

The work described is a result of research performed in close collaboration with Drs. Mladen A. Vouk and Donald L. Bitzer of North Carolina State University's Computer Science

Department, Dr. David I. Rosnick (while at North Carolina State University), and Mr. David C. Schmidt, a Department of Energy Computational Sciences Graduate Fellow at the University of Illinois—UC. The author would like to thank Drs. Anna M. Johnston (formerly with Sandia National Laboratories) and William E. Hart and Jean-Paul Watson of Sandia National Laboratories for their contributions to this work.

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

REFERENCES

1. R. Roman-Roldan, P. Bernaola-Galvan, and J. L. Oliver, "Application of information theory to DNA sequence analysis: A review," *Pattern Recognition*, 29(7): 1187–1194, 1996.
2. R. Sarkar, A. B. Roy, and P. K. Sarkar, "Topological information content of genetic molecules—I," *Math. Biosci.*, 39: 299–312, 1978.
3. T. B. Fowler, "Computation as a thermodynamic process applied to biological systems," *Int. J. Bio-Med. Comput.*, 10(6): 477–489, 1979.
4. C. E. Shannon and W. Weaver, *The Mathematical Theory of Communication*, University of Illinois Press, Urbana, IL, 1949.
5. K. Palaniappan and M. E. Jernigan, "Pattern analysis of biological sequences," in *Proceedings of the 1984 IEEE International Conference on Systems, Man, and Cybernetics*, Halifax, Canada, 1984, pp. 421–426.
6. H. Almagor, "Nucleotide distribution and the recognition of coding regions in DNA sequences: An information theory approach," *J. Theor. Biol.*, 117: 127–136, 1985.
7. T. D. Schneider, "Theory of molecular machines. II. Energy dissipation from molecular machines," *J. Theor. Biol.*, 148: 125–137, 1991.
8. T. D. Schneider, "Theory of molecular machines. I. Channel capacity of molecular machines," *J. Theor. Biol.*, 148: 83–123, 1991.
9. S. F. Altschul, "Amino acid substitution matrices from an information theoretic perspective," *J. Mol. Biol.*, 219: 555–565, 1991.
10. P. Salamon and A. K. Konopka, "A maximum entropy principle for the distribution of local complexity in naturally occuring nucleotide sequences," *Comput. Chem.*, 16(2): 117–124, 1992.
11. J. L. Oliver, P. Bernaola-Galvan, J. Guerrero-Garcia, and R. Roman-Roldan, "Entropic profiles of DNA sequences through chaos-game-derived images," *J. Theor. Biol.*, 160: 457–470, 1993.
12. F. M. De La vega, C. Cerpa, and G. Guarneros, "A mutual information analysis of tRNA sequence and modification patterns distinctive of species and phylogenetic domain," in *Pacific Symposium on Biocomputing*, 1996, pp. 710–711.
13. T. D. Schneider and D. N. Mastronarde, "Fast multiple alignment of ungapped DNA sequences using information theory and a relaxation method," *Discrete Appl. Math.*, 71: 259–268, 1996.
14. B. J. Strait and T. Gregory Dewey, "The Shannon information entropy of protein sequences," *Biophys. J.*, 71: 148–155, 1996.

15. A. Pavesi, B. De Iaco, M. Ilde Granero, and A. Porati, "On the informational content of overlapping genes in prokaryotic and eukaryotic viruses," *J. Mol. Evol.*, 44(6): 625–631, 1997.
16. D. Loewenstern and P. N. Yianilos, "Significantly lower entropy estimates for natural DNA sequences," in *Proceedings of the Data Compression Conference*, Snowbird, Utah, IEEE Press, Piscataway, NJ, 1997.
17. T. D. Schneider, "Information content of individual genetic sequences," *J. Theor. Biol.*, 189: 427–441, 1997.
18. T. D. Schneider, "Measuring molecular information," *J. Theor. Biol.*, 201: 87–92, 1999.
19. L. L. Gatlin, *Information Theory and the Living System*, Columbia University Press, New York, 1972.
20. H. Yockey, *Information Theory and Molecular Biology*, Cambridge University Press, New York, 1992.
21. M. Eigen, "The origin of genetic information: Viruses as models," *Gene*, 135: 37–47, 1993.
22. P. Sweeney, *Error Control Coding: An Introduction*, Prentice-Hall, Englewood Cliffs, NJ, 1991.
23. J. B. Anderson and S. Mohan, *Source and Channel Coding: An Algorithmic Approach*, Kluwer Academic, Norwell, MA, 1991.
24. A. Dholakia, *Introduction to Convolutional Codes with Applications*, Kluwer Academic, Norwell, MA, 1994.
25. S. Lin and D. J. Costello Jr., *Error Control Coding: Fundamentals and Applications*, Prentice-Hall, Englewood Cliffs, NJ, 1983.
26. E. E. May, M. A. Vouk, D. L. Bitzer, and D. I. Rosnick, "An error-correcting code framework for genetic sequence analysis," *J. Franklin Inst.*, 341: 89–109, 2004.
27. A. Dholakia, D. L. Bitzer, and M. A. Vouk, "Table based decoding of rate one-half convolutional codes," *IEEE Trans. Commun.*, 43: 681–686, 1995.
28. D. L. Bitzer, A. Dholakia, H. Koorapaty, and M. A. Vouk, "On locally invertible rate-$1/n$ convolutional encoders," *IEEE Trans. Inform. Theory*, 44: 420–422, 1998.
29. E. E. May, "Towards a biological coding theory discipline," *New Thesis*, January 2004.
30. G. Battail, "Does information theory explain biological evolution?" *Europhys. Lett.*, 40(3): 343–348, November 1997.
31. E. May, M. Vouk, D. Bitzer, and D. Rosnick, "Coding theory based models for protein translation initiation in prokaryotic organisms," *BioSystems*, 76: 249–260, 2004.
32. B. Lewin, *Genes*, Vol. 5, Oxford University Press, New York, 1995.
33. E. Eni May, *Analysis of Coding Theory Based Models for Initiating Protein Translation in Prokaryotic Organisms*, Ph.D. Thesis, North Carolina State University, Raleigh, NC, March 2002.
34. G. Rosen and J. Moore, "Investigation of coding structure in DNA," in *ICASSP 2003*, Hong Kong, 2003.
35. J. Watson, N. Hopkins, J. Roberts, J. Steitz, and A. Weiner, *Molecular Biology of the Gene*, Benjamin Cummings Publishing Company, Menlo Park, CA, 1987.
36. E. Szathmary, "The origin of the genetic code: Amino acids as cofactors in an RNA world," *Trends Genet.*, 16(1): 17–19, 2000.
37. R. D. Knight and L. F. Landweber, "Guilt by association: The arginine case revisited," *RNA*, 6(4): 499–510, 2000.

38. L. Gold and G. Stormo, "Translational initiation," in *Escherichia coli and Salmonella typhimurium, Cellular and Molecular Biology*, in F. C. Neidhardt (Ed.), ASM Press, Washington, D.C., 1987, 1302–1307.
39. D. E. Draper, "Translation initiation," in *Escherichia coli and Salmonella, Cellular and Molecular Biology*, in F. C. Neidhardt (Ed.), ASM Press, Washington, D.C., 1996, pp. 902–908.
40. T. D. Schneider, G. D. Stormo, L. Gold, and A. Dhrenfeucht, "Information content of binding sites on nucleotide sequences," *J. Mol. Biol.*, 188: 415–431, 1986.
41. L. S. Liebovitch, Y. Tao, A. Todorov, and L. Levine, "Is there an error correcting code in DNA?" *Biophys. J.*, 71: 1539–1544, 1996.
42. D. MacDonaill, "A parity code interpretation of nucleotide alphabet composition," *Chem. Commun.*, 18: 2062–2063, 2002.
43. T. M. Cover and J. A. Thomas, *Elements of Information Theory*, Wiley, New York, 1991.
44. D. C. Schmidt and E. E. May, "Visualizing ECC properties of *E. coli* K-12 translation initiation sites," Paper presented at the Workshop on Genomic Signal Processing and Statistics, Baltimore, MD, 2004.
45. J. W. Drake, B. Charlesworth, D. Charlesworth, and J. F. Crow, "Rates of spontaneous mutation," *Genetics*, 148: 1667–1686, 1998.
46. J. W. Drake, "A constant rate of spontaneous mutation in DNA-based microbes," *Proc. Natl. Acad. Sci.*, 88: 7160–7164, 1991.
47. A. Bebenek, G. T. Carver, H. Kloos Dressman, F. A. Kadyrov, J. K. Haseman, V. Petrov, W. H. Konigsberg, J. D. Karam, and J. W. Drake, "Dissecting the fidelity of bacteriophage RB69 DNA polymerase: Site-specific modulation of fidelity by polymerase accessory proteins," *Genetics*, 162: 1003–1018, 2002.
48. E. E. May, M. A. Vouk, D. L. Bitzer, and D. I. Rosnick, "Coding model for translation in *E. coli* K-12," Paper presented at the First Joint Conference of EMBS-BMES, Atlanta, GA, 1999.
49. E. E. May, M. A. Vouk, D. L. Bitzer, and D. I. Rosnick, "Coding theory based models for protein translation initiation in prokaryotic organisms," *BioSystems*, 76: 249–260, 2004.
50. E. E. May, "Comparative analysis of information based models for initiating protein translation in *Escherichia coli* K-12," M.S. Thesis, North Carolina State University, December 1998.
51. R. Kotrys and P. Remlein, "The genetic algorithm used in search of the good TCM codes," Paper presented at the Fourth International Workshop on Systems, Signals and Image Processing, IWSSIP'97, Poznan, Poland, 1997, pp. 53–57.
52. D. L. Bitzer and M. A. Vouk, "A table-driven (feedback) decoder," Paper presented at the Tenth Annual International Phoenix Conference on Computers and Communications, 1991, Phoenix, Arizona, pp. 385–392.
53. D. I. Rosnick, *Free Energy Periodicity and Memory Model for E. coli Codings*, Ph.D. Thesis, North Carolina State University, Raleigh, NC, 2001.
54. D. G. Arques and C. J. Michel, "A code in the protein coding genes," *BioSystems*, 44: 107–134, 1997.
55. N. Stambuk, "On circular coding properties of gene and protein sequences," *Croatica Chem. Acta*, 72(4): 999–1008, 1999.
56. N. Stambuk, "Symbolic Cantor Algorithm (SCA): A method for analysis of gene and protein coding," *Periodicum Biologorum*, 101(4): 355–361, 1999.

57. E. R. Berlekamp, *Algebraic Coding Theory*, McGraw-Hill Book Company, New York, 1968.
58. D. R. Powell, D. L. Dowe, L. Allison L, and T. I. Dix, "Discovering simple DNA sequences by compression," Paper presented at the Pacific Symposium on Biocomputing, Maui, Hawaii, 1998, pp. 597–608.
59. D. M. Loewenstern, H. M. Berman, and H. Hirsh, "Maximum a posteriori classification of DNA structure from sequence information," Paper presented at the Pacific Symposium on Biocomputing, Maui, Hawaii, 1998, pp. 669–680.
60. D. Loewenstern and P. N. Yianilos, "Significantly lower entropy estimates for natural DNA sequences," *J. Comput. Biol.*, 6(1): 125–142, 1999.
61. O. Delgrange, M. Dauchet, and E. Rivals, "Location of repetitive regions in sequences by optimizing a compression method," Paper presented at the Pacific Symposium on Biocomputing, Big Island, Hawaii, 1999, pp. 254–265.
62. C. C. Maley, "DNA computation: Theory, practice, and prospects," *Evol. Comput.*, 6(3): 201–229, 1998.
63. L. Adleman, P. Rothemund, S. Roweis, and E. Winfree, "On applying molecular computation to the data encryption standard," *J. Comput. Biol.*, 6(1): 53–63, 1999.
64. L. F. Landweber and L. Kari, "The evolution of cellular computing: Nature's solution to a computational problem," *BioSystems*, 52: 3–13, 1999.
65. L. Kari, J. Kari, and L. F. Landweber, "Reversible molecular computation in ciliates," in J. Karhumaki, H. Maurer, G. Paun, and G. Rozenberg (Eds.), *Jewels are Forever*, Springer-Verlag, Berlin, 1999, pp. 353–363.
66. D. Boneh, C. Dunworth, R. J. Lipton, and J. Sgall, "Making DNA computers error resistant," in L. F. Landweber and E. B. Baum (Eds.), *DNA Based Computers II*, American Mathematical Society, Providence, RI, 44: 165–172, 1998.
67. D. H. Wood, "Applying error correcting codes to DNA computing," Paper presented at the Fourth DIMACS Workshop on DNA Based Computers, Philadelphia, 1998.
68. B. Hayes, "The invention of the genetic code," *Am. Sci.*, 86(1): 8–14, 1998.
69. S. W. Golomb, "Efficient coding for the deoxyribonucleic channel," *Proc. Symp. Appl. Math.*, 14: 87–100, 1962.
70. R. Sengupta and M. Tompa, "Quality control in manufacturing oligo arrays: A combinatorial design approach," *J. Comput. Biol.*, 9(1): 1–22, 2002.
71. J. Reif and T. LaBean, "Computationally inspired biotechnologies: Improved DNA synthesis and associative search using error-correcting codes and vector-quantization," Paper presented at DNA Computing: Sixth International Meeting on DNA-Based Computers, Leiden, Netherlands, 2000.
72. E. E. May, M. A. Vouk, D. L. Bitzer, and D. I. Rosnick, "Coding theory based maximum-likelihood classification of translation initiation regions in *Escherichia coli* K-12," Paper presented at the 2000 Biomedical Engineering Society Annual Meeting, Seattle, WA, 2000.
73. J. Henderson, S. Salzberg, and K. H. Fasman, "Finding genes in DNA with a hidden markov model," *J. Comput. Biol.*, 4(2): 127–1441, 1997.
74. A. Krogh, I. Saira Mian, and D. Haussler, "A hidden Markov model that finds genes in *E. coli* DNA," *Nucleic Acids Res.*, 22(22): 4768–4778, 1994.
75. M. Borodovsky and J. McIninch, "GENMARK: Parallel gene recognition for both DNA strands," *Comput. Chem.*, 17(2): 123–133, 1993.

CHAPTER 8

Complex Life Science Multidatabase Queries

ZINA BEN MILED, NIANHUA LI, YUE HE, MALIKA MAHOUI, and OMRAN BUKHRES

8.1. INTRODUCTION

The confederation of widely distributed life science databases is a critical scientific problem which lies at the foundation of the nascent field of proteomics and of biological and biomedical phenotype analysis. Databases are the intermediaries between experimental observation and our ability to synthesize larger scale understanding of biological systems and the manifold interactions in which they participate and to which they respond. Unfortunately, the relevant databases have been organized around disciplinary interests, subject areas, or institutional convenience.

Collins et al. state, "The central information technology problems of the next decade will be the creation of a means through which to query a heterogeneous set of life science databases, generally via the Internet" [1, p. 688]. In order to query multiple life science databases, support for the interoperability among these databases should be provided. Facilitating the interoperability of heterogeneous, autonomous, geographically distributed, and/or semistructured Web databases (e.g., life science databases) is currently an emerging and active area of research [2–5]. The challenges arise from the large volume of the life science data, the rate at which these data are increasing in size, and the heterogeneity and complexity of the data format. There are also hundreds of life science databases which use different nomenclatures, file formats, and data access interfaces.

There are three different approaches to addressing the issue of interoperability among biological databases: data warehousing, link driven, and federated databases.

In data warehousing, remote databases are copied on a local server and a unique interface is built to allow multidatabase queries to be issued using this single interface. Data warehousing is based on a thorough understanding of the data schema of

Genomics and Proteomics Engineering in Medicine and Biology. Edited by Metin Akay

each individual database included in the data warehouse. Specification of the data schema is not often published for life science databases. Furthermore, as data continue to grow, data warehousing becomes impractical for biological studies that extend beyond a few databases. Several existing projects perform data integration by use of data warehousing such as Genomics Unified Schema (GUS) [6].

The link-driven approach is based on providing static links between data or records in different databases. This approach often suffers from the fact that many cross references between various records in the databases are unidirectional. Additionally, the queries that can be issued by the users are limited to the scope predefined by the static links. The link-driven approach is used by several life science systems such as Sequence Retrival System (SRS) [7] and the National Center for Biotechnology Information (NCBI)/Entrez [8], which, for example, links sequence data from NCBI [9] to structure data from the Protein Data Bank (PDB) [10]. One of the major advantages of this approach is its ease of implementation.

Database federation has been studied in the context of relational and homogeneous databases [11] (e.g., business databases). More recently, TAMBIS [12], BioKleisli [13], and DiscoveryLink [14] have extended this approach to the life sciences. Federated databases provide a unified portal for the data in all the databases participating in the integration while preserving the local autonomy of these databases. This approach is based on the semantic understanding of the relationships between different objects in different databases. This feature is particularly important in the life sciences because of the complexity of the domain. Furthermore, there is no limit to the scope of the queries that can be issued against federated databases.

For each of the above-mentioned three approaches, a selected system that represents one of the leading efforts in the field is described next. The selected systems are GUS for data warehousing, SRS for link driven, and TAMBIS for federated databases. There are several other systems that aim at providing the users with a single-access interface to multiple biological databases. Some of these systems are discussed in Section 8.4.

GUS uses a data-warehousing approach to support the interoperability of life science databases. Sequences and their associated annotation are stored in GUS. These data are obtained from several sequence databases such as SwissProt and stored locally using a relational database model. The design of GUS addresses one of the major issues associated with data warehousing, which is the automatic update of the data warehouse. This process includes detecting when changes occur in the databases and subsequently how to initiate the necessary updates automatically in the data warehouse.

SRS links several biological data sources and answers queries through a single front-end Web interface. The data sources are not transparent to the user in SRS. The user must specify which data sources should be used in order to answer a given query. This lack of transparency requires that the user be familiar with the various data sources integrated by SRS and their content. Data sources in SRS are semistructured flat files. Each of the data files is associated with a wrapper that extracts information in the form of objects from the corresponding data source. SRS also includes metadata that describe the various data sources.

TAMBIS integrates a specific set of life science databases that consists of protein and nucleic acid data sources. It is based on a global domain ontology, which consists of more than 1800 biological concepts. These concepts are used to express the multidatabase queries and to decompose these queries into subqueries that can be mapped to specific data sources. Unlike SRS, the integrated databases are transparent to the user.

TAMBIS uses the federated database approach to support the interoperability among biological databases. As mentioned earlier, database federation benefits from several enabling capabilities. However, these capabilities come at the expense of a complex design. One of these complexities arises from answering multidatabase queries given the limited query capabilities of the individual databases participating in the integration.

Consider, for example, the following query: *What is the 3D structure of all alcohol dehydrogenases that belong to the enzyme family EC 1.1.1.1 and is located within the human chromosome region 4q21–4q23?* This example query, among others, motivated the design of BACIIS [15], the Biological and Chemical Information Integration System being discussed in this chapter. BACIIS uses a federated database approach similar to TAMBIS. The above example query is difficult because some of the integrated databases may not support enzyme family as an acceptable keyword while others may not allow queries that use the logical AND operation to combine the two keywords *enzyme family* and *human chromosome region*. Additionally, some life science federated database systems such as DiscoveryLink require that the user enter all the databases that may be needed to answer a given multidatabase query. As will be discussed later in this chapter, the execution plans of complex life science queries may include intermediary databases that cannot directly accept the input or the output constraints of the user query. These intermediary databases are used to translate the attributes of one database to the attributes of another database. For example, SwissProt [16] can accept Online Mendelian Inheritance in Man (OMIM) [17] numbers and generate PDB IDs. Requiring that the user enters all the databases that may be involved in executing a query is not practical for life science databases since the user may not be familiar with the numerous biological databases (more than 400) that are available. BACIIS uses an approach that allows the execution of this query and other complex queries in a completely transparent manner while hiding the complexities from the user and preserving the autonomy of each database participating in the integration.

The aim of this chapter is to (1) introduce BACIIS and its functionality, (2) present an efficient query execution method that can be used to execute complex multidatabase queries while preserving the local autonomy of the geographically distributed life science Web databases and while hiding the complexity of the query execution plan from the user, and (3) expose the reader to other approaches for the integration of life science databases.

The general architecture of BACIIS is introduced in Section 8.2. The mapping engine and the query planner are the main two components of BACIIS that are involved in the execution of the multidatabase queries. These components are

discussed in Section 8.3. Related work is summarized in Section 8.4. Future trends are reviewed in Section 8.5.

8.2. ARCHITECTURE

Figure 8.1 shows the general three-tier architecture of BACIIS. This architecture hides the heterogeneity of the various databases from the user through the use of a unified user interface. In order for a user to accomplish the same results provided by BACIIS, he or she would have to perform two tasks manually: (1) query the individual Web databases manually and (2) combine the returned information. In BACIIS, the wrappers (Fig. 8.1) act as intelligent proxy users in order to extract information from the individual Web databases. They perform the first task automatically on behalf of the user while the information integration layer (Fig. 8.1) performs the second task also automatically.

The various components of the information integration layer are shown in Figure 8.2. At the core of the information integration layer is a mediator. The mediator transforms the data from its format in the source database to a common format used for all the databases within the integration layer. The use of a mediator in the information integration system is not unique to BACIIS. Mediators have been used in various previous federated database systems [18–20]. The unique feature of BACIIS is that it uses a knowledge base to guide multidatabase query formulation and execution. In BACIIS, each remote database schema is mapped onto a domain knowledge base. This domain knowledge base is independent from the data schema of the remote databases. It will only change when the biological domain evolves to include new discoveries and concepts.

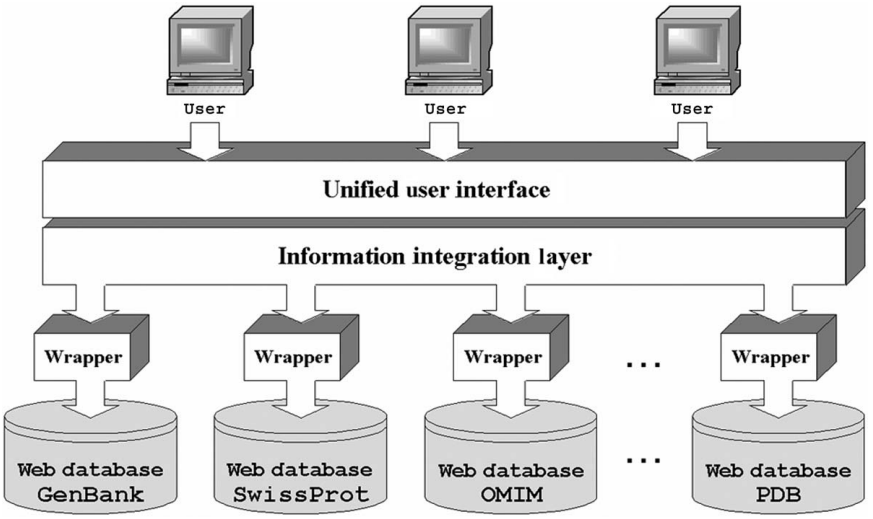


FIGURE 8.1. BACIIS three-tier architecture.

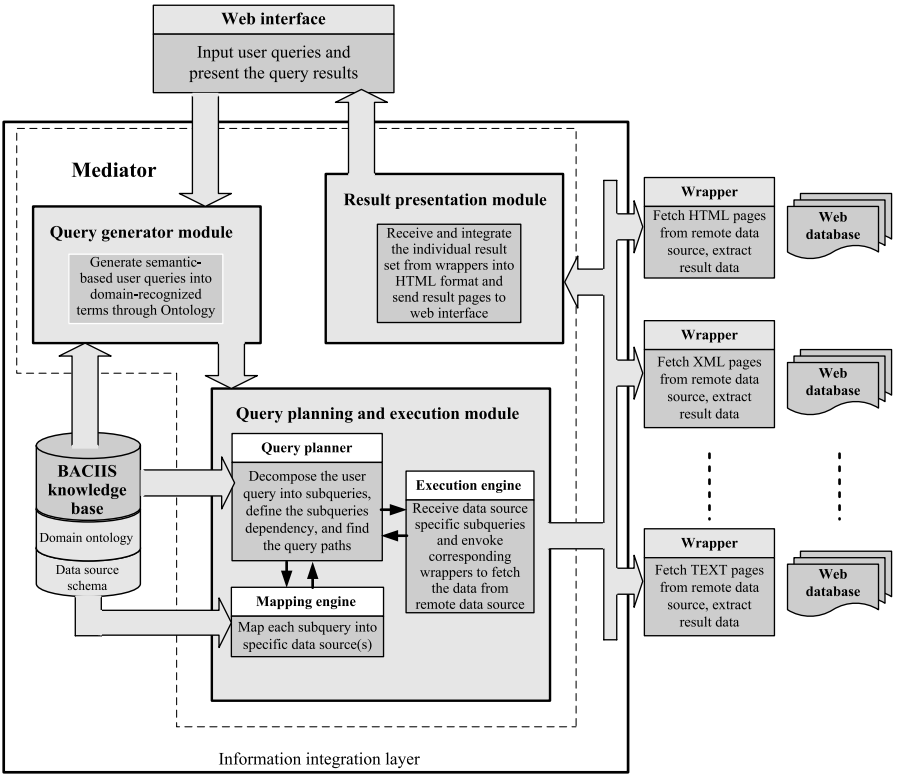


FIGURE 8.2. Architecture of information integration layer.

The BACIIS knowledge base consists of an ontology [21] and the data source schema. The ontology is necessary for terminology consensus across the various databases within a domain. Furthermore, it provides the domain knowledge used for understanding concepts in the biological domain. Under the BACIIS knowledge base, specific data source models are defined for each database participating in the integration. These data source models describe the content of each database using the objects and the relations defined by the ontology.

The ontology is also used in the query generator (Fig. 8.2) module of the mediator to translate the user queries into domain-recognized terms from the ontology. Queries generated by the query generator module are forwarded to the query planning and execution module (Fig. 8.2) of the mediator. This module consists of the query planner, the mapping engine, and the execution engine. The query planner receives the queries generated by the query generator module and decomposes them into subqueries according to the ontology properties and relations. The query planner also defines the dependency among the subqueries. The mapping engine is used to formulate these subqueries and to determine which specific source databases contain the desired information. Based on the mapping found by

the mapping engine, the query planner will then select the query execution paths consisting of a sequence of subqueries and databases that are combined to execute the overall query submitted by the user.

The result and presentation module receives individual subquery results from the wrappers, translates them by using a data model based on the ontology relations, integrates them, and generates an HTML output file for presentation to the user.

A major research issue in using the federated database approach to integrate life science Web databases is the generation of an efficient execution plan that is not guided by the user. For example, SRS requires the user to specify the databases that have to be used to answer a given query. This may not be practical because it requires a knowledge of all the available databases, their query capabilities, and their relationship to other databases. In BACIIS this information is stored in the knowledge base (ontology and data source models) and thus the user does not need to provide it when he or she enters a complex query.

8.3. QUERY EXECUTION PLANS

Query execution plans are generated by the query planning and execution module in the mediator of BACIIS. This process and the challenges associated with this process are illustrated in this section by using the example query presented in Section 1, which is repeated here for convenience: What is the 3D structure of all alcohol dehydrogenases that belong to the enzyme family EC 1.1.1.1 and is located within the human chromosome region 4q21–4q23? This query is entered in the BACIIS interface as follows, where the Boolean operators that combine the clauses are underlined:

```
   [(alcohol dehydrogenase AND (NOT NADP))
OR (E.C. number = 1.1.1.1)] AND [cytogenetic
            region = 4q21-4q23].
```

The BACIIS interface uses the query-by-example approach, which simplifies the entry of complex queries such as the above query. The interface uses the ontology to guide the user in formulating the query. In the above example query, the clauses (alcohol dehydrogenase AND (NOT NADP) and (E.C. number = 1.1.1.1) are equivalent in biological terms. They are both included in the query and combined with the logical operator OR in order to provide more coverage in the returned result. Some of the records may be excluded if one of the clauses is omitted. Also, the clause (NOT NADP) is combined with alcohol dehydrogenase by using the AND operator in order to exclude the records that belong to the enzyme family with E.C. number = 1.1.1.2 . Internally in BACIIS the query is interpreted by using the left anchored operator precedence.

The goal of BACIIS is to answer multidatabase queries without any intervention from the user. Achieving this goal is hindered by the limited query capabilities of some of the life science Web databases. First, several Web databases may not

allow the use of the Boolean operators AND, OR, and NOT. Second, life science Web databases may not allow queries based on the BACIIS ontology terms. For example, the above query contains two terms, *enzyme family* and *cytogenetic region*, that may not be accepted by life science Web databases. For instance, GenBank [22] does not accept *enzyme family* and PDB does not accept *cytogenetic region*. Although some of these life science databases have limited capabilities, they contain relevant information and therefore cannot be ignored.

8.3.1. Query Decomposition

Query decomposition transforms a complex query into subqueries. This is accomplished by breaking the query string into substrings along the partitions of logical operators. The result of the query decomposition tree for the example query introduced in Section 8.1 is shown in Figure 8.3.

In this figure, node 1 is processed and all the paths originating with the databases that can accept the query in node 1 (i.e., [(alcohol dehydrogenase AND (NOT NADP)) OR (E.C. number = 1.1.1.1)] AND [cytogenetic region = 4q21–4q23]) and terminating with the databases that can provide *protein-3D-structure* are determined. It is very unlikely that a single life science database can accept the query of node 1. Therefore, this query needs to be decomposed further. Without an information integration system for life science databases, the user would need to

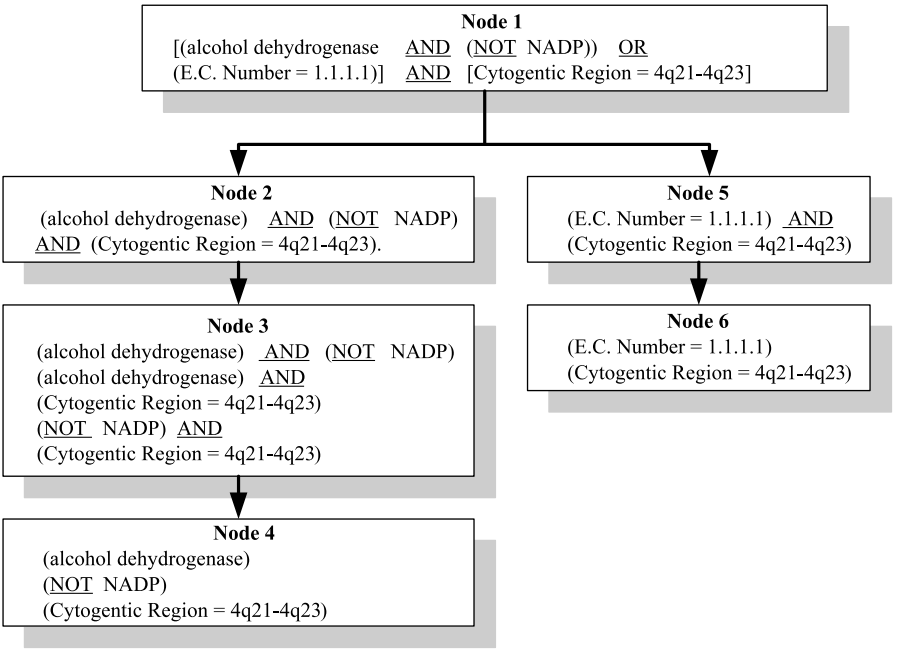


FIGURE 8.3. Query decomposition tree.

decompose the query manually, process the various subqueries, and combine the results of these subqueries. This fact is one of the motivations underlying the need for the integration of life science databases. The query of node 1 is decomposed along the OR operator into two nodes: node 2 (i.e., [alcohol dehydrogenase AND (NOT NADP)] AND [cytogenetic region = 4q21–4q23]) and node 5 (i.e., [E.C. number = 1.1.1.1] AND [cytogenetic region = 4q21–4q23]). For these nodes also, the paths that start with databases that can accept the two subqueries and terminate in a database that can generate protein-3D-structure are identified. The goal of query decomposition is to explore all the possible execution paths for the original query submitted by the user. Each of the nodes 2 and 5 will result in a set of paths. If the final execution plan includes these nodes, a given path from the first set will be combined with a path from the second set. Since the union (OR) of nodes 2 and 5 forms the original query, the union of the data sets resulting from the first and second paths is computed in order to collect the resulting data set for the original query. The union is performed by the information integration layer in BACIIS because of the limited query capabilities of the individual life science databases. Similarly, if two subqueries are joined using the AND operator, then the information integration layer in BACIIS will compute the intersection of the resulting data sets.

The decomposition process continues until nodes with simple subqueries (i.e., one input constraint with no operators) are obtained. For example, the query decomposition terminates in Figure 8.3 with nodes 4 and 6. Each of these nodes contains a set of simple subqueries.

The first step in the query processing is the decomposition (Fig. 8.3). Once the query decomposition tree is obtained, the second step consists of identifying all the databases that can accept the original query (node 1) or any of its subqueries (nodes 2 through 6) and provide the desired query output (i.e., protein-3D-structure for the example query). For example, in order to process node 3, the databases that can accept either one of the following subqueries must be determined:

- alcohol dehydrogenase AND (NOT NADP)
- alcohol dehydrogenase AND cytogenetic region = 4q21–4q23
- (NOT NADP) AND cytogenetic region = 4q21–4q23

The third step consists of combining the result of the previous step into paths that originate from the databases that can service any of the above queries and terminate in a final database that contains the query result (i.e., protein-3D-structure information for the example query). In the life sciences, this step requires a new approach that differs from traditional databases. The second and third steps in the query processing of life science multidatabase queries are described in the following sections.

8.3.2. Database List Generation

The query planner in BACIIS uses two lists: an input database list and an output database list. For each query these two lists are created based on the knowledge base of BACIIS. The input list contains all the databases that can accept the original

query or any of the subqueries in the nodes of the query decomposition tree. The output list consists of the databases that can generate the result requested by the user. For the example query, this result would be protein-3D-structure information. The only database that can provide this information in BACIIS is PDB. Therefore, the output database list for this query contains only PDB. However, for the same query, the input database list is more complex and includes several databases.

Each of the nodes in the query decomposition tree will result in a set of databases that will be included in the input database list. For example, node 6 in Figure 8.3 has the following two subqueries:

- Cytogenetic region = 4q21–4q23
- E.C. number = 1.1.1.1

For this node, two database sets will be created. One set contains all the databases that can accept *cytogenetic region* as an input such as Genome Database (GDB), NCBI-Genome, and OMIM. The other set contains all the databases that can accept *E.C. number* as input such as Enzyme Nomenclature [23], PIR, and PDB. All the combinations of two databases (e.g., GDB/Enzyme Nomenclature, GDB/PIR, GDB/PDB, NCBI-Genome/Enzyme Nomenclature, NCBI-Genome/PIR) for these sets are enumerated and used as a potential origin for a partial execution path. The combination is used because the two subqueries are related with the AND operator.

8.3.3. Path Generation

Because the example query introduced in Section 8.1 is a complex query, the following discussion is limited to the subqueries expressed by node 6 in Figure 8.3. However, the process can be similarly applied to the remaining nodes of the query decomposition tree of Figure 8.3. All the possible paths from the input constraints (i.e., cytogenetic region = 4q21–4q23 and E.C. number = 1.1.1.1) to the output protein-3D-structure for node 6 are shown in Figure 8.4. These paths are labeled A1 through A4 for the first input constraint (i.e., cytogenetic region = 4q21–4q23) and B1 through B4 for the second input constraint (i.e., E.C. number = 1.1.1.1). For example, in path A1, the input cytogenetic region = 4q21–4q23 is fed to the GDB database, which returns PDB IDs.

As mentioned in Section 8.3.1 the input database list contains the following list of nine database combinations where each combination contains two databases: GDB/Enzyme Nomenclature, GDB/PIR, GDB/PDB, NCBI-Genome/Enzyme Nomenclature, NCBI-Genome/PIR, and so on. The first database in each combination can accept the query cytogenetic region = 4q21–4q23 as an input and the second database can accept the query E.C. number = 1.1.1.1. The output protein-3D-structure can only be generated by the PDB database in BACIIS. However, PDB cannot be searched by either cytogenetic region or E.C. number.

The query planner in BACIIS allows the querying of PDB by cytogenetic region and E.C. number. This process is accomplished by constructing paths originating

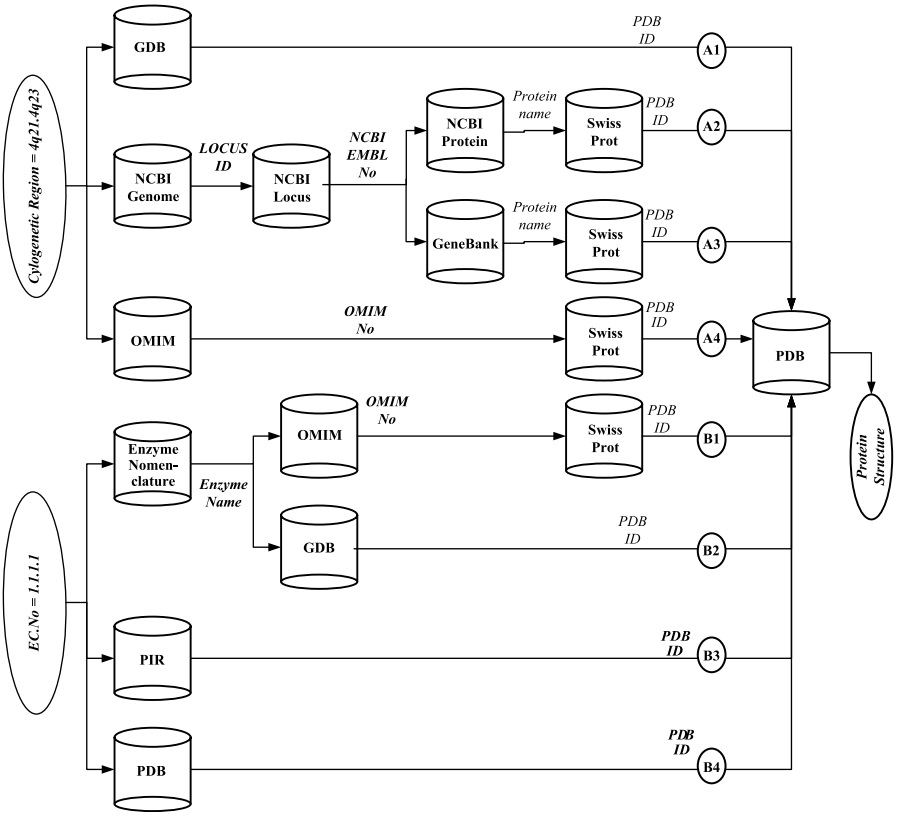


FIGURE 8.4. All possible query paths for subqueries of node 6.

with the combination of the databases in the input database list, traveling through intermediate databases, and discovering common PDB acceptable input constraints between the paths. These paths are constructed by combining a path from the set A1 through A4 and a path from the set B1 through B4 since any one of the paths A1 through A4 accepts cytogenetic region = 4q21–4q23 as input and generates protein-3D-structure and any one of the paths B1 through B4 accepts E.C. number = 1.1.1.1 and also generates protein-3D-structure. For example, paths A1 and B4 correspond to the input database combination GDB/PDB. The result of A1 consists of a set of PDB IDs and the result of B4 also corresponds to a set of PDB IDs. The intersection of these two sets is computed to generate the result of the following subquery: *3D-protein-structure information for (E.C. number = 1.1.1.1) AND (cytogenetic region = 4q21–4q23).*

Figure 8.4 illustrates why it may be impractical to require that the user guides the query processing by supplying the databases that may be involved in the query. Figure 8.4 contains 10 different databases. Thus, the user must be familiar with each of these databases and their query capabilities. Furthermore, since multiple combinations of paths from A1 through A4 and B1 through B4 can generate the

required result, the user must also be able to choose the appropriate set of databases in order for the integration system to generate a fast and efficient plan. For example in Figure 8.4, the execution plan consisting of A1 and B4 is more efficient and has a much lower response time than the execution plan consisting of A2 and B1.

The problem of deriving the path from each combination of input databases to output databases can be solved through a general STRIPS-like planner [24, 25] such as GraphPlan [26]. Traditionally, in a query planner, paths are discovered starting from the input database list and ending in the output database list. The modified planner in BACIIS, however, starts from the output database list and works backward to reach the input list. As mentioned in Section 8.3.1, all the subqueries within a node of the query decomposition tree are linked by logical AND (conjunctive). Therefore, all the databases in each input database combinations must be considered in the query plan because each combination corresponds to one subquery. For example, there are nine combinations of two databases (e.g., GDB/Enzyme Nomenclature, GDB/PIR, GDB/PDB, NCBI-Genome/Enzyme Nomenclature, NCBI-Genome/PIR) for node 6, as shown in Figure 8.3. The first database of each combination corresponds to cytogenetic region = 4q21–4q23, and the second corresponds to E.C. number = 1.1.1.1. When generating the query plan for each combination, both databases must be considered because both subqueries cytogenetic region = 4q21–4q23 and E.C. number = 1.1.1.1. must be satisfied by the query result.

When using STRIPS-like planning, *initial states* are essential but not necessary. In another words, some initial states may not be used in the plan. Therefore, this planning method cannot be used directly since all the initial states which represent the query clauses must be included. To circumvent this limitation, the BACIIS path generation problem is mapped to the STRIPS-like planning domain in a reverse manner. That is, the planning is expressed as a path search problem from the goals to the initial states rather than from the initial states to the goals.

This modified STRIPS-like planner-based approach can generate an adequate query execution plan. For instance, the plan between the input database combination OMIM/Enzyme Nomenclature and the output database PDB may consist of paths A4 and B1 of Figure 8.4. However, by using the traditional STRIPS-like planning algorithm, the planner will merge the data retrieval step from the OMIM database in the two paths to reduce the total number of steps and operators in the final plan (Fig. 8.5), which is not desired in this case.

8.4. RELATED WORK

As previously mentioned, there are three main methods that are used to integrate life science databases: data warehousing, link driven, and database federation.

SeqStore [27], Gene Expression Datamart [28], Incyte [29], and GUS utilize the data-warehousing or data mart approach (a data mart is a "small" warehouse designed to support a specific activity). This approach allows the systems to have a better performance, higher quality of data, and fewer dependencies on network connectivity. Therefore, these systems are suitable for projects involving

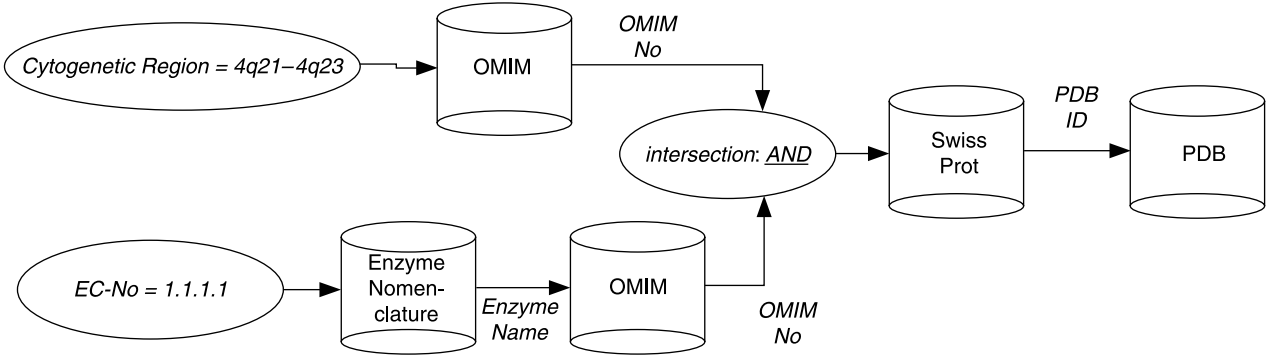


FIGURE 8.5. Efficient query execution plan that uses early intersection.

"production strength," especially those involving annotation, where more control over the data is required to guarantee correctness. However, data warehouses are often expensive to initialize and maintain. It is also difficult to keep the data up to date. For example, GUS keeps a large schema consisting of over 180 tables for the integration of sequence and annotation information from three databases: GenBank/EMBL/DDBJ, dbEST, and SwissProt. The data provided by GUS lags by a couple of months. In contrast, BACIIS and other federated database systems, such as TAMBIS, provide access to up-to-date information.

Link-driven systems, such as SRS, NCBI/Entrez, LinkDB [30], and GeneCards [31], combine the indexes of several source databases into a link table. Links are established between specific entries in the different databases. Whenever one data source entry is returned for a given user query, hyperlinks are retrieved from the link table and are made available on the result page. The link-driven approach often does not make any attempts to reconcile any semantic heterogeneity between the different databases during query execution. The static links inherent to the link-driven approach make it difficult to maintain and to scale up.

As in BACIIS, TAMBIS, BioKleisli, OPM [32], and DiscoveryLink utilize the federated database approach. This approach can be further divided into two subclasses: tightly coupled and loosely coupled. A tightly coupled database federation includes a global data schema onto which queries are expressed [33]. A loosely coupled database federation does not rely on a global data schema, and the source databases must be specified in the query by using a multidatabase query language [34]. As previously mentioned, this requirement is not trivial given the large number of available biological databases.

DiscoveryLink is the implementation of Garlic [35] for biological databases. It is a loosely coupled database federation for the integration of traditional databases rather than Web databases. BioKleisli is another example of a loosely coupled database federation. It is an application of the CPL-Kleisli system over data sources critical to the Human Genome Project. User queries are constructed by using the Collection Programming Language (CPL). Databases are accessed through Kleisli drivers. Kleisli provides drivers for different types of databases, including Sybase, Oracle, ASN.1-Entrez [36], OPM, AceDB [37], and BLAST [38]. As in the case of DiscoveryLink, when using BioKleisli, the databases involved in the query execution must be specified by the user.

TAMBIS is built on top of BioKleisli and uses a tightly coupled database federation approach which is similar to BACIIS. User queries in TAMBIS are constructed using ontology terms. The domain ontology approach allows TAMBIS and BACIIS to be queried based on biological knowledge rather than the knowledge of the query capabilities of the specific databases. In both systems, the limited query capabilities of the databases participating in the integration are transparent to the user.

One of the main differences between BACIIS and TAMBIS is the approach used to generate the query execution plans. The approach used by TAMBIS can lead to query plans that may not be executable. Consider the following query: *Provide the names of all the human proteins which belong to the enzyme family with E.C. number = 1.1.1.1*. This query contains two clauses: (1) *organism_name = human* and (2) *E.C. number = 1.1.1.1*. For the first clause, TAMBIS generates a

plan that enters the first clause in SwissProt and retrieves SwissProt entry names. For the second clause, TAMBIS uses E.C. number = 1.1.1.1 as input to the Enzyme Nomenclature database and retrieves Enzyme Nomenclature entry names. The final query plan cannot be constructed in this case because there is no correspondence between the SwissProt entry names and the Enzyme Nomenclature entry names. Thus, the intersection of the result data sets from the two paths cannot be performed. This happens because in TAMBIS the attributes (e.g., Enzyme Nomenclature entry name and SwissProt entry name) of an object (e.g., *protein*) are mapped to the object itself and the object is often associated with a single database. This example illustrates further the difficulty resulting from the heterogeneity of the database when integrating life science databases. In BACIIS, this situation does not occur because BACIIS does not perform this mapping and uses the attributes of the objects directly when generating a query execution plan. This, however, makes the query planner in BACIIS more complex than the query planner used in TAMBIS, which of course may have an impact on the query execution time. This issue is discussed in the next section as it represents one of the important future trends in the field.

8.5. FUTURE TRENDS

In this chapter, a federated database system (BACIIS) for life science Web databases was introduced. This system allows users to submit multidatabase queries against a set of heterogeneous, autonomous, geographically distributed Web databases. BACIIS uses an efficient query planner to generate an execution plan for complex life science multidatabase queries. This planner addresses the limited capabilities of the Web databases by using query decomposition, relying on support of the knowledge base in BACIIS, and by augmenting the execution engine of BACIIS with computational capabilities that replace some of the missing functionalities of the Web databases.

One of the weaknesses of the federated database approach such as the one used in BACIIS is its lengthy response time, which is due to the multiple and repeated remote access to the Web databases. Long response times are often the reason why data warehouses trade the currency of the returned data for faster response times. It may be possible to address long response times by taking into consideration access costs when generating query execution plans. This will penalize Web databases with long response times and exclude them from the query execution plan if there are alternate databases that can generate the result of the query faster.

Because of the above weakness, there is a recent trend for federated database systems to attach a cost model to query planning [34, 39]. Most cost models estimate data source searching costs according to data source delays and resulting cardinality. In TAMBIS, for example, the execution cost of a data retrieval method is estimated using the formula $t \times n^2$, where t and n are the predicted query execution time and the number of instances returned, respectively. DiscoveryLink and its precursor Garlic provide a more elaborate cost model [39]. Costs are estimated within the

global optimizer as well as within the individual wrappers for the integrated data sources. Within the wrapper, statistics, such as the base cardinality and distribution of data values, are collected to estimate the total cost (i.e., the cost in seconds for executing the operator once) and the reexecution cost (i.e., the cost for executing the operator a second time). Within the global optimizer, the previously mentioned statistics are used to estimate the central processing unit and input–output costs for invoking the wrappers as well as the selectivity of the data sources.

In BACIIS, the cost model is based on remote database query processing time, network communication time, and mediator data integration time. Furthermore, the cost model used in BACIIS allows the user to indicate if data quality should be traded for the query response time.

It should be noted that for database federation the optimization of the query execution cost has different objectives depending on whether the federation is loosely or tightly coupled. In loosely coupled federation, data sources are prespecified in the queries. Therefore, the optimization focuses on reducing the execution cost of operators such as *join* and *sort* [40]. The planning problem for tightly coupled database federation is more complex because of the choice of the source databases. Another dimension to the optimization of the query execution cost is highlighted in the planning algorithm proposed by Eckman et al. [34]. This algorithm considers the semantic equivalence of possible plans in addition to their execution costs. Biological databases have various query capabilities, data domains, and data quality levels. Therefore, the same query may retrieve different result sets if different data sources are selected, which suggests that semantic equivalence has to also be taken into account. However, a complete set of standards for biological databases need to be defined before semantic estimation can be possible when optimizing the query execution cost of the queries submitted to an integration system for life science databases.

ACKNOWLEDGMENT

This project was supported in part by the National Science Foundation, CAREER DBI-DBI-0133946 and NSF DBI-0110854.

REFERENCES

1. F. S. Collins, A. Patrions, E. Jordan, A. Chakravarti, R. Getsland, and R. Walters, "New goala for the U.S. human genome project: 1998–2003," *Science*, 282: 682–689, October 1998.
2. J. Hammer, H. Garcia-Molina, K. Ireland, Y. Papakonstantinou, J. Ullman, and J. Widom, "Integrating and accessing heterogeneous information sources in TSIMMIS," in *Proceedings of the AAAI Symposium on Information Gathering*, Stanford, CA, March 1995, pp. 61–64.

3. A. Y. Levy, A. Rajarman, and J. J. Ordill, "Querying heterogeneous information sources using source descriptions," *Proc. of 22nd International Conference on Very Large Databases, VLDB96,* 251–262, 1996.
4. A. Bouguettaya, B. Benatallah, L. Hendra, and M. Ouzzani, "World wide database—integrating the web, CORBA and databases," in *Proceedings of the ACM SIGMOD International Conference on Management of Data*, Philadelphia, 1999, pp. 594–596.
5. L. M. Haas, R. J. Miller, B. Niswonger, M. T. Roth, P. M. Schwarz, and E. L. Wimmers, "Transforming heterogeneous data with database middleware: Beyond integration," *IEEE Data Eng. Bull.*, 22(1): 31–36, June 1999.
6. S. B. Davidson, J. Crabtree, B. Brunk, J. Schug, V. Tannen, C. Overton, and C. Stoeckert, "K2/Kleisli and GUS: Experiments in integrated access to genomic data sources," *IBM Sys. J.*, 40(2): 512–531, 2001.
7. T. Etzold and P. Argos, "SRS: An indexing and retrieval tool for flat file data libraries," *Comput. Appl. Biosci.*, 9: 49–57, 1993.
8. Available: http://www.ncbi.nlm.nih.gov/Entrez.
9. Available: http://www.ncbi.nlm.nih.gov/.
10. Available: http://www.rcsb.org/pdb.
11. Y. Arens, C. Y. Chee, C. N. Hsu, and C. A. Knoblock, "Retrieving and integrating data from multiple information sources," *Int. J. Intell. Coop. Inform. Sys.*, 2(2): 127–158, 1993.
12. N. W. Paton, R. Stevens, P. G. Baker, C. A. Goble, S. Bechhofer, and A. Brass, "Query processing in the TAMBIS bioinformatics source integration System," in *Proceedings of the Eleventh International Conference on Scientific and Statistical Databases (SSDBM)*, IEEE Press, New York, 1999, 138–147.
13. S. B. Davidson, C. Overton, V. Tanen, and L. Wong, "BioKleisli: A digital library for biomedical researchers," *J. Dig. Libr.*, 1(1): 36–53, November 1997.
14. L. M. Haas, J. E. Rice, P. M. Schwarz, W. C. Swops, P. Kodali, and E. Kotlar, "DiscoveryLink: A system for integrated access to life sciences data sources," *IBM Sys. J.*, 40(2): 489–511, 2001.
15. Z. Ben Miled, O. Bukhres, Y. Wang, N. Li, M. Baumgartner, and B. Sipes, "Biological and chemical information integration system," in *Proceedings of Network Tools and Applications in Biology*, Genoa, Italy, May 2001, http://www.nettab.org/index.html.
16. Available: http://www.expasy.ch/sprot/.
17. Available: http://www.ncbi.nlm.nih.gov/entrez/Omim/.
18. G. Wiederhold, "Mediators in the architecture of future information systems," *IEEE Comp.*, March 1992, pp. 38–49.
19. J. L. Ambite, N. Ashish, G. Barish, C. A. Knoblock, S. Minton, P. J. Modi, I. Muslea, A. Philpot, and S. Tejada, "ARIADNE: A system for constructing mediators for internet sources," in *Proceedings of the ACM SIGMOD International Conference on Management of Data*, 1998, pp. 561–563.
20. A. Y. Levy, "The information manifold approach to data integration," *IEEE Intell. Sys.*, 1312–1316, 1998.
21. Z. Ben Miled, Y. Wang, N. Li, O. Bukhres, J. Martin, A. Nayar, and R. Oppelt, "BAO, a biological and chemical ontology for information integration" *Online J. Bioinform.*, 1: 60–73, 2002.
22. Available: http://www.ncbi.nlm.nih.gov/Genbank/.

23. Available: http://www.chem.qmul.ac.uk/iubmb/enzyme/.
24. D. S. Weld, "Recent advances in AI planning," *AI Mag.*, 20(2): 93–123, 1999.
25. R. E. Fikes and N. J. Nilsson, "STRIPS: A new approach to the application of theorem proving in problem solving," *Artificial Intell.*, 2: 189–208, 1971.
26. A. Blum and M. L. Furst, "Fast planning through planning graph analysis," *Artificial Intell.*, 90(1–2): 281–300, 1997.
27. Available: http://www.gcg.com/.
28. Available: http://netgenics.com/.
29. Available: http://www.incyte.com/.
30. W. Fujibuchi, S. Goto, H. Migimatsu, I. Uchiyama, A. Ogiwara, Y. Akiyama, and M. Kanehisa, "DBGET/LinkDB: An integrated database retrieval system," *Pacific Symp. Biocomput.*, 3: 683–694, 1997.
31. M. Rebhan, V. Chalifa-Caspi, J. Prilusky, and D. Lancet, "GeneCards: Encyclopedia for genes, proteins, and diseases," Technical report, Bioinformatics Unit and Genome Center, Weizmann Institute of Science, Rehovot, Israel, 1997.
32. I. A. Chen, A. S. Kosky, V. M. Markowitz, and E. Szeto, "Constructing and maintaining scientific database views in the framework of the object protocol model," in *Proc. SSDBM*, IEEE Press, New York, 1997, pp. 237–248.
33. C. T. Yu and W. Meng, *Principles of Database Query Processing for Advanced Applications*, Morgan Kaufmann, San Francisco, 1998.
34. B. A. Eckman, Z. Lacroix, and L. Raschid, "Optimized seamless integration of biomolecular data," in *IEEE International Symposium on Bio-Informatics and Biomedical Engineering (BIBE 2001)*, 2001.
35. M. T. Roth and P. M. Schwartz, "Don't scrap it, wrap it! A wrapper architecture for legacy data sources," in *Proceedings of the Twenty-Third VLDB Conference*, 1997, pp. 266–275.
36. G. D. Schuler, J. A. Epstein, H. Ohkawa, and J. A. Kans, "Entrez: Moledular biology database and retrieval system," *Methods Enzymol.*, 266: 141–162, 1996.
37. S. Walsh, M. Anderson, and S. W. Cartinhour, "ACEDB: A database for genome information," *Methods Biochem. Anal.*, 39: 299–318, 1998.
38. S. F. Altschul, T. L. Madden, A. A. Schaffer, J. Zhang, Z. Zhang, W. Miller, and D. J. Lipman, "Gaped BLAST and PSI-BLAST: A new generation of protein database search programs," *Nucleic Acids Res.*, 25(17): 3389–3402, September 1997.
39. M. T. Roth, F. Ozcan, and L. M. Haas, "Cost models DO matter: Providing cost information for diverse data sources in a federated system," *VLDB*, 599–610, 1999.
40. L. Haas, D. Kossman, E. Wimmers, and J. Yang, "Optimizing queries across diverse data sources," in *Proceedings of the 23rd International Conference on Very Large Databases*, 1997, pp. 276–285.

CHAPTER 9

Computational Analysis of Proteins

DIMITRIOS I. FOTIADIS, YORGOS GOLETSIS, CHRISTOS LAMPROS, and COSTAS PAPALOUKAS

9.1. INTRODUCTION: DEFINITIONS

Proteins are biological macromolecules, which are characterized by an enormous variety of biological *functionality*. For example, some proteins have important enzymatic action accompanied by a certain specialization, while others are responsible for creating various types of receptors for neurotransmitters, hormones, and other substances. In many cases, they form a variety of ionic channels, transport and store vital substances, and constitute the basic cell structural elements.

A protein is made up of a long series of amino acids linked together into chains of varying length. In the human body there exist approximately 30,000 to 40,000 different proteins. These amino acid sequences are folded up to take on the three-dimensional structure of the protein. Determining the three-dimensional shape of the protein is crucial since it determines its function. It is well known that the biological action or the functionality of proteins depends mainly on their *configuration* (structure) in space. This structure is defined by the linear sequence of amino acids that constitute the protein and the formation they fold in space. The three-dimensional structure of proteins can be revealed using X-ray crystallography [1] and nuclear magnetic resonance (NMR) spectroscopy [2]. However, these methods are laborious, time consuming, and expensive. Consequently, there is a need to retrieve information concerning protein folding using other, more convenient methods.

In the last two decades, efforts have been devoted to discover the *primary structure* (amino acid sequence) for a large number of proteins [3]. The experimental results have shown that all the necessary information concerning protein three-dimensional configuration is coded in its linear amino acid sequence [4]. Bearing this in mind, a protein's structure could be predicted using only the information given by the primary sequence [5], but with a limited success so far. Recently, the projects worldwide dealing with the determination of the complete sequence

Genomics and Proteomics Engineering in Medicine and Biology. Edited by Metin Akay

of DNA for a number of organisms (humans as well) have resulted in an increasingly accumulated information for thousands of protein sequences [6]. Most of these proteins are determined by recognizing gene open reading frames (ORFs) and are accompanied without any knowledge about their function. Tools for the development of new and the improvement of older protein structure prediction algorithms and methods can be designed using computer-based approaches.

Structure prediction from primary sequence remains still an unsolved problem. The correlation between amino acid sequence and protein structure has not yet been understood completely. However, new knowledge in the form of relation rules is being continuously discovered. It should be mentioned that the number of distinctive protein-folding patterns is finite [7, 8] and smaller than expected when considering the large number of possible combinations with which the amino acid residues can compose polypeptide chains. Based on the number of experimentally defined distinctive structures [9–11] and recent analyses of entire chromosomes and genomes, the total number of different foldings is estimated at around 30,000 to 40,000.

Since the experimental data confirm that the *primary structure* encompasses all the essential information about *protein folding* [4], several efforts have been made for predicting the three-dimensional structure and consequently the protein's functionality using only the amino acid sequence. Moreover, proteins with low *homology* (similarity) in the primary structure, even less than $\frac{1}{3}$, have shown remarkable resemblance in the overall structure and function. In other words, the protein three-dimensional structure is retained unlike the primary one [7, 12, 13]. On the other hand, very similar amino acid sequences can compose proteins with different biological functions. Finally, it should be mentioned that in many cases proteins appear to have local (i.e., in parts) structural or even functional similarities, which are called *domains*. Consequently, function prediction of proteins from the primary structure is possible but highly complicated.

The employment of the protein's *secondary structure* can assist significantly in the protein-folding problem. Secondary structure incorporates information concerning the protein's geometric regulation while specific combinations in the secondary sequence define certain functions. Therefore, reliable prediction of secondary structure can successfully determine such patterns liable for specific biological roles of the protein.

Another important aspect in the computational analysis of proteins is the gap between the large number of identified primary structures and that of the defined proteins' three-dimensional structures. To address this problem, *homology modeling* (or comparative modeling) has been proposed, which is a process applied in a protein of unknown structure when a homologous protein of identified structure exists. According to this process, the structure of the unknown protein is modeled using the known structure of the homologous one. A drawback of this method is the side chains. Homology refers primarily to the main chain of the protein; thus the adjustment of the side chains in the overall modeling demands the calculation of energy equations as well. Another approach for the same task is *threading*, which is used when high homology cannot be detected. Threading or remote homology modeling can reveal distant similarities, but its success depends highly on the reliable sequence alignment [14].

TABLE 9.1 Secondary-Structure Formations According to DSSP Method

DSSP	Type of Formation	Equivalent Symbol
H	α-helix	H
G	3_{10}-helix	H
I	Π-helix	H
E	Extended (β-strand)	E
B	Residue in isolated β-bridge	E
T	Turn	L
S	Bend	L
No designation	Other	L

Thus, more simplified and robust methods are preferred for protein structure prediction. Those methods employ evolutionary information and are applied mostly for *secondary-structure estimation*, while some efforts have been made to determine interactions among the residues in the primary structure. The central idea behind secondary-structure prediction is that segments of successive residues tend to be in certain secondary-structure formations [15–19]. Hence, secondary-structure estimation is reduced to a problem of pattern recognition, where various algorithms can be used. To simplify even more this task, most commonly only three formations are determined: (a) α-helix (denoted with H), (b) β-sheet (E), and (c) coil or loop (L). A number of algorithms have been developed for this purpose [20], with the most effective being those that combine different approaches [21]. Alternatively, the DSSP (definition of secondary structure of proteins) method [22] can be adopted, which defines eight different secondary-structure formations related with the three former ones, as shown in Table 9.1. Here, the B sequence is equivalent to the EE one and the B-B to the LLL.

So far we have presented some of the concepts and definitions for the computational analysis of proteins. Based on them we will proceed presenting other core subjects, such as protein databases, which are discussed in the next section. Subsequently, protein sequence motifs and domains will be presented followed by a detailed description of the problem of protein sequence alignment, pairwise and multiple. All these are considered as prerequisites to study protein modeling, protein classification, and structure prediction. A very promising new field for the computational analysis of proteins is natural language processing and is described next. We conclude by presenting future trends in this specific research area.

9.2. DATABASES

Any kind of protein information or data that come from experimental research or computational analysis are stored in publicly available databases. Most of these databases are accompanied by software tools for data processing and analysis. Each protein database retains a certain type of data and is associated with other

similar databases. In the following the major protein databases are presented. These can be divided in two main categories: (a) protein sequence databases and (b) protein structure databases.

Sequence databases contain a set of protein sequences and usually include additional comments. This is the case with Swiss-Prot [3, 23] (http://www.expasy.org/sprot/). It is maintained by the Swiss Institute of Bioinformatics in collaboration with the European Bioinformatics Institute (EBI). It contains more than 1.5×10^5 protein records with bibliographic references, secondary-structure information, comments concerning the biological function, if it is known, links to other databases relevant to each registry, and other useful information.

The Protein Information Resource (PIR, http://www.nbrf.georgetown.edu/) databank [24, 25] consists of a number of databases all related to proteins, with the PIR–International Protein Sequence Database (PSD) being the core of them. Like Swiss-Prot, PSD is a database of protein sequences accompanied by additional comments. The data in PSD resulted from the collaboration of PIR with the Munich Information Center for Protein Sequences (MIPS) and the Japanese International Protein Information Database (JIPID). It contains approximately 3×10^5 entries and is divided in four sections: PIR1, PIR2, PIR3, and PIR4. PIR1 and PIR2 contain almost 99% of the records and are very similar in terms of classification and annotation. PIR3 contains those records that have not yet been verified and annotated. PIR4 consists of sequences not found in nature or not expressed under physical conditions. Also it contains sequences that have been composed de novo in the laboratory. The entries in PIR4 have been verified and annotated by PIR experts.

Studies concerning the homology of proteins having the same functionality have discovered discrete patterns in the primary structure. These biologically significant patterns characterize the proteins' families and are included in the PROSITE database [26, 27] in the form of regular expressions (available at http://www.expasy.org/prosite). Today, PROSITE consists of patterns and profiles (or signatures) designed to rapidly and reliably support the determination of which known family of proteins (if any) a new sequence belongs to or which domain(s) it contains using the appropriate computational tools. Currently, patterns for more than 1300 families are stored in PROSITE. For each family detailed analysis is also provided concerning the structure and function of the corresponding proteins. The design of PROSITE follows five leading concepts:

(a) It contains as many biologically meaningful patterns and profiles as possible, so that it can be helpful in the determination of protein function.
(b) The patterns or profiles chosen are specific enough so that they will not detect too many unrelated sequences; still they should detect most, if not all, sequences that clearly belong to the set under consideration.
(c) Each of the entries in PROSITE is fully documented and the documentation includes a concise description of the protein family or domain that it is designed to detect as well as a summary of the reasons leading to the development of the pattern or profile.

(d) Each entry is periodically reviewed to ensure that it is still valid.

(e) There is a very close relationship with the Swiss-Prot protein sequence data bank.

Update of PROSITE and the annotations of the relevant Swiss-Prot entries is regular, while software tools based on PROSITE are used to automatically update the feature table lines of Swiss-Prot entries relevant to the presence and extent of specific domains.

Concerning the structure databases, the Protein Data Bank (PDB, http://www.rcsb.org/pdb/) [1, 28] is the major representative and consists of records of experimentally determined three-dimensional structures of biological macromolecules. These records contain structure information expressed by atomic coordinates, primary- and secondary-structure residues, detailed data from X-ray crystallography or/and NMR spectroscopy, and various references. So far, approximately 29,000 protein structures have been registered in the PDB. Before being made available to the public, each submitted structure is evaluated for its accuracy using specialized software. If the data prove to be correct, then the record acquires a characteristic identifier and is appended to the PDB.

More specialized information suitable for the systematic study of proteins is contained in the Class Architecture Topology Homology (CATH, http://www.biochem.ucl.ac.uk/bsm/cath_new/index.html) database [11, 29]. This database consists of a hierarchical classification of independent structural elements (domains) of those protein records deposited in the PDB that have resolution equal or less than 3 Å. Four major levels are used in this classification: class, architecture, topology (fold family), and homologous superfamily. CATH uses mainly automated methods for classification, but in special cases human reasoning is also employed since it provides better results. The proteins consisting of more than one domain are analyzed in their distinct components using automated algorithms for domain recognition. According to this automated process, 53% of the structures have been classified. The remaining structures were analyzed by either assessing the information provided by the above algorithms or arguments in the literature. The classification is based on the elements of the secondary structure and is realized for four categories: (a) mainly alpha (the majority in the protein's secondary structure are α-helices), (b) mainly beta (the majority are extended β-strands), (c) alpha–beta (α/β and $\alpha+\beta$ structures), and (d) various formations of secondary structures. The architecture classification is realized using the general structure of the domains and the orientation of the secondary-structure elements. Their interconnection (e.g., barrels) is not taken into account. In the topology classification the structures are categorized using both the orientation of the elements and their interconnection. Finally, in the level of homologous superfamilies the structural elements are grouped together according to their sequence similarity.

Another structure database is the Structural Classification of Proteins (SCOP, http://scop.mrc-lmb.cam.ac.uk/scop/) databank [30], which provides a detailed and comprehensive description of structural and evolutionary relationships for all proteins of known structure stored in the PDB. To identify these relations and

accomplish the classification, the proteins are processed empirically after thorough study and comparison of the protein structures. Automated methods are used only for the homogenization of the data which are included in the database. The classification levels are the family, the superfamily, the fold, and the class. It should be mentioned that the first two levels describe near and far evolutionary relationships while the third describes geometric ones. The distinction between evolutionary relationships and those that arise from proteins' physics and chemistry is a feature that is unique to this database.

More precisely, at the family level the sequence similarity is $\geq 30\%$, except in cases where the structures and functions are very similar, implying common ancestor but the sequence similarity is $< 30\%$ (e.g., globulins with 15%). At the superfamily level, proteins with low sequence similarity are classified together but their structure and function indicate a probable common ancestor. At the third level, the proteins that fold in the same way in terms of orientation and topological connections are grouped together. Finally, at the last level, four basic structural classes are defined according to secondary-structure elements: (a) all α (the structure is formed from α-helices), (b) all β (the structure consists of β-strands), (c) α/β (α-helices and β-strands alternate in the protein structure), and (d) $\alpha + \beta$ (α-helices and β-strands are found in distinct regions in the structure). SCOP also provides for each structure links to atomic coordinates, images of the structures, interactive viewers, sequence data, data on any conformational changes related to function, and references in the literature.

To optimize the use of the protein databases available worldwide, integrated systems have been designed for effective information retrieval. The Sequence Retrieval System (SRS, http://srs.ebi.ac.uk) is a powerful and practical system for data management. Via a user-friendly graphical interface SRS is able to perform searches in more than 400 databases. A major advantage of SRS is that it can retrieve data from various sources not necessarily having the same format. Entrez (http://www.ncbi.nlm.nih.gov/Entrez/) is a similar system designed for the databases contained in the National Center for Biotechnology Information (NCBI). Entrez is able to search in different types of databases simultaneously, such as databases with nucleotide or protein sequences, biomolecular structures, and genomes and in MEDLINE via the same interface. Also, more complicated searches can be realized for each database separately. However, Entrez is limited compared to SRS since it operates only with the NCBI databases.

9.3. SEQUENCE MOTIFS AND DOMAINS

The majority of the identified amino acid sequences are not accompanied by any information concerning their protein function. One way to understand these sequences and define their function is to relate them with known proteins using annotated databases. The general rule is to exploit the similarity between the sequences of two proteins on the hypothesis that similar sequences yield similar functions. *Pairwise alignment* is a technique developed for aligning a pair of

sequences in order to detect the similarities between them, so it can help in understanding the function of a newly identified sequence when the latter is compared with a sequence with already known function. However, in some cases the sequence of an unknown (not characterized) protein is too distantly related to all proteins of known structure to detect its resemblance by pairwise sequence alignment.

On the other hand, relationships can be revealed by the occurrence in the protein's sequence of a particular cluster of residue types, which is known as *pattern*, *motif*, *signature*, or *fingerprint*. These motifs arise because specific regions of a protein which may be important, for example for their binding properties or their enzymatic activity, are conserved in both structure and sequence. Moreover, conserved blocks within groups of related sequences (families) can often highlight features which are responsible for structural similarity between proteins and can therefore be used to predict the three-dimensional structure of a protein. This means that biologically significant patterns in the form of regular expressions can be used against sequences of unknown function [26].

While sequence patterns are very useful, there are a number of protein families as well as functional or structural regions or *domains* which cannot be detected using motifs due to their extreme sequence divergence. Most proteins are large enough to contain more than one of those domains. More specifically, a structural domain refers to a segment of a polypeptide chain that can fold into a three-dimensional structure irrespective of the presence of other chain segments. The separate domains of a given protein may interact extensively or may be joined only by a length of polypeptide chain. A protein with several domains may use these domains for functional interactions with different molecules [31]. There are many more proteins than actual functional domains, due to the existence of the same domain in several proteins. Typical examples of important functional domains which are weakly conserved are the globins and the SH2 and SH3 domains. In such domains only a few sequence positions are well conserved.

Apart from the *structural domains*, there are also the *homologous domains*. Such a domain refers to an extended sequence pattern, generally found by sequence alignment methods indicating a common evolutionary origin among the aligned sequences. A homology domain is generally longer than motifs. The domain may include all the given protein sequence or only a part of the sequence. Some domains are complex and made up of several smaller homology domains which are joined to form a larger one during evolution. A domain that covers an entire sequence is called the homeomorphic domain [24].

The use of techniques based on *profiles* allows the detection of such proteins or domains. A profile is a table of position-specific amino acid weights and gap costs which is usually obtained from a well-conserved region in a multiple sequence alignment. These numbers, which are also referred to as scores, are used to calculate a similarity score for any alignment between a profile and a sequence or parts of a profile and a sequence. The profile is moved along the target sequence to locate the best scoring regions using a dynamic programming algorithm. An alignment with a similarity score higher than or equal to a given cut-off value constitutes a motif occurrence. A distinguishing feature between a pattern and a profile is that the

former is usually confined to a small region albeit with high sequence similarity whereas the latter attempts to characterize a protein family or domain over its entire length.

Motifs in general can be distinguished in two classes: deterministic and probabilistic. A *deterministic motif* encloses grammatical inference properties in order to describe syntactically a conserved region of related sequences using an appropriate scoring function based on matching criteria. The expressive power of deterministic patterns can be extended with the incorporation of special symbols, which allow a certain number of mismatches. On the other hand, a *probabilistic motif* is described by a probabilistic model that assigns a probability to the match between the motif and a sequence. The position weight matrix (PWM) provides a simplified model of probabilistic ungapped motifs representing the relative frequency of each character at each motif position. There are also more complicated probabilistic motifs that allow gaps, insertions, and deletions. The profiles (such as those included in the PROSITE database) are types of probabilistic motifs, while hidden Markov models (HMMs) are another example.

The notion of motif can also have structural context. A *structural motif* refers to a combination of several secondary structural elements produced by the folding of adjacent sections of the polypeptide chain into a specific three-dimensional configuration. An example is the helix–loop–helix motif. Structural motifs are also referred to as super secondary structures and folds.

The same happens with the notion of the profile. A *structural profile* is a scoring matrix representing which amino acids should fit well and which should fit poorly at sequential positions in a known protein structure. Profile columns represent sequential positions in the structure and profile rows represent the 20 amino acids. As with a sequence profile, the structural profile is moved along a target sequence to find the highest possible alignment score by a dynamic programming algorithm. Gaps may be included and receive a penalty. The resulting score provides an indication to whether or not the target protein might adopt such a structure.

In protein sequence analysis *motif identification* is one of the most important problems covering many application areas. Motifs are biologically informative in the sense of efficiently modeling sequences and holding useful information about biological families. Therefore, the proteins belonging to a family can be considered as sequences of motifs separated by an arbitrary number of randomly selected characters which indicate the background information. The last observation is also associated with the problem of multiple alignment of sequences, where motif occurrences represent the alignment regions that can be visualized more easily compared to the background information.

Instead of using motifs for extracting conservative information and identifying structurally and functionally important residues, the notion of motifs can also be used for characterizing biological families and searching for new family members. Motifs may enclose powerful diagnostic features, generate rules to determine whether or not an unknown sequence belongs to a family, and thus define a characteristic function for that family. This leads to the development of diagnostic

features (*fingerprints*) that contain groups of conserved motifs used to characterize the family [32].

There are many computational approaches which address *motif identification* in a set of biological sequences which differ according to the type of motifs discovered. The Sequence Alignment and Modeling (SAM) approach [33], Gibbs sampling [34], Multiple Em for Motif Elicitation (MEME) [35], and probabilistic suffix trees [36] represent probabilistic methods for finding multiple shared motifs within a set of unaligned biological sequences. Among those, MEME is a very well known approach. The MEME algorithm fits a two-component finite-mixture model to a set of sequences using the expectation–maximization (EM) algorithm, where one component describes the motif (ungapped substrings) and the other describes the background (other positions in the sequences). Multiple motifs are discovered by sequentially applying a new mixture model with two components to the sequences remaining after erasing the occurrences of the already identified motifs. At the website of the MEME system (http://meme.sdsc.edu/meme/website) the user can submit a set of sequences to the MEME server and receive a reply with the motif occurrences discovered by the system. Besides that, the motif identification implemented in MEME can be the initial step for *protein classification*.

There are also recent improvements in *motif discovery*, such as the greedy mixture learning method [37]. This method learns a mixture of motif models by incrementally adding motif components to a mixture until reaching some stopping criteria. Starting with one initial component that models the background, at each step a new component is added which corresponds to a candidate motif. The greedy mixture learning method describes the problem through likelihood maximization using the EM algorithm, but it is advantageous in identifying motifs with significant conservation (more distinguishable motifs). It leads also to the development of larger protein fingerprints, as the number of discovered motifs is larger. The greedy EM approach can provide Meta-MEME-like models (MEME motif-based HMMs [38]) with more representative motifs and thus enhance the capability of motif-based HMMs to classify protein sequences in the correct category.

9.4. SEQUENCE ALIGNMENT

The alignment of two sequences, known as *pairwise alignment*, the *multiple alignment* (*MA*) and the search of homologous proteins in a database are considered fundamental tasks with significant importance in the computational analysis of proteins. Their necessity becomes even more demanding due to the continuous increment of the biological databases and their high accessibility. Alignment methods can be used to correlate unknown proteins and extract evolutionary and functional information as well.

The main algorithms used for pairwise sequence alignment are the Needleman–Wunsch [39], the Smith–Waterman [40], the BLAST (Basic Local Alignment Search Tool) [41], and the FastA (Fast-All) [42]. On the other hand, for multiple

sequence alignment a well-known algorithm is CLUSTALW [43]. BLAST and FastA are the most common heuristic alignment algorithms. BLAST can be found at the NCBI website (http://www.ncbi.nlm.nih.gov/BLAST/) and FastA at the EBI website (http://www.ebi.ac.uk/fasta33/). Since they are heuristics, they operate in a straightforward manner and are characterized by a fast application. Also, they find the most likely solution, which is not necessarily the optimal one but one that is very close to optimal (near optimal). They can be used to either compare two protein sequences or search a protein database using a given protein.

Several improvements to the original versions of both methods have been presented. Still, BLAST is a fast algorithm which attempts to match a word (i.e., a sequence of residues) of length W above a predefined threshold T [41], which permits a trade-off between speed and sensitivity. A higher value of T results in faster processing but also in increased probability to lose weak similarities. All the matched words are then extended in both directions in order to produce (local) alignments with a similarity above a second threshold S.

FastA, on the other hand, searches for small optimal local alignments using the notion of words [42]. The sensitivity and speed of the searching procedure are proportionally opposite and depend on the size of the word (*k-tup* variable). The overall process starts by detecting all the segments consisting of multiple words, which are combined in the next step in order to provide the final alignment in the last step utilizing also a proper number of gaps. It should be noted that while BLAST and FastA have slight differences in their underlying algorithms, their results are consistent in most cases.

CLUSTALW implements a sophisticated progressive alignment algorithm in order to gradually multiply align the protein sequences. It is available at the EBI website (http://www.ebi.ac.uk/clustalw/). A variant of CLUSTALW is CLUSTALX [43], which operates in two different modes: (a) the multiple alignment mode and (b) the profile alignment mode. In the first mode, all the sequences are compared against each other and a cladogram, (or dendrogram) is constructed. According to this cladogram, the proteins are grouped together based on their similarity. In the second mode, various types of profiles can be used to guide the alignment procedure. An additional feature of CLUSTAL family algorithms is that they can be adjusted easily to generate phylogenetic trees.

All the above algorithms employ a set of parameters which need to be determined prior to their application. Considering two protein sequences of length n each, all the possible alignments for them are given as

$$\binom{2n}{n} = \frac{(2n)!}{(n!)^2} \approx \frac{2^{2n}}{\sqrt{2\pi n}} \tag{9.1}$$

This number is very large even for small proteins and therefore *dynamic programming* algorithms have been proposed to enhance alignment approaches of low

computational effort [39, 40]. These algorithms employ a *scoring* formula which counts the amino acid sequence similarity (in the simplest case one is counted for identity and zero for dissimilarity), while the addition of gaps during the alignment process can contribute negatively in the overall score. It should be mentioned that a different penalty must be used when a new gap is added (gap open penalty) and another one when an existing gap is extended (gap extension penalty). More complex formulas include evolutionary and functional relations between the amino acids through the use of substitution matrices.

The choice of the *substitution matrix* is one of the most important aspects in sequence alignment, local or global. In general, all algorithms utilize a scoring function in which a positive or a negative quantity is added for each amino acid correlation based on a 20×20 matrix. Two basic types of matrices exist: the point accepted mutation (PAM) and the blocks substitution matrix (BLOSUM), which describe roughly the evolutionary relation among the 20 amino acids. However, these types have many variants with different values as elements and each variant is specialized in a certain evolutionary relation between the proteins. In other words, a different matrix should be used when the two proteins are homologous and another when they are distantly related.

Tables 9.2 and 9.3 show the two most frequently used substitution matrices [44], which are the BLOSUM62 [45] and the PAM250 [46], respectively. We can see that similar, evolutionary and physicochemically, amino acids correspond to a positive score while the opposite happens for the unrelated ones. Concerning the *gap penalties*, the first gap usually takes a penalty with a larger value (the gap open penalty) than the rest (the gap extension penalty). Consequently, small or zero values for the above values will yield local alignments while large negative values will produce strict and local alignments.

Another important issue in sequence alignment is the assessment of *statistical significance* [47, 48]. This measures the likeliness of an alignment and distinguishes the accidental from the significant ones. Usually it is expressed through the use of the *expected value* (*E*-value) and depends on the substitution matrix and the gap penalties. The *E*-value can also be normalized, especially in cases of local alignment, so that the different lengths of the proteins under study can be taken into account. Furthermore, the *E*-value is more straightforward and comprehensible to biologists. It should be mentioned that FastA estimates the statistical significance based on the employed database. This is generally more accurate than the BLAST approach, which uses a predetermined data set with known family members but is not faster and in cases of small amounts of data is less effective.

In the following frame a typical BLAST output is shown for two sequences where all the previously mentioned alignment parameters are presented:

```
Sequence 1 gi 12585199 Chromobox protein homolog 4 (Polycomb 2 homolog) (Pc2)
(hPc2). Length 558 (1 .. 558)
Sequence 2 gi 17433290 Chromobox protein homolog 7.
       Length 251 (1 .. 251)
```

```
NOTE:The statistics (bitscore and expect value) is calculated based on the size
of nr database

Score = 133 bits (334), Expect=2e-29
Identities = 84/211 (39%), Positives = 119/211 (55%), Gaps = 11/211 (5%)

Query:      1  MELPAVGEHVFAVESIEKKRIRKGRVEYLVKWRGWSPKYNTWEPEENILDPRLLIAFQNR  60
               MEL A+GE VFAVESI KKR+RKG+VEYLVKW+GW PKY+TWEPEE+ILDPRL++A++ +
Sbjct:      1  MELSAIGEQVFAVESIRKKRVRKGKVEYLVKWKGWPPKYSTWEPEEHILDPRLVMAYEEK  60
mutagenized 32                              *
mutagenized 31                             *
Chromo.     11           ************************************************
CBX7        1  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
Query:      61 ERQEQLMGYRKRGPKPKPLVVQ--VPTFARRSNVLTGLQDSSTDNRAKLDLGA-QGKGQG  117
               E +++  GYRKRGPKPK L++Q       R S+   G +         L  G+ +G  +
Sbjct:      61 EERDRASGYRKRGPKPKRLLLQRLYSMDLRSSHKAKGKEKLCFSLTCPLGSGSPEGVVKA 120
Conflict    77                 *
Chromo.     61 *********
CBX7        61 ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
Query:      118 HQYELNSKKHHQYQPHSKEGKPRPPGKSGKYYYQLNSKKHHPYQPDPKMYDLQYQGGHKE 177
                  EL  K            KPR   K    Y +L+ KK  P  P+ + +  + +   +E
Sbjct:      121 GAPELVDKGPLVPTLPFPLRKPRKAHK----YLRLSRKKFPPRGPNLESHSHRRELFLQE  176
CBX7        121 ++++++++++++++++++++++++++  ++++++++++++++++++++++++++++++
Query:      178 APSPTCPDLGAK----SHPPDKWAQGAGAKG 204
                 P+P       +     + PP++ A    A+G
Sbjct:      177 PPAPDVLQAAGEWEPAAQPPEEEADADLAEG  207
CBX7        177 ++++++++++++++++++++++++++++++++
CPU time:     0.03 user secs.     0.01 sys. secs    0.04 total secs.
Lambda     K        H
   0.312    0.132     0.393
Gapped
Lambda     K        H
   0.267   0.0410     0.140

Matrix: BLOSUM62
Gap Penalties: Existence: 11, Extension: 1
Number of Sequences: 1
Number of Hits to DB: 1082
Number of extensions: 658
Number of successful extensions: 2
Number of sequences better than 10.0: 1
Number of HSP's better than 10.0 without gapping: 1
Number of HSP's gapped: 2
Number of HSP's successfully gapped: 1
Number of extra gapped extensions for HSPs above 10.0: 0
Length of query: 558
Length of database: 760,792,870
Length adjustment: 135
Effective length of query: 423
Effective length of database: 760,792,735
Effective search space: 321815326905
Effective search space used: 321815326905
Neighboring words threshold: 9
Window for multiple hits: 0
X1: 16 (7.2 bits)
X2: 129 (49.7 bits)
X3: 129 (49.7 bits)
S1: 42 (21.8 bits)
S2: 79 (35.0 bits)
```

TABLE 9.2 BLOSUM62 Substitution Matrix

	A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	V	*
A	4	−1	−2	−2	0	−1	−1	0	−2	−1	−1	−1	−1	−2	−1	1	0	−3	−2	0	−4
R	−1	5	0	−2	−3	1	0	−2	0	−3	−2	2	−1	−3	−2	−1	−1	−3	−2	−3	−4
N	−2	0	6	1	−3	0	0	0	1	−3	−3	0	−2	−3	−2	1	0	−4	−2	−3	−4
D	−2	−2	1	6	−3	0	2	−1	−1	−3	−4	−1	−3	−3	−1	0	−1	−4	−3	−3	−4
C	0	−3	−3	−3	9	−3	−4	−3	−3	−1	−1	−3	−1	−2	−3	−1	−1	−2	−2	−1	−4
Q	−1	1	0	0	−3	5	2	−2	0	−3	−2	1	0	−3	−1	0	−1	−2	−1	−2	−4
E	−1	0	0	2	−4	2	5	−2	0	−3	−3	1	−2	−3	−1	0	−1	−3	−2	−2	−4
G	0	−2	0	−1	−3	−2	−2	6	−2	−4	−4	−2	−3	−3	−2	0	−2	−2	−3	−3	−4
H	−2	0	1	−1	−3	0	0	−2	8	−3	−3	−1	−2	−1	−2	−1	−2	−2	2	−3	−4
I	−1	−3	−3	−3	−1	−3	−3	−4	−3	4	2	−3	1	0	−3	−2	−1	−3	−1	3	−4
L	−1	−2	−3	−4	−1	−2	−3	−4	−3	2	4	−2	2	0	−3	−2	−1	−2	−1	1	−4
K	−1	2	0	−1	−3	1	1	−2	−1	−3	−2	5	−1	−3	−1	0	−1	−3	−2	−2	−4
M	−1	−1	−2	−3	−1	0	−2	−3	−2	1	2	−1	5	0	−2	−1	−1	−1	−1	1	−4
F	−2	−3	−3	−3	−2	−3	−3	−3	−1	0	0	−3	0	6	−4	−2	−2	1	3	−1	−4
P	−1	−2	−2	−1	−3	−1	−1	−2	−2	−3	−3	−1	−2	−4	7	−1	−1	−4	−3	−2	−4
S	1	−1	1	0	−1	0	0	0	−1	−2	−2	0	−1	−2	−1	4	1	−3	−2	−2	−4
T	0	−1	0	−1	−1	−1	−1	−2	−2	−1	−1	−1	−1	−2	−1	1	5	−2	−2	0	−4
W	−3	−3	−4	−4	−2	−2	−3	−2	−2	−3	−2	−3	−1	1	−4	−3	−2	11	2	−3	−4
Y	−2	−2	−2	−3	−2	−1	−2	−3	2	−1	−1	−2	−1	3	−3	−2	−2	2	7	−1	−4
V	0	−3	−3	−3	−1	−2	−2	−3	−3	3	1	−2	1	−1	−2	−2	0	−3	−1	4	−4
*	−4	−4	−4	−4	−4	−4	−4	−4	−4	−4	−4	−4	−4	−4	−4	−4	−4	−4	−4	−4	1

Note: Asterisk denotes a translation stop.

TABLE 9.3 PAM250 Substitution Matrix

	A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	V	*
A	2	−2	0	0	−2	0	0	1	−1	−1	−2	−1	−1	−4	1	1	1	−6	−3	0	−15
R	−2	6	0	−1	−4	1	−1	−3	2	−2	−3	3	0	−4	0	0	−1	2	−4	−2	−15
N	0	0	2	2	−4	1	1	0	2	−2	−3	1	−2	−4	−1	1	0	−4	−2	−2	−15
D	0	−1	2	4	−5	2	3	1	1	−2	−4	0	−3	−6	−1	0	0	−7	−4	−2	−15
C	−2	−4	−4	−5	12	−5	−5	−3	−3	−2	−6	−5	−5	−4	−3	0	−2	−8	0	−2	−15
Q	0	1	1	2	−5	4	2	−1	3	−2	−2	1	−1	−5	0	−1	−1	−5	−4	−2	−15
E	0	−1	1	3	−5	2	4	0	1	−2	−3	0	−2	−5	−1	0	0	−7	−4	−2	−15
G	1	−3	0	1	−3	−1	0	5	−2	−3	−4	−2	−3	−5	−1	1	0	−7	−5	−1	−15
H	−1	2	2	1	−3	3	1	−2	6	−2	−2	0	−2	−2	0	−1	−1	−3	0	−2	−15
I	−1	−2	−2	−2	−2	−2	−2	−3	−2	5	2	−2	2	1	−2	−1	0	−5	−1	4	−15
L	−2	−3	−3	−4	−6	−2	−3	−4	−2	2	6	−3	4	2	−3	−3	−2	−2	−1	2	−15
K	−1	3	1	0	−5	1	0	−2	0	−2	−3	5	0	−5	−1	0	0	−3	−4	−2	−15
M	−1	0	−2	−3	−5	−1	−2	−3	−2	2	4	0	6	0	−2	−2	−1	−4	−2	2	−15
F	−4	−4	−4	−6	−4	−5	−5	−5	−2	1	2	−5	0	9	−5	−3	−3	0	7	−1	−15
P	1	0	−1	−1	−3	0	−1	−1	0	−2	−3	−1	−2	−5	6	1	0	−6	−5	−1	−15
S	1	0	1	0	0	−1	0	1	−1	−1	−3	0	−2	−3	1	2	1	−2	−3	−1	−15
T	1	−1	0	0	−2	−1	0	0	−1	0	−2	0	−1	−3	0	1	3	−5	−3	0	−15
W	−6	2	−4	−7	−8	−5	−7	−7	−3	−5	−2	−3	−4	0	−6	−2	−5	17	0	−6	−15
Y	−3	−4	−2	−4	0	−4	−4	−5	0	−1	−1	−4	−2	7	−5	−3	−3	0	10	−2	−15
V	0	−2	−2	−2	−2	−2	−2	−1	−2	4	2	−2	2	−1	−1	−1	0	−6	−2	4	−15
*	−15	−15	−15	−15	−15	−15	−15	−15	−15	−15	−15	−15	−15	−15	−15	−15	−15	−15	−15	−15	0

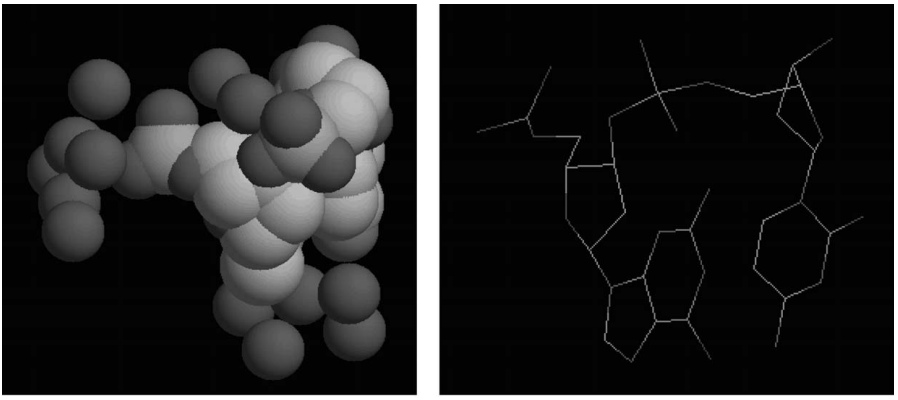
Note: Asterisk denotes a translation stop.

9.5. MODELING

Knowing the three-dimensional structure of proteins is essential for comprehending their physicochemical attributes. However, the larger the complexity of the molecule, the more difficult it is to visualize the three-dimensional formation of its atoms; therefore modeling techniques are utilized. Protein models can assist in the understanding of the molecule's function when its structure has been determined and also in determining the structure itself. They are designed using experimental data (i.e., X-ray or/and NMR) and biological and physicochemical theoretical concepts. Usually their development is iterative, which means that a number of models are constructed and tested successively until they satisfy certain criteria.

There are two basic types of models: (a) space filling and (b) wire frame or skeletal [49, 50]. In the *space-filling models* (Courtauld type) the atoms are represented in shape and size by solid colored units and are interconnected using links (Fig. 9.1*a*). They are useful in studying the overall structure of the protein molecule and its interactions. Their main disadvantage is that they cannot be used in examining internal regions of the protein. In the *wire-frame models* the atomic bonds are represented by lines in terms of length and direction which can rotate around the core of the atoms (Fig. 9.1*b*). These models depict the basic geometry of the molecules and allow the study of all possible configurations. The distances and angles used in the models are determined from the mean value of experimental results taken from a number of micromolecules. This approach often leads to diverged models that need redesigning.

The main drawback of the above models is that their construction is highly time consuming and proportional to the size of the protein molecule. Computer graphics provide tools facilitating the whole process, but still the two-dimensional representation on the computer monitor may conceal important structural information.



(*a*) (*b*)

FIGURE 9.1. (*a*) Space-filling model and (*b*) wire-frame model. The models were created using RASMOL.

For this reason several representation modes have been developed with each depicting certain parts or views of the molecule. Hence, besides the space-filling and wire-frame models, other common models are the ball-and-stick, the ribbons, the surfaces, the animation (which is a dynamic model), and the surface attributes [49, 50]. Several software packages have been developed for molecular graphic representation. A file with the atomic coordinates is the input to these programs while the user can choose in what type of model the molecule can be presented. RASMOL (http://www.umass.edu/microbio/rasmol/), Swiss-PdbViewer (http://www.expasy.ch/spdbv/), and CHIME (http://www.mdlchime.com/chime/) are well-known packages for protein modeling.

9.6. CLASSIFICATION AND PREDICTION

The automated classification of proteins into categories based on their amino acid sequence has been a subject of scientific research for many years. Protein sequences are very difficult to be understood and modeled due to their complex and random-length nature. However, proteins with similar structure/function share a common ancestor and similar amino acid sequence. During evolution the protein sequences will get gradually change mainly in three ways: substitution of one amino acid for another, insertion of an extra amino acid, or deletion of an amino acid. Nevertheless, the protein will still carry out a similar function. Thus, the attempt to group in *families* all the proteins which share a common function is a difficult task. If we compare all the sequences in a family, we can generate probabilities for each amino acid appearing in each position in the sequence. The comparison can be based on an alignment where large chunks of the amino acid sequences align with each other, but still this is a complex task for real protein sequences.

On the other hand, all proteins sharing a similar function should share similar three-dimensional structures and amino acid sequences. These *homolog* proteins have evolutionary relationships and the task is to find such proteins (homology detection). In general, proteins may be classified according to both structural and sequence similarity. For structural classification, the sizes and spatial arrangements of secondary structures are compared with known three-dimensional structures existing in available databases. All these databases facilitate structural comparison and provide a better understanding of structure and function.

The current classifiers for homology detection involve a number of tools for sequence alignment (see Section 9.4). Hidden Markov models have also been adapted to solve the problem of matching distantly related homologies of proteins. An HMM is a statistical model considering all possible combinations of matches, mismatches, and gaps to generate an alignment of a set of sequences. Hidden Markov models use statistical properties of the database to match other protein members. Each HMM (Fig. 9.2) consists of a set of states ST and a set of possible transitions TR between them. Each state stochastically emits a signal, an amino acid in our case; the procedure is then transmitted to some other state with a probability depending on the previous state. The procedure continues until the total of each

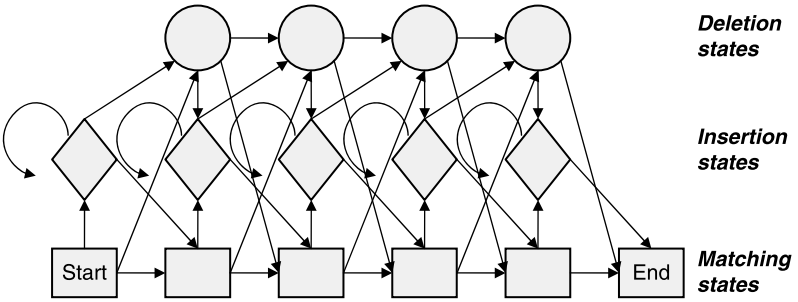


FIGURE 9.2. Schematic diagram of a HMM used for protein analysis. Three types of states are employed: matching, insertion (for gap representation), and deletion (for mismatch representation).

sequence is emitted. There is also a starting state, where the process starts, and a set of transition probabilities from the starting state to each of the possible states. This set of probabilities sums to unity and so does the set of emissions of possible signals in each state and the set of transitions from each state. Normally, a different model is built for each protein family and sequences are then run through the different models. The sequence is then assigned to the model which produces the highest probability. This method has proved very popular and successful and has been shown to perform well in the classification of proteins [51].

Hidden Markov models have been successfully applied in the SAM software system [33]. SAM is used by many organizations for the classification of protein sequences. The SAM HMMs generate sequences with various combinations of matches, mismatches, insertions, and deletions and assign them a probability, depending on the values of the various parameters of the model. It adjusts the parameters so that the model represents as closely as possible the observed variation in a group of related protein sequences. The sequences do not have to be aligned prior to the application of the method. Models are trained with the Baum–Welch algorithm, a type of EM algorithm. SAM can be found online at http://www.cse.ucsc.edu/research/compbio/sam.

Furthermore, the notion of *motifs* has been used in protein classification to reduce the complexity of the HMMs modeling a candidate category in this method. That happens with the adoption of motif-based HMMs in the Meta-MEME system [38], which takes as input the motifs discovered by MEME [35]. The PWMs can be incorporated in the framework of the motif-based HMM and sequences of unknown categorization can be scored against that model. Those HMMs can be either linear or fully connected. Meta-MEME HMMs do not require large training sets and they work properly when the amount of data for training the model is limited. Nevertheless, the reduction in complexity in Meta-MEME is usually accompanied by less accurate results in classification compared to SAM in larger training sets.

In general, pairwise comparison techniques do not perform well as statistical models, but still both models miss certain remote homologies. Moreover, it is

difficult to correctly classify proteins which have major secondary structural similarities when they do not show probable evolutionary origins. This means that the methods cannot distinguish the proteins which belong at the same fold of the SCOP hierarchy, especially when they do not belong to the same superfamily.

Identification of the fold to which a protein with unknown structure belongs is enough to determine the type structure and understand its function. So we can avoid the direct folding approach, which leads to the adoption of thermodynamic optimization for determining the way a polypeptide really folds. Such methods, called *structure-based methods*, identify the structural relationship without directly using any sequence information [52–54]. They create an energy function describing how well a probe sequence matches a target fold. The energy function is often obtained from a database of known protein structures and may, for instance, describe the environment of each residue or the probability to find two residues at a certain distance from each other. Instead, one can relate the unknown protein with proteins of known structure whose fold is already known and classify it to the fold which better satisfies the similarity criteria. This is the indirect approach and here comes the problem of *fold recognition*. The techniques already mentioned for sequence analysis are widely used for fold recognition and are known as *sequence-based methods*.

Among the sequence-based approaches, those that employ HMMs are the most commonly used and demonstrate the higher performance. However, their main drawback is the employment of large model architectures that demand large data sets and high computational effort for training. As a consequence, where these data sets are not available (e.g., small classes or folds), their performance deteriorates. In the following we present a recently introduced HMM that uses a reduced state-space topology to deal with this problem and serves as a classification tool for computational analysis of proteins [55]. It employs a small number of states and adopts a training algorithm with very low complexity and fast application. This HMM simultaneously learns amino acid sequence and secondary structure for proteins of known three-dimensional structure and then is used for two tasks, protein structural class prediction and fold recognition. Secondary-structure information is introduced to the model to increase its performance. The problem here is the multiclass classification of sequences, so the method employed should classify a query sequence of unknown structural category in one of the candidate categories. For class prediction a Bayesian multiclass classification approach was used while for fold recognition a two-stage classifier was adopted. The obtained results are equivalent or even better from other similar approaches (SAM) whereas the computational load is significantly smaller.

More specifically, the number of states of the model is equal to the number of different possible secondary-structure formations according to the DSSP alphabet (H, B, E, G, I, T, and S). It is trained with a low-complexity likelihood maximization algorithm for every candidate class and fold. The test sequences of the classes are assigned to a class according to the Bayesian classification scheme, where different models are trained for each class and the test sequences are assigned to that class whose model gives the maximum probability. The corresponding test sequences

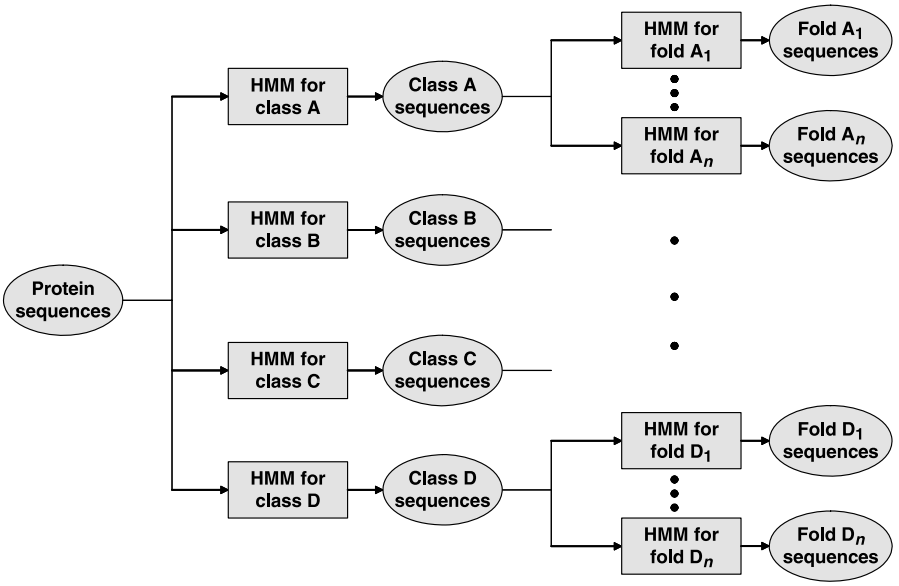


FIGURE 9.3. Two-stage HMM classifier. In the first stage the appropriate class is identified while in the second stage the correct fold is defined.

of the folds are assigned to each fold according to a two-stage classifier. In the first stage, the models of the classes are used to assign the sequences of each fold test set to the appropriate class. Those correctly assigned classes are then assigned to folds using the scores produced by the models of each class fold in the second stage. So the Bayesian classification scheme is used in both stages shown in Figure 9.3.

An alternative approach for fold recognition is the *prediction-based method*, which predicts the secondary sequence of the protein and subsequently uses it to determine the structure. Proteins having a similar fold by definition have very similar secondary structure, meaning that even when amino acid compositions are unrelated, the secondary structure should largely be the same within a fold. Since secondary structure can be predicted with an accuracy of more than 70% [56, 57], several attempts have been made to use this information to improve fold recognition methods. These methods add a positive score to the sequence alignment score if the predicted secondary sequence for a certain residue agrees with the secondary-structure state of the residue.

Besides HMMs another machine learning approach used for protein classification is the evolutionary algorithm. *Genetic algorithms* (GAs) are the best example for this category. The basic idea of a GA is to maintain a population of knowledge structures (called chromosomes) each one representing a candidate solution to the problem. This population evolves over time through competition and controlled variation with the application of genetic operators. In an iterative process (see Fig. 9.4), each chromosome is evaluated according to the quality of the solution

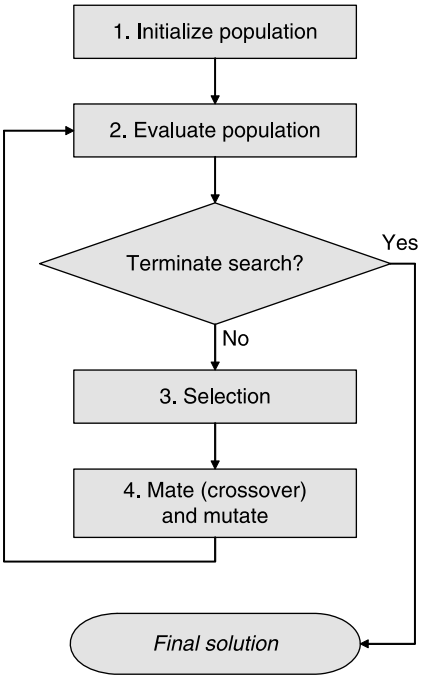


FIGURE 9.4. Typical architecture of a GA.

that it represents (fitness). Chromosomes are selected according to their fitness (by the selection operator) and are combined (mated) in order to produce new chromosomes (by the crossover operator), hopefully combining the "good characteristics" of the parent chromosomes. Some alterations in chromosomes are allowed (by the mutation operator) in order to ensure that all parts of the search space will be reached. The whole process ends if either a certain fitness value is achieved or a maximum number of iterations is reached.

In structure-based fold recognition GAs are very common, as they are used to optimize the energy function which describes how well the query sequence fits with the target fold. Furthermore, GAs are one of the main means of energy minimization in direct protein-folding methods where the best formation of amino acid residues in space must be found. So they are employed in identifying three-dimensional structures of arbitrary polypeptides in arbitrary environments. In that case the most probable formation, that is, that with the minimum energy, gives the tertiary structure of the protein and consequently the fold of the protein. Such an optimization is very complicated as the possible solutions are too many, so only GAs can get through it in a satisfactory way.

The fast messy GA [58] is used for that purpose, which is a particular type of evolutionary algorithm. It is used for the exploitation of domain constraints (such as dihedral angle constraints inspired by the Ramachandran plot, which are

important structural characteristics) and the exploitation of prior secondary-structure analysis. The fast messy GAs have proved to be an interesting and effective computational technique for identifying three-dimensional structures. There are also other GA applications used to minimize an energy function in order to find the lowest energy conformation of a polypeptide based on the description of the environment of each residue, for example using the hydrophobicity of each residue [59].

Genetic algorithms, have also been applied as optimization tools in multiple alignment and subsequent protein classification and prediction [60, 61]. Here chromosomes are sequences or even whole alignments. The whole alignment task begins with a set of initial alignments and iterates through realignment and model assessment. During the iterative process new alignments are constructed by the application of a number of operators, such as crossover or mutation (which for the specific case can be gap insertion, gap deletion, or gap shift operations) (Fig. 9.5).

Feed-forward artificial neural networks (ANNs) can also be used in protein classification and prediction. In general, ANNs are pattern recognition tools widely used in several research fields. They consist of three types of layers, namely the input, hidden, and output layers (Fig. 9.6). In some architectures the hidden layer contains two or more sublayers. All the layers are constituted from a number of processing units, the nodes or neurons. These neurons are interconnected and through a training process the interconnections or weights take specific values adjusted to the classification problem under consideration. According to this scheme, a feedforward ANN has been developed to recognize proteins belonging to the $\alpha\beta$-class [62]. To train and test the neural network, data from the CATH database was employed. Before introducing the sequence data to the ANN for classification, these were encoded and transformed using a simple conversion process and the Fourier transform, respectively. Although this application of

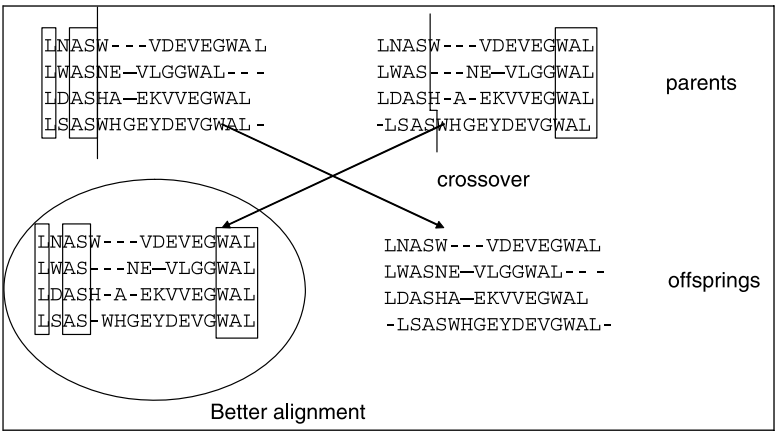


FIGURE 9.5. GA operator (crossover) improving alignment.

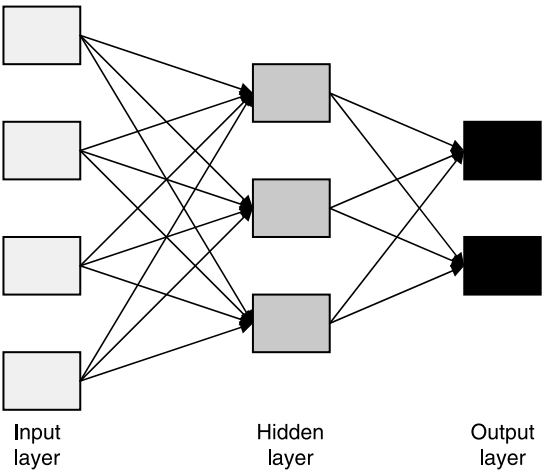


FIGURE 9.6. Typical architecture of feedforward ANN.

ANNs is limited to a certain class only, the results showed that such an approach can be further exploited in protein classification.

A support vector machine (SVM) is another tool that is capable of yielding significant performance in protein classification. In general, SVMs are used for a variety of tasks such as function regression, which alters the parameters of a function to match a curve. They are also very efficient for pattern recognition by producing a decision surface between two sets of data points. The SVM is a special type of ANN that approximately implements structural risk minimization (SRM) as opposed to traditional empirical risk minimization (ERM), used in typical ANNs.

The SVM approaches have been implemented for multiclass protein fold recognition with adequate results [63]. In addition, SVMs have been used in combination with HMMs to detect remote protein homologies, and this approach has been proved to be very successful in finding proteins belonging to the same superfamily [51]. In this method a generative HMM is used as a way of extracting features from the variable-length protein sequences. This HMM represents the superfamily of interest and is trained using sequences from that family. Positive and negative training sequences are then run through the model, and the feature vectors produced, which represent the original protein sequences, can then be modeled in Euclidean space. A SVM is then used to classify the data points into the superfamily of interest.

9.7. NATURAL LANGUAGE PROCESSING

The recent growth in gene sequencing and microarray technologies has led to an explosive development of information and knowledge on genes and their protein products. Although this new knowledge is stored in experiment-specific databases, high-level knowledge about genes, such as function, is still disseminated as written

natural language in journal articles and comment fields of database records [64]. The volume of results makes human-based knowledge extraction almost impossible. On the other hand, natural language processing (NLP) methods can efficiently extract knowledge and make it accessible for computational analysis. As we will see in the following paragraphs, this is an alternative/complementary approach to the techniques described in the previous paragraphs since it can be applied in cases such as homology and molecular interaction identification. First, we describe the major features of a NLP system.

Natural language processing can be defined as the application of computational tools to text for the extraction of knowledge/information. This broad definition leads to the accommodation of several approaches. Therefore, NLP applications can be from highly statistical to highly symbolic. Grammars, rules, and semantics are used for text processing, while often some relationship is examined according to a similarity measure.

Specific NLP components are used in each application, the specific combination of which depends on both the goal and the approach of the specific application. In general, a first step includes identification of the parts of interest (sentences/words), often called tokenization. Specific heuristics identify sentence/word boundaries while sometimes the process is supported by a lexicon look-up. A preprocessing can also be applied, including decapitalizing, removal of special characters, and so on. According to the level of syntactic analysis applied, a part-of-speech tagging can be used where the part of speech (e.g., noun, verb) of each word is estimated. Grammar is used in this step. This parsing often leads to a structured representation of the sentence. Semantics can also be applied in this step. Finally, a mapping mechanism often identifies the similarities of sentences or phrases. Although we tried to describe the NLP process in a "simplistic" way, specific linguistic phenomena and language richness pose a high level of complexity. Specific approaches and heuristics are developed for each step while the use of lexicons and ontologies supports language processing.

Natural language processing has been used in biomedical applications for many years. However, it is only since 1999 that we can identify applications in protein analysis. Although other specific applications may exist, in the following we group the relevant applications in four main categories: synonym, homology, and relation/pathway identification and gene function extraction. A prerequisite step in almost all the approaches is gene or protein identification. In the following paragraphs we briefly present the application of NLP in these approaches:

1. *Protein Identification* The identification of protein/gene names is not a trivial task since there are frequent deviations from recommended nomenclatures or even different naming practices are exercised [64]. The NLP approaches to identify protein names range from simpler dictionary-based approaches to more complicated syntax- and context-based ones. Dictionary-based approaches search from a list of known names. Some can even allow for small variations of gene names; for example, in [65] BLAST was adapted to search a database allowing for approximate textual matches (here sequences are composed of text characters). Since the

volume of new information makes dictionary maintenance a hard task, more advanced methods examine the morphology of words (e.g., [66]) for specific "signals" such as suffixes, prefixes, and so on. For example, words with the suffix-*ase* may be proteins. The use of syntax analysis, such as part-of-speech tagging (i.e., identification of nouns, verbs, etc.) may enhance identification results. Context-based approaches identify words in the context which may signal the existence of proteins in the vicinity of these words [64, 67].

2. *Synonym Identification* The process in biology research aims at producing additional names for the same substance or, vice versa, discovering that different existing names describe the same substance. For example, *lymphocyte associate receptor of death* is synonymous to *LARD*, to *Apo3*, to *DR3*, to *TRAMP*, to *wsl*, and to *TnfRSF12*. This protein may appear with any of these names in the literature, making it hard for biologists/researchers to find relevant knowledge. Synonym identification approaches can be found which, according to the content of strings, identify multiword synonyms [68] or map abbreviations to full texts [69, 70]. Vector space models and the calculation of the cosine as a similarity measure between vectors and rule-based and machine learning techniques have been applied in the above-mentioned approaches. At least one application [71] extends the idea of similarity to contextual similarity; that is, two terms can be judged to be similar/synonymous if they appear in similar (surrounding) contexts. In this way new synonyms can be identified. For example, in [71] they claim to have found a great number of synonyms in their data set not appearing in Swiss-Prot, which when extended to the whole database could lead to the identification of a significant number of novel synonym pairs.

3. *Homology Identification* Homology inferring through sequence similarity can be enhanced with information coming from literature searches. In [72] the PSI-BLAST algorithm, which is a modification of BLAST, was enhanced using literature similarity at each iteration of its database search. A vector space model is applied for the calculation of the similarity between two documents. In this way a *literature-based similarity* between sequences can be obtained. Sequences that lack sufficient literature-based similarity are removed; BLAST is applied among the more "literature-based similar" ones. Supplementing sequence similarity with information from the biomedical literature can increase the accuracy of homology search results.

4. *Relation/Pathway Identification* The automated identification of molecular interactions from the literature could allow biologists to extract knowledge from recently discovered facts and experimental details. Matching of specific prespecified templates (patterns) or rules has been used to extract protein–protein interactions [73, 74]. Since matching a pattern implies a text following a specific pattern, which does not occur often, shallow parsing of phrases [75] allows for identification of certain phrasal components and the extraction of local dependencies between them. Certain approaches (e.g., [73]) recognize noun phrases surrounding verbs of interest (e.g., *activate*, *bind*, *interact*, *regulate*, *encode*, *signal*, and *function*). Figure 9.7 presents the application of such an approach that yields the identification of the interaction between STD1 and TBP proteins.

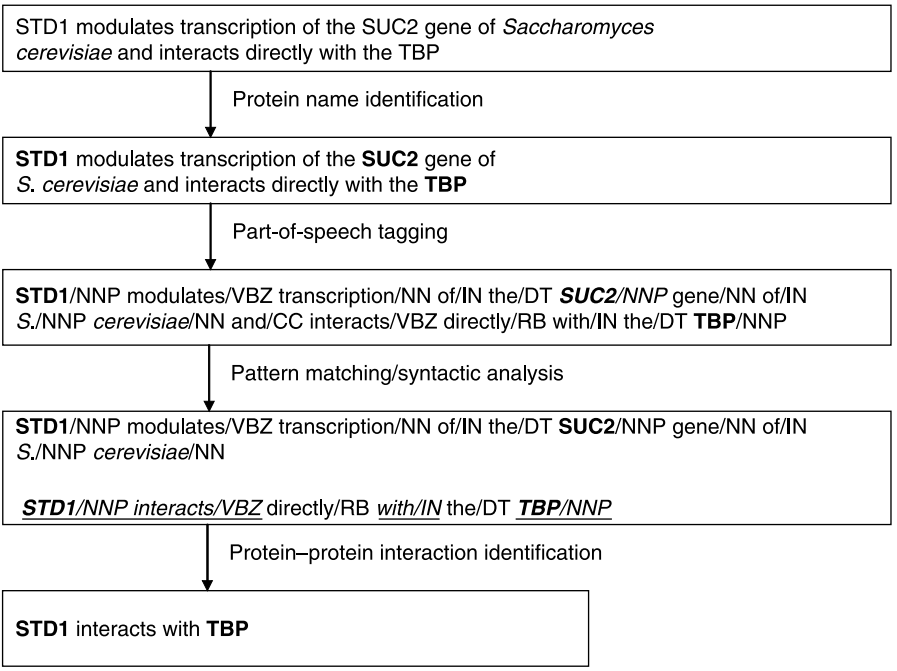


FIGURE 9.7. Protein–protein interaction identification (adapted from [74]).

In [75] partial parsing is performed for noun phrases and then through a discourse analysis identifies coreferring noun phrases. Finally, specific patterns are filled. GENIES [76] can recognize about 125 verbs and subsequently partitions them into 14 broader semantic classes. GENIES employs full parsing and uses syntactic and semantic knowledge. The output of GENIES is a frame-based representation. For example the output for the phrase *mediation of sonic-hedgehog-induced expression of Coup-Tfii by a protein phosphatase* is (From [76])

```
[action,promote,[geneorprotein,phosphatase],
[action,activate, [geneorprotein,sonic hedgehog],
[action,express,X,[geneorprotein],Coup_Tfiii]]]]
```

This extraction of information becomes even harder in such phrases as “an active phorbol ester must therefore, presumably by activation of protein kinase C, cause dissociation of a cytoplasmic complex of NF-kappa B and I kappa B by modifying I kappa B,” where three biological reactions are implied: (i) “an active phorbol ester activates protein kinase C,” (ii) “the active phorbol ester modifies I kappa B,” and (iii) “the active phorbol ester a cytoplasmic complex of NF-kappa B and I kappa B.”

5. *Gene Function Extraction* Natural language processing can also be applied for the extraction of function of gene products from the literature. In [77] an approach similar to sequence alignment is applied, named *sentence alignment*, where a sentence is divided into five segments (Prefix, tag 1, infix, tag 2, and suffix) where tags are gene products or functions. Alignment for prefix, infix, and suffix within texts is used for the validation of truth of the evidence of the specific functions. They also use GeneOntology [78] as a source for synonyms. GeneOntology is also used in [79] for assigning IDs of biological processes to each gene and protein with the use of NLP. Applying dictionary-based name recognition, shallow parsing, and pattern matching, they define actor–object relations, where actors are genes and objects are functions. Then with a keyword-based process they assign the GeneOntology ID to the gene/protein/family.

9.8. FUTURE TRENDS

More and more proteins are identified on a monthly basis leading to an increasing need for understanding their biological role. Forthcoming research should focus on the development of more sophisticated and, of course, more accurate structure prediction and function determination methods. The vast information on proteins deposited in many databases worldwide can be exploited by automated systems which can process it faster and more efficiently than current methods. Besides determining the overall function of each protein, another crucial task is to detect particular patterns in their structure responsible for specialized biological processes. Such patterns can be of great importance in drug discovery and design since additional and more coherent information can be obtained by them. Data-mining techniques can offer much to this direction due to their ability to extract knowledge from a set of data.

REFERENCES

1. J. L. Sussman, D. Ling, J. Jiang, N. O. Manning, J. Prilusky, O. Ritter, and E. E. Abola, "Protein Data Bank (PDB): Database of three-dimensional structural information of biological macromolecules," *Acta Cryst.*, 54: 1078–1084, 1998.
2. R. Kaptein, R. Boelens, R. M. Scheek, and W. F. van Gunsteren, "Protein structures from NMR," *Biochemistry*, 27: 5389–5395, 1988.
3. A. Bairoch and R. Apweiler, "The SWISS-PROT protein sequence databank and its supplement TrEMBL in 1998," *Nucleid Acids Res.*, 26: 38–42, 1998.
4. C. B. Anfinsen, "Principles that govern the folding of protein chains," *Science*, 181: 223–230, 1973.
5. B. Rost and C. Sander, "Bridging the protein sequence-structure gap by structure predictions," *Annu. Rev. Biophys. Biomol. Struct.*, 25: 113–136, 1996.
6. P. Bork, C. Ouzounis, and C. Sander, "From genome sequences to protein function," *Curr. Opin. Struct. Biol.*, 4: 393–403, 1994.

7. C. Chothia and A. M. Lesk, "The relation between the divergence of sequence and structure in proteins," *EMBO J.*, 5: 823–826, 1986.
8. A. V. Finkelstein and B. A. Reva, "Search for the stable state of a short chain in a molecular field," *Prot. Eng.*, 5: 617–624, 1992.
9. L. Holm and C. Sander, "The FSSP database: Fold classification based on structure-structure alignment of proteins," *Nucleic Acids Res.*, 24: 206–210, 1996.
10. T. J. P. Hubbard, A. G. Murzin, S. E. Brenner, and C. Chothia, "SCOP: A structural classification of proteins database,". *Nucleic Acids Res.*, 25: 236–239, 1997.
11. C. A. Orengo, A. D. Michie, D. T. Jones, M. B. Swindells, and J. M. Thornton, "CATH—A hierarchic classification of protein domain structures," *Structure*, 5(8): 1093–1108, 1997.
12. R. F. Doolittle, *Of URFs and ORFs: A Primer On How to Analyze Derived Amino Acid Sequences*, University Science Books, Mill Valley CA, 1986.
13. A. M. Lesk, *Protein Architecture—A Practical Approach*, Oxford University Press, New York, 1991.
14. C. M.-R. Lemer, M. J. Rooman, and S. J. Wodak, "Protein structure prediction by threading methods: Evaluation of current techniques," *Proteins*, 23: 337–355, 1995.
15. A. G. Szent-Gyorgyi and C. Cohen, "Role of proline in polypeptide chain configuration of proteins," *Science*, 126: 697, 1957.
16. E. R. Blout, C. de Loze, S. M. Bloom, and G. D. Fasman, "Dependence of the conformation of synthetic polypeptides on amino acid composition," *J. Am. Chem. Soc.*, 82: 3787–3789, 1960.
17. H. A. Scheraga, "Structural studies of ribonuclease III. A model for the secondary and tertiary structure," *J. Am. Chem. Soc.*, 82: 3847–3852, 1960.
18. P. Y. Chou and G. D. Fasman, "Prediction of the secondary structure of proteins from their amino acid sequence," *Adv. Enzymol.*, 47: 45–148, 1978.
19. J. Garnier, D. J. Osguthorpe, and B. Robson, "Analysis of the accuracy and implications of simple methods for predicting the secondary structure of globular proteins," *J. Mol. Biol.*, 120: 97–120, 1978.
20. W. Kabsch and C. Sander, "How good are predictions of protein secondary structure?" *FEBS Lett.*, 155: 179–182, 1983.
21. G. E. Schulz, C. D. Baryy, J. Friedman, P. Y. Chou, and G. D. Fasman, "Comparison of the predicted and observed secondary structure of T4 phage lysozyme," *Nature*, 250: 140–142, 1974.
22. W. Kabsch and C. Sander, "Dictionary of protein secondary structure: Pattern recognition of hydrogen bonded and geometrical features," *Biopolymers*, 22: 2577–2637, 1983.
23. V. L. Junker, R. Apweiler, and A. Bairoch, "Representation of functional information in the SWISS-PROT data bank," *Bioinformatics*, 15: 1066–1077, 1999.
24. W. C. Barker, F. Pfeffer, and D. C. George, "Superfamily classification in PIR international protein sequence database," *Methods Enzymol.*, 266: 59–71, 1996.
25. C. H. Wu, H. Huang, L. Arminski, J. Castro-Alvear, Y. Chen Y, Z. Z. Hu, R. S. Ledley, K. C. Lewis, H. W. Mewes, B. C. Orcutt, B. E. Suzek, A. Tsugita, C. R. Vinayaka, L. S. Yeh, J. Zhang, and W. C. Barker, "The Protein Information Resource: An integrated public resource of functional annotation of proteins," *Nucleic Acids Res.*, 30(1): 35–37, 2002.

26. A. Bairoch and P. Bucher, "Prosite: Recent developments," *Nucleic Acids Res.*, 22: 3583–3589, 1994.
27. K. Hofmann, P. Bucher, L. Falquet, and A. Bairoch, "The PROSITE database, its status in 1999," *Nucleic Acids Res.*, 27(1): 215–219, 1999.
28. H. M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T. N. Bhat, H. Weissig, I. N. Shindyalov, and P. E. Bourne, "The Protein Data Bank," *Nucleic Acids Res.*, 28(1): 235–242, 2000.
29. F. M. Pearl, N. Martin, J. E. Bray, D. W. Buchan, A. P. Harrison, D. Lee, G. A. Reeves, A. J. Shepherd, I. Sillitoe, A. E. Todd, J. M. Thornton, and C. A. Orengo, "A rapid classification protocol for the CATH Domain Database to support structural genomics," *Nucleic Acids Res.*, 29(1): 223–227, 2001.
30. A. G. Murzin, S. E. Brenner, T. Hubbard, and C. Chothia, "SCOP: A structural classification of proteins database for the investigation of sequences and structures," *J. Mol. Biol.*, 247: 536–540, 1995.
31. A. S. Siddiqui, U. Dengler, and G. J. Barton, "3Dee: A database of protein structural domains," *Bioinformatics*, 17: 200–201, 2001.
32. T. K. Attwood and M. E. Beck, "Prints—a protein motif fingerprint database," *Protein Eng.*, 7: 841–848, 1994.
33. R. Hughey and A. Krogh, "Hidden Markov models for sequence analysis: Extension and analysis of the basic method," *Comput. Appl. Biosci.*, 12(2): 95–107, 1996.
34. C. Lawrence, S. Altschul, M. Boguski, J. Liu, A. Neuwald, and J. Wootton, "Detecting subtle sequence signals: A Gibbs sampling strategy for multiple alignment," *Science*, 262(5131): 208–214, 1993.
35. T. L. Bailey and C. Elkan, "Fitting a mixture model by expectation maximization to discover motifs in biopolymers," in *Proceedings of the Second International Conference on Intelligent Systems for Molecular Biology*, 1994, AAAI Press, Menlo Park, CA, pp. 28–36.
36. G. Bejerano and G. Yona, "Variations on probabilistic suffix trees: statistical modeling and prediction of protein families," *Bioinformatics*, 17: 23–43, 2001.
37. K. Blekas, D. I. Fotiadis, and A. Likas, "Greedy mixture learning for multiple motif discovery in biological sequences," *Bioinformatics*, 19(5): 607–617, 2003.
38. W. N. Grundy, T. L. Bailey, C. P. Elkan, and M. E. Baker, "Meta-MEME: Motif-based Hidden Markov Models of Protein Families," *Comput. Appl. Biosci.*, 13: 397–406, 1997.
39. S. B. Needleman and C. D. Wunsch, "A general method applicable to the search for similarities in the amino acid sequence of two proteins," *J. Mol. Biol.*, 48: 443–453, 1970.
40. T. F. Smith and M. S. Waterman, "Identification of common molecular subsequences," *J. Mol. Biol.*, 147(1): 195–197, 1981.
41. S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman, "Basic local alignment search tool," *J. Mol. Biol.*, 215(3): 403–410, 1990.
42. W. R. Pearson and D. J. Lipman, "Improved tools for biological sequence comparison," *Proc. Natl. Acad. Sci.*, 85(8): 2444–2448, 1988.
43. J. D. Thompson, D. G. Higgins, and T. J. Gibson, "CLUSTAL W: Improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice," *Nucleic Acids Res.*, 22: 4673–4680, 1994.

44. W. R. Pearson, "Effective protein sequence comparison," *Methods Enzymol.*, 266: 227–258, 1996.
45. S. Henikoff and J. G. Henikoff, "Amino acid substitution matrices from protein blocks," *Proc. Natl. Acad. Sci. USA*, 89: 10915–10919, 1992.
46. M. O. Dayhoff, R. M. Schwartz, and B. C. Orcutt, "A model of evolutionary change in proteins. Matrices for detecting distant relationships," in M. O. Dayhoff (Ed.), *Atlas of Protein Sequence and Structure*, National Biomedical Research Foundation, Washington, DC, 1978, pp. S345–S358.
47. R. Arratia and M. S. Waterman, "A phase transition for the score in matching random sequences allowing deletions," *Ann. Appl. Probab.*, 4: 200–225, 1994.
48. W. R. Pearson and T. C. Wood, "Statistical significance in biological sequence comparison," in D. J. Balding, M. Bishop, and C. Cannings (Eds.), *Handbook of Statistical Genetics*, Wiley, Chichester, 2001, pp. 39–65.
49. H.-D. Höltje, W. Sippl, D. Rognan, and G. Folkers, *Molecular Modeling: Basic Principles and Applications*, Wiley-VCH, Weinheim, 2003.
50. A. Hinchliffe, *Molecular Modelling for Beginners*, Wiley, New York, 2003.
51. S. Payne, "Classification of protein sequences into homogenous families," MSc Thesis, University of Frankfurt, Germany, 2001.
52. J. U. Bowie, R. Luthy, and D. Eisenberg, "A method to identify protein sequence that fold into a known three-dimensional structure," *Science*, 253: 164–170, 1991.
53. D. T. Jones, W. R. Taylor, and J. M. Thornton, "A new approach to protein fold recognition," *Nature*, 358: 86–89, 1992.
54. H. Flockner, F. Domingues, and M. J. Sippl, "Proteins folds from pair interactions: A blind test in fold recognition," *Proteins: Struct. Funct. Genet.*, 1: 129–133, 1997.
55. C. Lampros, C. Papaloukas, Y. Goletsis, and D. I. Fotiadis, "Sequence based protein structure prediction using a reduced state-space Hidden Markov Model," submitted for publication.
56. B. Rost and C. Sander, "Prediction of protein secondary structure at better than 70% accuracy," *J. Mol. Biol.*, 232: 584–599, 1993.
57. B. Rost, "Review: Protein secondary structure prediction continues to rise," *J. Struct. Biol.*, 134: 204–218, 2001.
58. G. B. Fogel and D. W. Corne, *Evolutionary Computation in Bioinformatics*, Morgan Kaufmann, San Francisco, 2002.
59. R. Konig and T. Dandekar, "Improving genetic algorithms for protein folding simulations by systematic crossover," *Biosystems*, 50(1): 17–25, 1999.
60. J. Bino and A. Sali, "Comparative protein structure modelling by iterative alignment, model building and model assessment," *Nucleic Acids Res.*, 31(14): 3982–3992, 2003.
61. C. Notredame, "Using genetic algorithms for pairwise and multiple sequence alignments," in G. B. Fogel and D. W. Corne (Eds.), *Evolutionary Computation in Bioinformatics*, Morgan Kaufmann, San Francisco, 2003.
62. A. J. Shepherd, D. Gorse, and J. M. Thornton, "A novel approach to the recognition of protein architecture from sequence using Fourier analysis and neural networks," *Proteins: Struct., Func., Genet.*, 50: 290–302, 2003.
63. C. Ding and I. Dubchak, "Multi-class protein fold recognition using support vector machines and neural networks," *Bioinformatics*, 17: 349–358, 2001.

64. J. T. Chang, H. Schutze, and R. B. Altman, "GAPSCORE: Finding gene and protein names one word at a time," *Bioinformatics*, 22(2): 216–225, 2003.

65. M. Krauthammer, A. Rzhetsky, P. Morozov, and C. Friedman, "Using BLAST for identifying gene and protein names in journal articles," *Gene*, 259: 245–252, 2000.

66. K. Fukuda, A. Tamura, T. Tsunoda, and T. Takagi, "Toward information extraction: Identifying protein names from biological papers," *Pacific Symp. Biocomput.*, 3: 707–718, 1998.

67. T. Rindflesch, L. Tanabe, J. Weinstein, and L. Hunter, "EDGAR: Extraction of drugs, genes and relations from the biomedical literature," *Pacific Symp. Biocomput.*, 5: 517–528, 2000.

68. W. Hole and S. Srinivasan, "Discovering missed synonyms in a large concept-oriented metathesaurus," *Proc. AMIA Symp.*, 354–358, 2000.

69. M. Yoshida, K. Fukuda, and T. Takagi, "PNAD-CSS: A workbench for constructing a protein name abbreviation dictionary," *Bioinformatics*, 16: 169–175, 2000.

70. H. Yu, C. Friedman, and G. Hripcsak, "Mapping abbreviations to full forms in biomedical articles," *J. Am. Med. Inform. Assoc.*, 9: 262–272, 2002.

71. H. Yu and E. Agichtein, "Extracting synonymous gene and protein terms from biological literature," *Bioinformatics*, 19(1): S340–S349, 2003.

72. J. T. Chang, S. Raychaudhuri, and R. B. Altman, "Including biological literature improves homology search," *Pacific Symp. Biocomput.*, 6: 374–383, 2001.

73. T. Sekimizu, H. Park, and J. Tsujii, "Identifying the interaction between genes and gene products based on frequently seen verbs in medline abstracts," *Genome Inform. Ser.: Proc. Workshop Genome Inform.*, 9: 62–71, 1998.

74. T. Ono, H. Hishigaki, A. Tanigami, and T. Tagaki, "Automated extraction of information on protein-protein interactions from the biological literature," *Bioinformatics*, 17(2): 155–161, 2001.

75. J. Thomas, D. Milward, C. Ouzounis, S. Pulman, and M. Carroll, "Automatic extraction of protein interactions from scientific abstracts," *Proc. Pacific Symp. Biocomput.*, 5: 538–549, 2000.

76. C. Friedman, P. Kra, H. Yu, M. Krauthammer, and A. Rzhetsky, "GENIES: A natural-language processing system for the extraction of molecular pathways from journal articles," *Bioinformatics*, 17: S74–S82, 2001.

77. J.-H. Chiang and H.-C. Yu, "MeKE: Discovering the functions of gene products from biomedical literature via sentence alignment," *Bioinformatics*, 19(11): 1417–1422, 2003.

78. The Gene Ontology Consortium (http://www.genontology.org), "Gene Ontology: Tool for the unification of biology," *Nature Genet.*, 25: 25–29, 2000.

79. A. Koike, Y. Niwa, and T. Tagaki, "Automatic extraction of gene/protein biological functions from biomedical text," *Bioinformatics*, 21(7): 1227–1236, 2005.

CHAPTER 10

Computational Analysis of Interactions Between Tumor and Tumor Suppressor Proteins

E. PIROGOVA, M. AKAY, and I. COSIC

10.1. INTRODUCTION

Cancer cell development is attributed to different mutations and alterations in deoxyribonucleic acid (DNA). DNA is a large molecule structured from chains of repeating units of the sugar deoxyribose and phosphate linked to four different bases, abbreviated A, T, G, and C. DNA carries the genetic information of a cell and consists of thousands of genes. DNA controls all cell biological activities. Each gene serves as a recipe on how to build a protein molecule. Proteins perform important tasks for the cell functions or serve as building blocks. The flow of information from the genes determines the protein composition and thereby the functions of the cell. The DNA is situated in the nucleus, organized into chromosomes (Fig. 10.1). Every cell must contain the genetic information and the DNA is therefore duplicated before a cell divides (*replication*). When proteins are needed, the corresponding genes are transcribed into RNA (*transcription*). The RNA is first processed so that noncoding parts are removed (*processing*) and is then transported out of the nucleus (*transport*). Outside the nucleus, the proteins are built based upon the code in the RNA (*translation*).

A significant role in current cancer research is attributed to bioengineering, which is focused on understanding and interpretation of this disease, in terms of gene identification, protein, and DNA modelling, aiming to better understand and evaluate the biochemical processes in cells/tissues, the grounds of disease, and development of progressive diagnostics and drugs using engineering instrumentation and computer science methodologies.

Normally, cells grow and divide to form new cells, as the body needs them. When cells grow old, they die, and new cells take their place. The cell cycle is an ordered

Genomics and Proteomics Engineering in Medicine and Biology. Edited by Metin Akay

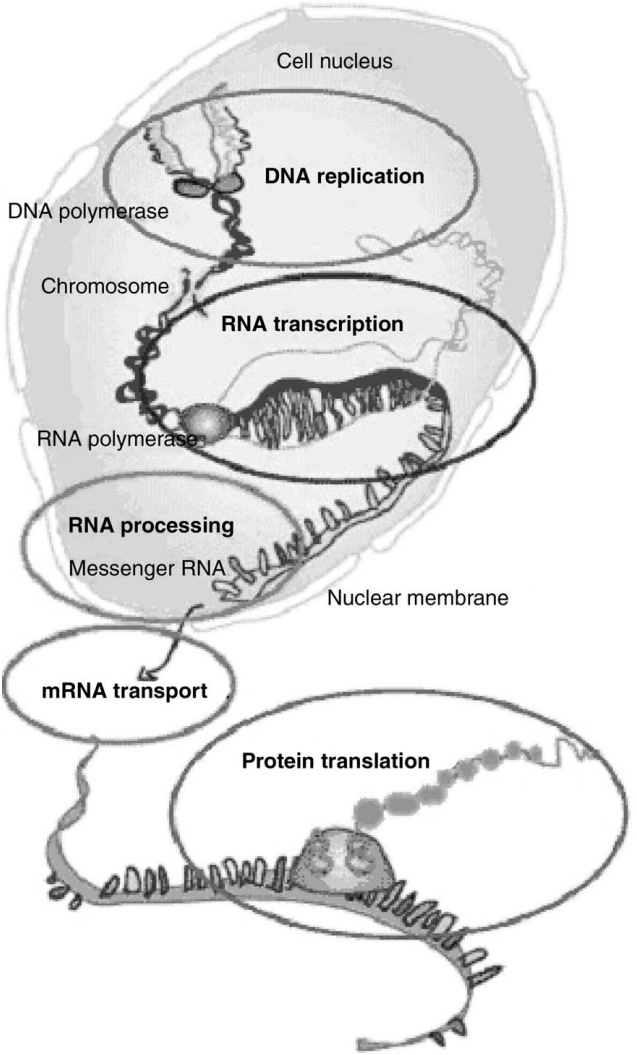


FIGURE 10.1. Cell nucleus.

set of events, culminating in cell growth and division into two daughter cells. Non-dividing cells are not considered to be in the cell cycle. The stages, pictured in Figure 10.2, are G1–S–G2–M. The G1 stage stands for gap 1. The S stage stands for synthesis. This is the stage when DNA replication occurs. The G2 stage stands for gap 2. The M stage stands for mitosis and refers to when nuclear (chromosomes separate) and cytoplasmic (cytokinesis) division occur.

How cell division and thus tissue growth are controlled is a very complex issue in molecular biology. Sometimes this orderly process goes wrong. New cells form

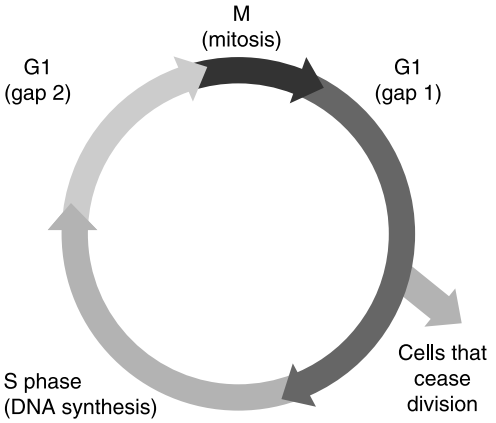


FIGURE 10.2. Stages of the cell cycle.

when the body does not need them, and old cells do not die when they should. These extra cells can form a mass of tissue called a growth or tumor. In cancer abnormal cells divide without control, the regulation of the cell cycle goes awry, and normal cell growth and behavior are lost. As a consequence cancer cells can invade nearby tissues and then spread through the bloodstream and lymphatic system to other parts of the body. Apparently, the cell will become cancerous when the right combination of genes is altered.

However, there are some genes that help to prevent cell malignant behavior and therefore are referred to as tumor suppressor genes. Tumor suppressor genes have been detected in the human genome and are very difficult to isolate and analyse. The *Rb* tumor suppressor gene is located on chromosome 13 of humans. This gene suppresses the development of cancer as its dominant phenotype. Therefore both alleles must be mutant for the disease to develop. The *Rb* gene product interacts with a protein called E2F, the nuclear transcription factor involved in cellular replication functions during the S phase of the cell cycle. When the *Rb* gene product is mutated, a cell division at the S phase does not occur and normal cells become cancerous. Located on human chromosome 17, *p53* is another gene with tumor suppressor activity. This protein contains 393 amino acids and a single amino acid substitution can lead to loss of function of the gene. Mutations at amino acids 175, 248, and 273 can lead to loss of function, and changes at 273 (13%) are the most common [1, 2]. These all act as recessive mutations. Dominant gain-of-function mutations have also been found that lead to uncontrolled cell division. Because these mutations can be expressed in heterozygous conditions, they are often associated with cancers. The genetic function of the *p53* gene is to prevent a division of cells with damaged DNA. Damaged DNA could contain genetic changes that promote uncontrolled cell growth. If the damage is severe, this protein can cause apoptosis, forcing "bad" cells to commit suicide. The *p53* levels are increased in damaged cells. In some way, *p53* seems to evaluate the extent of

damage to DNA, at least for damage by radiation. At low levels of radiation, producing damage that can be repaired, *p53* triggers arrest of the cell cycle until the damage is repaired. At high levels of radiation, producing hopelessly damaged DNA, *p53* triggers apoptosis. If the damage is minor, *p53* halts the cell cycle—and hence cell division—until the damage is repaired. About 50% of human cancers can be associated with a *p53* mutation, including cancers of the bladder, breast, cervix, colon, lung, liver, prostate, and skin. These types of cancers are also more aggressive and have a higher degree of fatalities [1–4].

Other genes, known as proto-oncogenes, can promote cancer if they acquire new properties as a result of mutations, at which point they are called oncogenes. Most common cancers involve both inactivation of specific tumor suppressor genes and activation of certain proto-oncogenes. Proto-oncogene proteins are the products of proto-oncogenes. Normally they do not have oncogenic or transforming properties but are involved in the regulation or differentiation of cell growth. They often have protein kinase (protein phosphotransferase) activity.

There is another interesting group of proteins that play a significant "defending role" in the cell life cycle—*heat shock proteins* (HSPs). They are a group of proteins that are present in all living cells. The HSPs are induced when a cell is influenced by environmental stresses like heat, cold, and oxygen deprivation. Under perfectly normal conditions HSPs act like "chaperones," helping new or distorted proteins fold into shapes essential for their function, shuttling proteins, and transporting old proteins to "garbage disposals" inside the cell. Also HSPs help the immune system recognize diseased cells [5–7]. Twenty years ago HSPs were identified as the key elements responsible for protecting animals from cancer, and studies toward antitumor vaccine development still continue today. Today HSP-based immunotherapy is believed to be one of the most promising areas of developing cancer treatment technology that is characterized by a unique approach to every tumor [5–7].

Recent findings in cancer research have established a connection between a T-antigen—common virus—and a brain tumor in children. The studies suggested the T-antigen, the viral component of a specific virus, called the JC virus, plays a significant role in the development of the most frequent type of malignant brain tumours by blocking the functionality of tumor suppressor proteins such as *p53* and *pRb*. The JC virus (JCV) is a neurotropic polyoma virus infecting greater than 70% of the human population worldwide during early childhood [1–4]. The JC virus possesses an oncogenic potential and induces development of various neuroectodermal origin tumors, including medulloblastomas and glioblastomas. Medulloblastomas are the second most common type of brain tumor in children, making up about 20% of cases. They grow fast and can spread widely throughout the body, and nearly half of all children affected with them die. Radiation exposure and certain genetic diseases are known to put children at risk, but in most cases the cause of medulloblastomas is still a mystery [2].

The most important role in this process is attributed to T-antigen, which has the ability to associate with and functionally inactivate well-studied tumor suppressor proteins *p53* and *pRb*. The immunohistochemical analyses revealed expression of

JCV T-antigen in the nuclei of tumor cells [1–4]. The findings of "in vitro" cancer research indicated that the sirnian virus gene *SV40* induces neoplastic transformation by disabling several key cellular growth regulatory circuits. Among these are the *Rb* and *p53* families of tumor suppressors. The multifunctional large T-antigen blocks the function of both *Rb* and *p53*. Large T-antigen uses multiple mechanisms to block *p53* activity, and this action contributes to tumor genesis, in part, by blocking *p53*-mediated growth suppression and apoptosis. Since the *p53* pathway is inactivated in most human tumors, T-antigen/*p53* interactions offer a possible mechanism by which *SV40* gene contributes to human cancer. Thus, analysis of the gene encoding *p53* and *pRb* proteins could serve to evaluate the effectiveness of a cancer treatment. Mutations in this gene occur in half of all human cancers, and regulation of the protein is defective in a variety of others. Novel strategies that exploit knowledge of the function and regulation of *p53* are being actively investigated [1–4]. Strategies directed at treating tumors that have *p53* mutations include gene therapy, viruses that only replicate in *p53*-deficient cells, and the search for small molecules that reactivate mutant *p53*. Potentiating the function of *p53* in a nongenotoxic way in tumors that express wild-type protein can be achieved by inhibiting the expression and function of viral oncoproteins [1–4].

Therefore an analysis of mutual relationships between viral proteins and two groups of "natural defenders"—tumor suppressors, *p53* and *pRb*, and HSPs—is of great importance in the development of new methodology or drug design for cancer treatment.

10.2. METHODOLOGY: RESONANT RECOGNITION MODEL

Currently a huge amount of scientific effort is directed at solving the problem of finding a cure for cancer. New and advanced drugs and methodologies have been developed and applied with some grade of success; however, the battle with cancer is still continuing. There is an urgent need for theoretical approaches that are capable of analyzing protein and DNA structure–function relationships leading to the design of new drugs efficient to fight many diseases, including cancer.

The resonant recognition model (RRM) [8, 9] essentially presents a nontraditional computational approach designed for protein and DNA structure–function analysis and based on a nontraditional "engineering view" of biomolecules. This methodology significantly differs from other protein analysis approaches, commonly used classical letter-to-letter or block-to-block methods (homology search and sequence alignment), in terms of the view of the physical nature of molecule interactions within the living cells. The RRM concepts present a new perspective on life processes at the molecular level, bringing a number of practical advantages to the fields of molecular biology, biotechnology, medicine, and agriculture.

The physical nature of the biological function of a protein or DNA is based on the ability of the macromolecule to interact selectively with the particular targets (other proteins, DNA regulatory segments, or small molecules). The RRM is a physicomathematical model that interprets protein sequence linear information using

digital signal-processing methods (Fourier and wavelet transform) [8–13]. Initially the original protein primary structure, that is, the amino acid sequence, is transformed into the numerical sequence by assigning to each amino acid in the protein molecule a physical parameter value relevant to the protein's biological activity. Accordingly to RRM main postulates, there is a significant correlation between spectra of the numerical presentation of the protein sequences and their biological activity [8, 9]. A number of amino acid indices (437 have been published up to now) have been found to correlate in some way with the biological activity of the whole protein. Previous investigations [14–19] have shown that the best correlation can be achieved with parameters related to the energy of delocalized free electrons of each amino acid. These findings can be explained by the fact that these electrons have the strongest impact on the electronic distribution of the whole protein. By assigning the electron–ion interaction potential (EIIP) [20] value to each amino acid, the protein sequence can be converted into a numerical sequence. These numerical series can then be analyzed by appropriate digital signal-processing methods (fast Fourier transform is generally used). The EIIP values for 20 amino acids as well as for 5 nucleotides (the whole procedure can be applied to DNA and RNA too) are shown in Table 10.1.

To determine the common frequency components in the spectra for a group of proteins, multiple cross-spectral functions are used. Peaks in such functions denote common frequency components for the sequences analyzed. Through an extensive study, the RRM has reached a fundamental conclusion: *One characteristic frequency characterizes one particular biological function or interaction* [8, 9]. This frequency is related to the biological function provided the following criteria are met:

- One peak only exists for a group of protein sequences sharing the same biological function.
- No significant peak exists for biologically unrelated protein sequences.
- Peak frequencies are different for different biological functions.

It has been found through extensive research that proteins with the same biological function have a common frequency in their numerical spectra. This frequency was found to be a characteristic feature for protein biological function or interaction. The results of our previous work are summarized in Table 10.2, where each functional group of proteins or DNA regulatory sequences is shown with its characteristic frequency and corresponding signal-to-noise ratio (S/N) within the multiple cross-spectral function [9].

It is assumed that the RRM characteristic frequency represents a crucial parameter of the recognition between the interacting biomolecules. In our previous work [8, 9, 14–19] we have shown in a number of examples of different protein families that proteins and their interacting targets (receptors, binding proteins, inhibitors) display the same characteristic frequency in their interactions. However, it is obvious that one protein can participate in more than one biological process, that is, revealing more than one biological function. Although a protein and its target

TABLE 10.1 Electron–Ion Interaction Potential (EIIP) Values for Nucleotides and Amino Acids

Nucleotide	*EIIP* (*Ry*)
A	0.1260
G	0.0806
T	0.1335
C	0.1340
U	0.0289
Amino Acid	*EIIP* (*Ry*)
Leu	0.0000
Ile	0.0000
Asn	0.0036
Gly	0.0050
Val	0.0057
Glu	0.0058
Pro	0.0198
His	0.0242
Lys	0.0371
Ala	0.0373
Tyr	0.0516
Trp	0.0548
Gln	0.0761
Met	0.0823
Ser	0.0829
Cys	0.0829
Thr	0.0941
Phe	0.0946
Arg	0.0959
Asp	0.1263

have different biological functions, they can participate in the same biological process, which is characterized by the same frequency. Therefore, we postulate that the RRM frequency characterizes a particular biological process of interaction between selected biomolecules. Moreover, further research in this direction has led to the conclusion that interacting molecules have the same characteristic frequency but opposite phases at that frequency. Once the characteristic frequency for the particular biological function/interaction is determined, it becomes possible to identify the individual "hot-spot" amino acids that contributed most to this specific characteristic frequency and thus to the observed protein's biological behavior. Furthermore, it is possible then to design bioactive peptides having only the determined characteristic frequency and consequently the desired biological function [18, 19, 21, 22, 23].

Here we present an application of the RRM approach to analysis of the mutual relationships between brain-tumor-associated viral proteins T-antigen and agnoprotein, tumor suppressor proteins *p53* and *pRb*, and HSPs. Also we present the results of the study of possible interactions between melatonin, interleukin-2

TABLE 10.2 Characteristic RRM Frequencies for Protein Groups and DNA Requlatory Sequences

Molecule Type	Frequency	No. Sequence	S/N	Error
DNA Regulatory Sequences				
promoters	0.3437	53	128	0.016
operators	0.0781	8	44	0.008
SOS operators	0.4687	5	13	0.050
enhancers	0.4883	10	467	0.024
Protein Sequences				
oncogenes	0.0313	46	468	0.004
kinases	0.4297	8	71	0.003
fibrinogens	0.4423	5	99	0.001
ACH receptors	0.4922	21	137	0.002
phages' repressors	0.1054	4	51	0.005
bacterial repress.	0.0839	4	56	0.004
repressors	0.0990	25	198	0.008
heat shock proteins	0.0947	10	326	0.005
interferons	0.0820	18	117	0.008
hemoglobins	0.0234	187	119	0.008
signal proteins	0.1406	5	31	0.016
proteases' inh.	0.3555	27	203	0.008
proteases	0.3770	80	511	0.004
trypsins, chym.tr	0.3447	18	257	0.004
chymotrypsin	0.2363	5	35	0.004
serine prot.	0.4609	41	504	0.004
restriction enzymes	0.2910	3	36	0.004
amylases	0.4121	12	170	0.002
neurotoxins	0.0703	16	60	0.004
growth factors	0.2929	105	200	0.016
ins.-like(IGF I,II)	0.4922	12	72	0.008
IGFBP (hum)	0.1602	6	172	0.001
FGFs	0.4512	7	121	0.005
NGFs	0.4040	8	192	0.008
glucagons	0.3203	13	71	0.034
homeo box proteins	0.0459	9	100	0.001
cytochromes B	0.0590	16	201	0.004
cytochromes C	0.4765	45	127	0.004
myoglobins	0.0820	49	128	0.004
lysozymes	0.3281	15	124	0.004
phospholipases	0.0430	29	115	0.004
actins	0.4800	12	163	0.002
myosins	0.3400	11	201	0.004
RNA polymerases	0.3350	10	256	0.001
protein A	0.0342	2	41	0.002

(IL-2), and viral and tumor suppressor proteins that can elucidate why melatonin and IL-2 might play a critical role as supplements in treatment of cancer diseases.

10.3. RESULTS AND DISCUSSIONS

10.3.1. Interactions Between Viral and Tumor Suppressor Proteins

The human neurotropic polyoma virus (JCV), produces a regulatory protein T-antigen, which is a key component in the completion of the viral life cycle. T-antigen has the ability to transform neural cells in vitro and its expression has been detected in several human neural-origin tumors. The JC virus most likely infects humans through the upper respiratory tract and remains in most people throughout their lives and, in some cases, causes minor subclinical problems. However, in people whose immune systems are depressed, either through chemotherapy given to organ transplant recipients or an illness such as AIDS, JCV can become active and may contribute to cancer in the brain [2]. Experimental findings revealed that interactions of viral oncoprotein T-antigen with tumor suppressor proteins could lead to induction of cancer [1–4].

In this study 8 JC viral T-antigen protein sequences, 13 *p53* protein sequences, and 9 *pRb* protein sequences were investigated concerning the understanding of the structure–function relationship within these proteins. A multiple cross-spectral analysis was performed for each selected protein group as well as for their mutual combination using the EIIP values (Figs. 10.3*a* to 10.3*f*). As a result, characteristic frequencies of analyzed protein groups were obtained and are shown in Table 10.3. The RRM analysis was applied to a group of 8 T-antigen proteins, and the common feature in terms of characteristic frequency was identified at $f = 0.2061 \pm 0.125$, $S/N = 129.58$. This frequency component is common to all analyzed sequences and therefore can be considered as the consensus characteristic of their common biological activity for all protein sequences in this functional group.

The *p53* tumor suppressor gene has proven to be one of the genes most often mutated in human cancers. It involves mainly point mutations leading to amino acid substitutions in the central region of the protein and thus causes its abnormal functions. Because *p53* and *pRb* proteins are the key players in defending our body against cancer, it is of great importance to determine the characteristic frequencies of these proteins that correspond to their biological functionality. Thus, the RRM procedure was repeated with *p53* and *pRb* tumor suppressor proteins and their cross-spectral functions obtained are shown in Figures 10.3*b* and 10.3*c*. The prominent characteristic frequencies of *p53* and *pRb* tumor suppressor proteins were identified at $f = 0.4326 \pm 0.077$, $S/N = 159.97$ and at $f = 0.4316 \pm 0.111$, $S/N = 164.28$, respectively. As was mentioned above each specific biological function of the protein is characterized by a single frequency. The similarity of characteristic frequencies of *p53* and *pRb* proteins (Table 10.3) is expected as both *p53* and *pRb* proteins are tumor suppressors sharing the same biological function. Thus, the frequency $f = 0.4326$ identified within the RRM analysis is

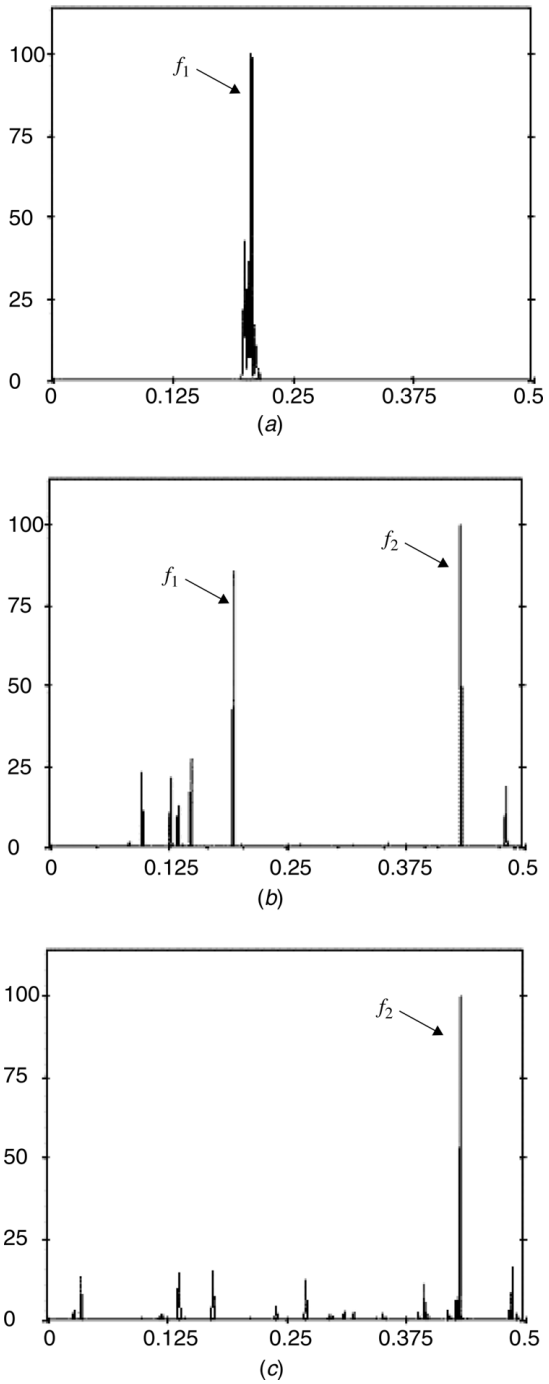
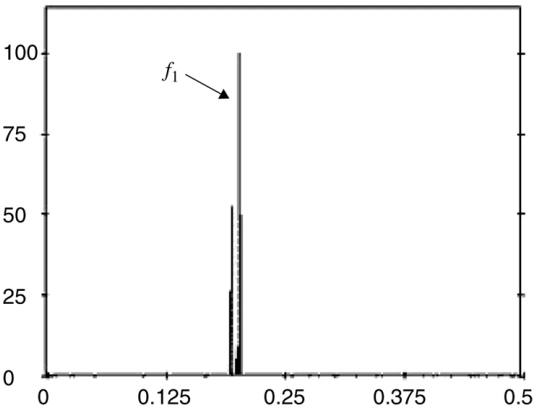
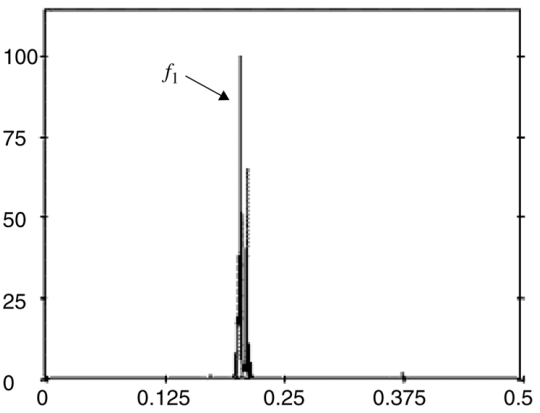


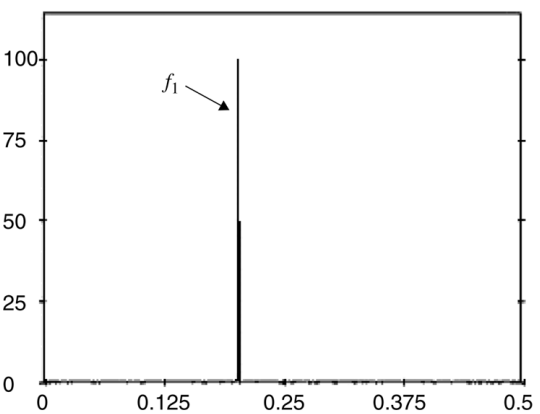
FIGURE 10.3. Multiple cross-spectral functions of protein groups: (*a*) JC viral T-antigen proteins, (*b*) *p53* proteins, (*c*) *pRb* proteins, (*d*) JC viral T-antigen and *p53* proteins, (*e*) JC viral T-antigen and *pRb* proteins, and (*f*) T-antigen, *p53* and *pRb* proteins. The prominent peak(s) denote common frequency components. The abscissa represents RRM frequencies, and the ordinate is the normalised intensity. The prominent peaks were found for T-antigen at $f_1 = 0.2061$ and for *p53* (*pRb*) proteins at $f_2 = 0.4326$.



(*d*)



(*e*)



(*f*)

FIGURE 10.3. *Continued.*

TABLE 10.3 Peak Frequency and Signal-to-Noise Ratio of Protein Groups

Protein Group	Frequency	No. Sequence	S/N	Standard Error, 1/No. Seq.
T-antigen	0.2061	8	129.58	0.125
Agnoprotein	0.3047	9	62.12	0.111
p53	0.4326	13	159.97	0.077
pRb	0.4316	9	164.28	0.111
T-antigen, *p53*	0.2021	21	312.36	0.048
T-antigen, *pRb*	0.2041	17	167.12	0.059
T-antigen, *p53*, *pRb*	0.2021	30	506.28	0.033
T-antigen, Agnoprotein	0.2480	17	130.86	0.059
T-antigen, Agnoprotein, *p53*	0.3564	30	288.49	0.033
T-antigen, Agnoprotein, *pRb*	0.2402	26	157.63	0.038
T-antigen, Agnoprotein, *p53*, *pRb*	0.2021	39	290.27	0.026
Agnoprotein, *p53*	0.3564	22	292.34	0.045
Agnoprotein, *pRb*	0.3096	18	176.23	0.056

considered as a characteristic feature of the specific biological activity of *p53* and *pRb* proteins—ability to stop the formation of tumors. After careful examining of the corresponding consensus spectrums of *p53* and *pRb* proteins (Figs. 10.3*b* and 10.3*c*), we observe more than one less significant peak corresponding to other biological functions determined within the RRM analysis.

It is known that the human polyomavirus (JCV) also contains an open reading frame within the late region of the viral genome that encodes a 71-aminoacid protein, the agnoprotein. Following accumulating evidence in support of an association between JCV infection and human brain tumors, the expression of agnoprotein in a series of 20 well-characterized medulloblastomas was assessed [24]. Importantly, some medulloblastoma samples that expressed agnoprotein had no sign of T-antigen expression. The *p53* protein was detected in only 6 of the 11 tumors in which agnoprotein was expressed. None of the 20 samples showed expression of the viral late capsid proteins, ruling out productive infection of the tumor cells with JCV. These data provide evidence that the JCV late gene encoding the auxiliary agnoprotein is expressed in tumor cells. The finding of agnoprotein expression in the absence of T-antigen expression suggests a potential role for agnoprotein in pathways involved in the development of JCV-associated medulloblastomas [24]. Despite all of this, the role of agnoprotein in the development of brain tumors is still unknown. Recent studies suggest, however, that the interaction of T-antigen with agnoprotein may affect T-antigen ability to control cell growth. The communication between these two viral proteins may impact the ability of the virus to induce brain tumors [24]. The researchers also found agnoprotein and T-antigen in about 50% of the samples, with some samples containing only agnoprotein. They postulated "the finding of agnoprotein expression in the absence of T-antigen expression suggests a potential role for agnoprotein in pathways that control the development of JCV-associated medulloblastomas [24]." Obviously further study is needed to prove that the virus plays a significant role in formation of medulloblastomas [24].

New findings in these studies can be used to develop therapeutic vaccines against T-antigen and agnoprotein. Such vaccines could conceivably prevent the JCV from inducing the formation of medulloblastomas.

Following the aim to explore the interactions between T-antigen and agnoprotein and the possibility of the influence of agnoprotein on the interactions between T-antigen and tumor suppressor proteins *p53* and *pRb*, the RRM analysis was performed. Peak frequency and signal-to-noise values are calculated and shown in Table 10.3. Multiple cross-spectral functions of viral protein T-antigen, agnoprotein, and their mutual interactions with tumor suppressor proteins *p53* and *pRb* are shown in Figures 10.4*a* to 10.4*g*.

10.3.2. Mutual Interactions between IL-2, Melatonin, Oncogene, and Viral and Tumor Suppressor Proteins

The human body normally produces IL-2. This protein is a type of biological response modifier, a substance that can improve the body's natural response to disease, enhances the ability of the immune system to kill tumor cells, and may interfere with blood flow to the tumor. Aldesleukin is IL-2 that is made in the laboratory for use in treating cancer and other diseases. Melatonin has now moved rapidly center stage from an area of pure research interest to one of possible therapeutic importance. Melatonin acts in multiple ways within the organism. Of particular interest is the role of melatonin in cancer biology, its potential either by itself or in combination with other drugs in cancer chemotherapy [25]. Numerous studies

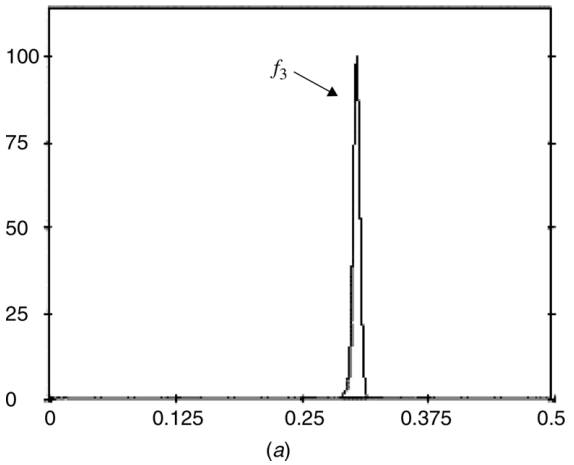
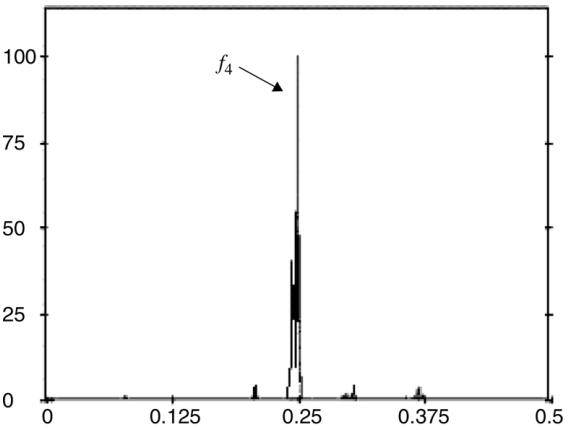
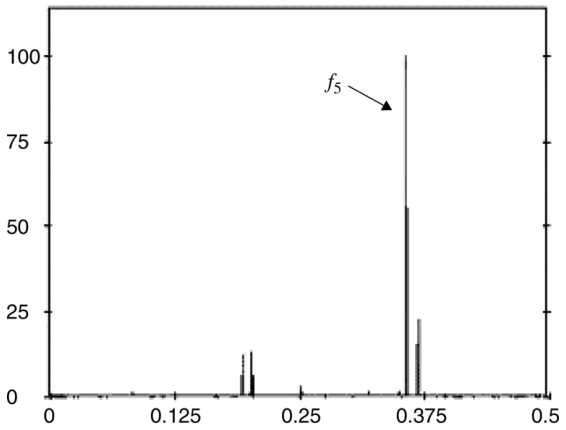


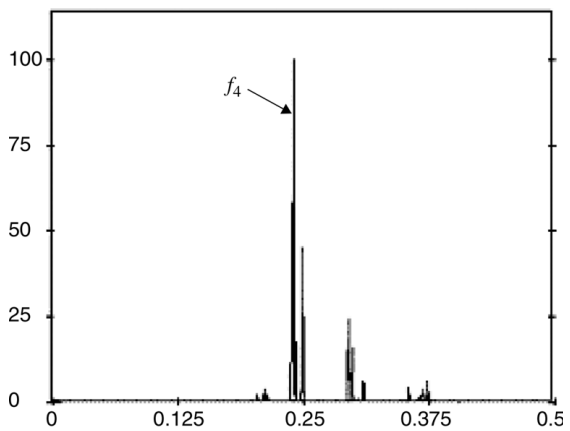
FIGURE 10.4. Multiple cross-spectral functions of protein groups: (*a*) Agnoprotein, (*b*) T-antigen and Agnoprotein, (*c*) T-antigen, Agnoprotein and *p53* proteins, (*d*) T-antigen, Agnoprotein and *pRb* proteins, (*e*) T-antigen, Agnoprotein, *p53* and *pRb* proteins, (*f*) Agnoprotein and *p53* proteins, and (*g*) Agnoprotein and *pRb* proteins. The prominent peaks were identified for Agnoprotein at $f_3 = 0.3047$, for T-antigen and Agnoprotein at $f_4 = 0.2480$ and for Agnoprotein and *p53* at $f_5 = 0.3564$ (Table 10.3).



(*b*)

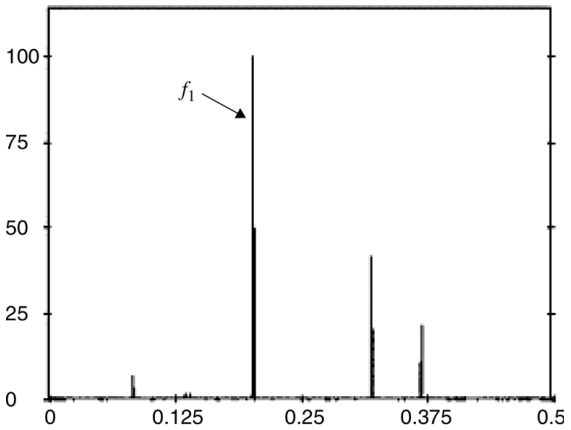


(*c*)

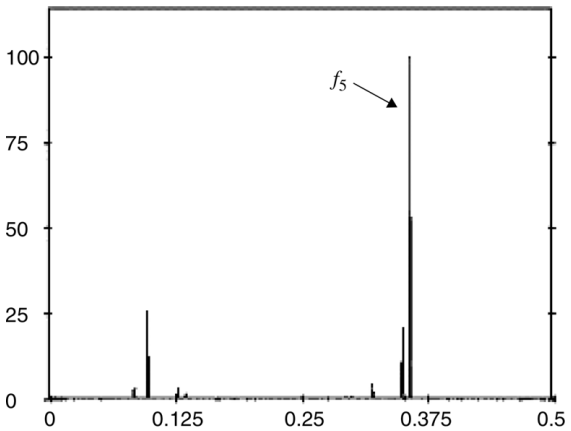


(*d*)

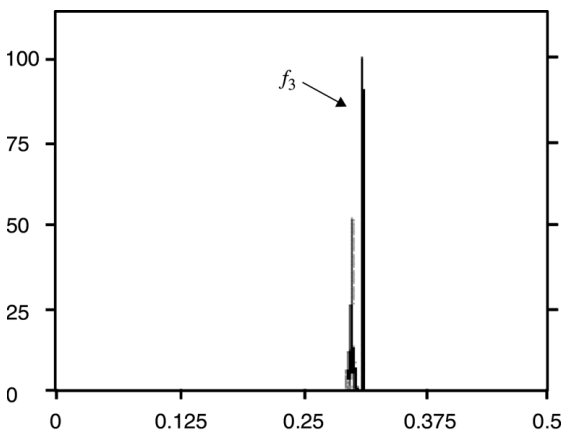
FIGURE 10.4. *Continued.*



(*e*)



(*f*)



(*g*)

FIGURE 10.4. *Continued.*

of melatonin and its effects on cancer have been conducted. Some suggest that melatonin extends survival and improves the quality of life for patients with certain types of untreatable cancers. Melatonin combined with IL-2 has been studied as an anticancer treatment [25–28]. In one study of 80 cancer patients, use of melatonin in conjunction with IL-2 led to more tumor regression and better survival rates than treatment with IL-2 alone. However, it was also reported that the results of 32 clinical studies designed to measure the effects of melatonin on cancer were mixed and inconclusive [28]. A study of melatonin's ability to ease the side effects of chemotherapy drugs found that high doses of the hormone had little effect. It was summarized that the antitumor activity of IL-2 is augmented by melatonin, resulting in a decrease in the number of IL-2 doses needed to exert an anticancer response. Moreover, melatonin may increase the antitumor activity of IL-2 by inhibiting tumor growth factor production. A pilot study was done using low-dose IL-2 plus melatonin in 14 patients with untreatable endocrine tumors. The results suggest that low-dose IL-2 and melatonin may be a well-tolerated therapy for advanced endocrine tumors. Also the results of the study show the objective tumor regression was noted in 3 of the 14 patients (lung, kidney, and liver tumors) [28].

Taking into account the existing documented evidence of the possible influence of melatonin on the biological performance of IL-2, a computational analysis of mutual interactions between melatonin, IL-2, and oncogene proteins using the RRM approach was performed. The values of characteristic frequencies and signal-to noise ratios of each protein group analyzed are shown in Table 10.4. Multiple cross-spectral functions of analyzed protein groups are shown in Figures 10.5*a* to 10.5*g*. In addition, the RRM was used to determine the characteristic frequencies of melatonin protein and its interactions with viral and tumor suppressor proteins. The peak frequencies of these selected proteins are shown in Table 10.5. The resulting cross-spectral functions of the analyzed proteins can be observed from Figures 10.6*a* to 10.6*l*.

It is proposed that the RRM characteristic frequencies present the common feature of the interacting sequences and thus a common interaction. In our previous work it was also proposed that this characteristic frequency could represent the oscillations of a physical field, which are responsible for information transfer between the interacting biomolecules [9]. As a consequence, it is postulated that

TABLE 10.4 Peak Frequency and Signal-to-Noise Ratio of Protein Groups

Protein Group	Frequency	Signal-to-Noise	No. Sequence	Standard Error, 1/No. Seq.
Oncogene	0.0317	408.77	45	0.022
IL-2	0.0303	222.68	23	0.043
Melatonin	0.0205	403.58	28	0.036
IL-2, Melatonin	0.0283	486.17	51	0.020
IL-2, Oncogene	0.0322	435.09	68	0.015
IL-2, Melatonin, Oncogene	0.0303	500.60	96	0.010
Melatonin, Oncogene	0.3379	493.50	73	0.014

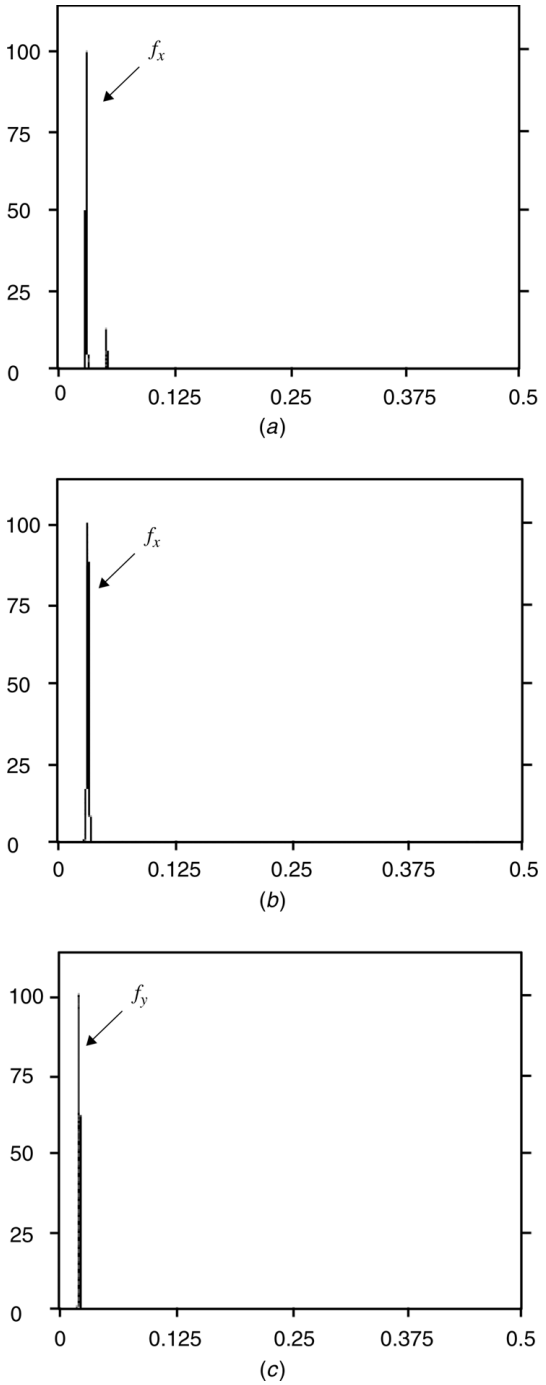
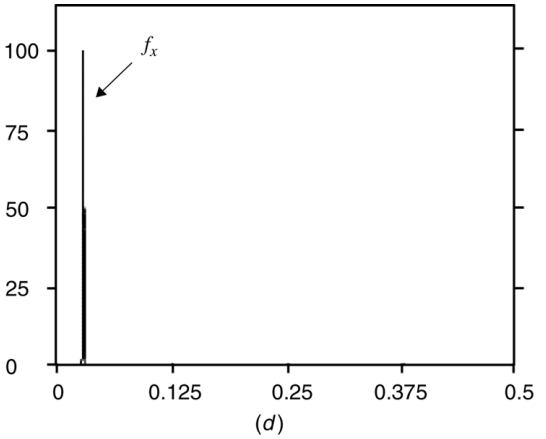
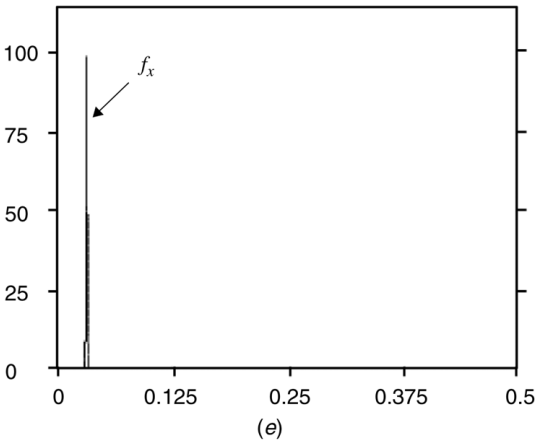


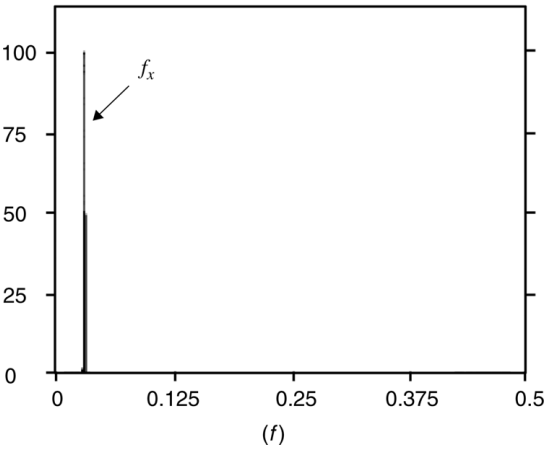
FIGURE 10.5. Multiple cross-spectral functions of protein groups analysed: (*a*) Oncogene, (*b*) Inerleukin-2, (*c*) Melatonin, (*d*) Interleukin-2 and Melatonin, (*e*) Interleukin-2 and Oncogene, (*f*) Interleukin-2, Melatonin and Oncogene, and (*g*) Melatonin and Oncogene proteins. Prominent peaks were identified for Oncogenes at $f_x = 0.0317$, Melatonin at $f_y = 0.0205$, Melatonin and Oncogene at $f_z = 0.3379$.



(*d*)



(*e*)



(*f*)

FIGURE 10.5. *Continued.*

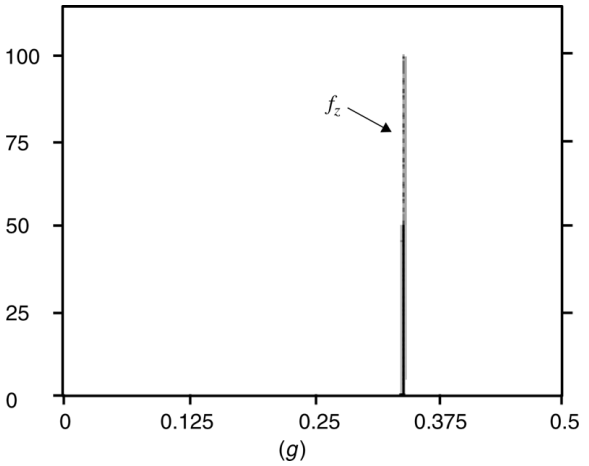


FIGURE 10.5. *Continued.*

RRM characteristic frequency is a relevant parameter for mutual recognition between biomolecules and is significant in describing the interaction between proteins and their substrates or targets. Therefore, it is concluded that the RRM characteristic frequency may dictate the specificity of the protein interactions [9].

From Figure 10.3*d* we can observe only one dominant peak corresponding to the common frequency component for the combined group of T-antigen and *p53* proteins at $f = 0.2021 \pm 0.048$, $S/N = 312.36$. Analogous results were obtained in the analysis of interactions between T-antigen and *pRb* proteins. A single

TABLE 10.5 Peak Frequency and Signal-to-Noise Ratio Values of Protein Groups

Protein Group	Frequency	S/N	No. Sequence	Standard Error, 1/No. Seq.
Melatonin, T-antigen	0.0205	447.57	36	0.028
Melatonin, Agnoprotein	0.3359	274.32	37	0.027
Melatonin, Agnoprotein, T-antigen	0.3359	211.13	45	0.022
Melatonin, *p53*	0.0215	512.76	41	0.024
Melatonin, *pRb*	0.0205	487.00	37	0.027
Melatonin, *p53*, *pRb*	0.0215	512.50	50	0.020
Melatonin, *p53*, T-antigen	0.0215	507.15	49	0.020
Melatonin, *pRb*, T-antigen	0.0205	500.93	45	0.022
Melatonin, *p53*, *pRb*, T-antigen	0.0215	511.70	58	0.017
Melatonin, *p53*, *pRb*, Agnoprotein	0.0215	511.67	59	0.017
Melatonin, *p53*, *pRb*, Agnoprotein, T-antigen	0.0215	498.93	76	0.013

peak corresponding to the protein's biological activity was identified at $f = 0.2041 \pm 0.059$, S/N = 167.12 (Fig. 10.3*e*). These characteristic frequencies are very close to each other (Table 10.3), and they overlap with each other within the calculation error, indicating their mutual recognition. Therefore, we can conclude that this identified frequency can be considered a characteristic feature of the mutual interactions between the analyzed proteins, T-antigen and *p53* and T-antigen and *pRb*, respectively, that might cause cell damage and tumor induction in the brain.

Finally, we also explored the possibility of a mutual three-component interaction between T-antigen, *p53*, and *pRb* proteins by applying the RRM cross-spectral analysis to all three functional groups of proteins. Interestingly, we have found that there is a very prominent frequency component (Fig. 10.3*f*) at $f = 0.2021 \pm 0.033$, S/N = 506.28 (Table 10.3) shared by all analyzed sequences. It should be noted that this frequency is the same (within the calculation error of ± 0.033) as the characteristic frequency of T-antigen and *pRb* proteins (Fig. 10.3*e*) and of T-antigen and *p53* proteins, respectively (Fig. 10.3*d*). Thus, the results obtained indicate the common characteristic frequency for all three interacting proteins—T-antigen, *p53*, and *pRb* proteins—at $f = 0.2021$, which is a characteristic feature of tumor formation. This would confirm the RRM main concept that proteins and their targets recognize/interact with each other based on the same (similar) characteristic frequency. Consequently, we conclude that the

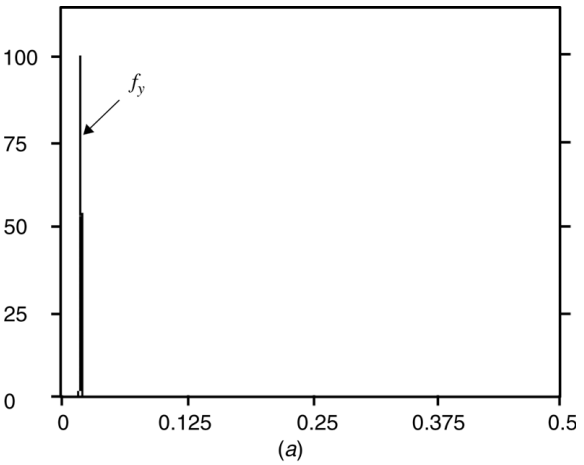
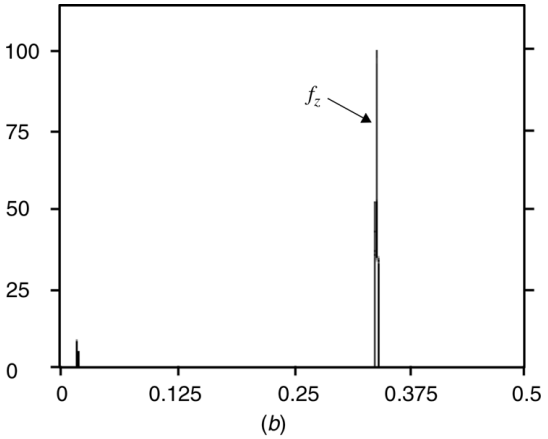
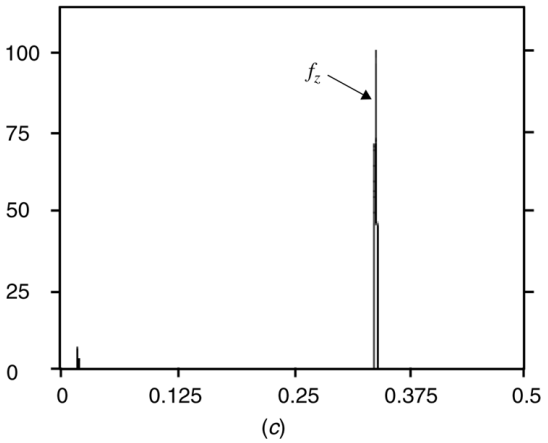


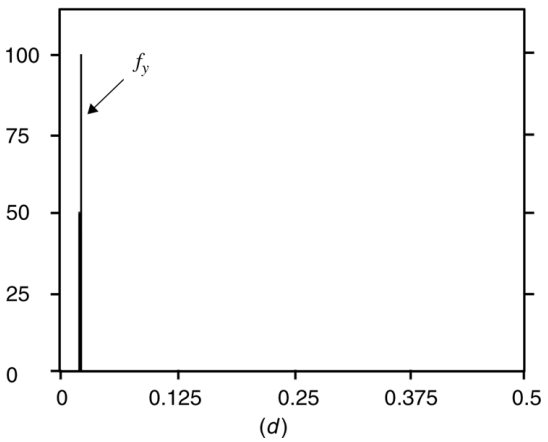
FIGURE 10.6. Multiple cross-spectral function of protein groups: (*a*) Melatonin and T-antigen, (*b*) Melatonin and Agnoprotein, (*c*) Melatonin, T-antigen and Agnoprotein, (*d*) Melatonin and *p53*, (*e*) Melatonin and pRb, (*f*) Melatonin, *p53* and *pRb*, (*g*) Melatonin, *pRb* and T-antigen, (*j*) Melatonin, *p53*, *pRb*, T-antigen, (*k*) Melatonin, *p53*, *pRb*, Agnoprotein, and (*l*) Melatonin, *p53*, *pRb*, Agnoprotein, T-antigen. Prominent peaks were found for Melatonin and T-antigen at $f_y = 0.0205$ and Melatonin, T-antigen, Agnoprotein at $f_z = 0.3359$.



(*b*)

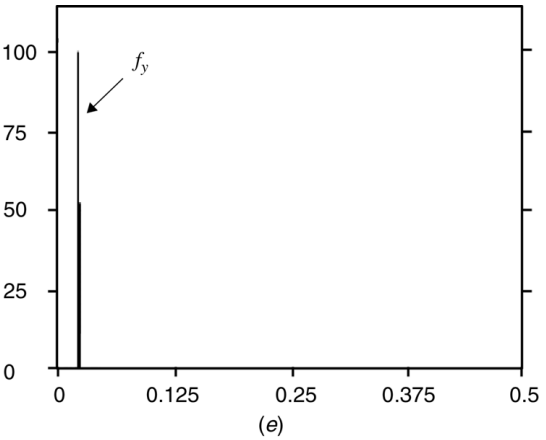


(*c*)

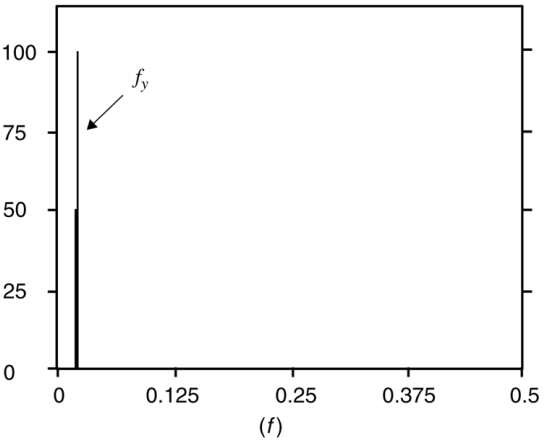


(*d*)

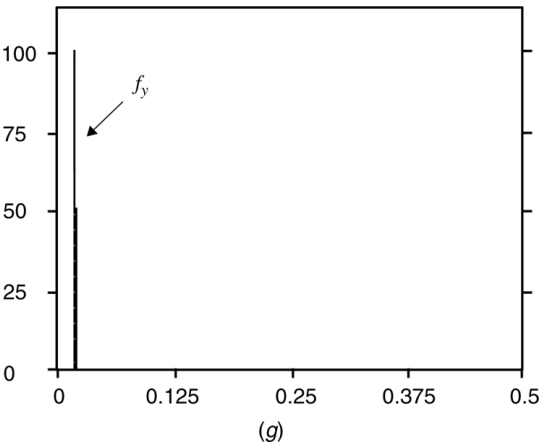
FIGURE 10.6. *Continued.*



(*e*)

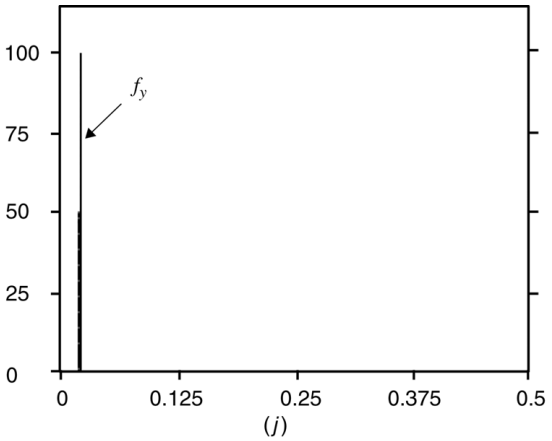


(*f*)

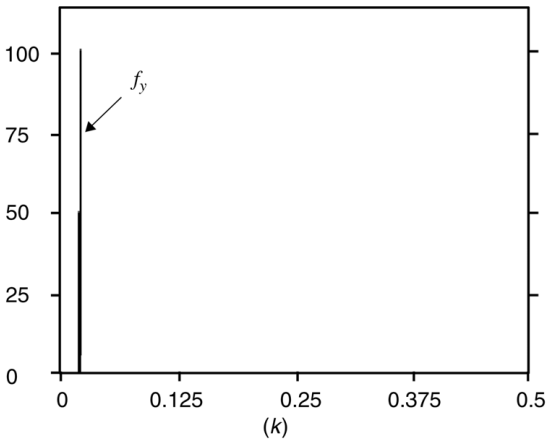


(*g*)

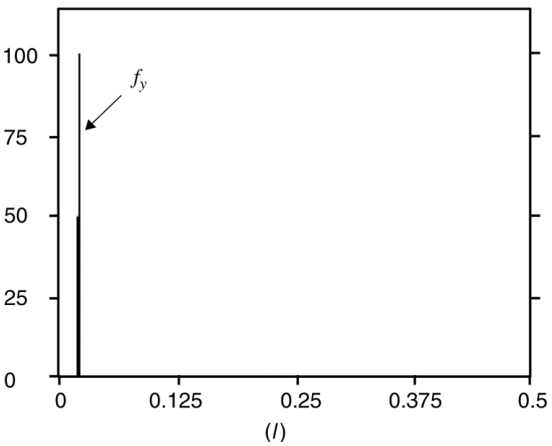
FIGURE 10.6. *Continued.*



(*j*)



(*k*)



(*l*)

FIGURE 10.6. *Continued.*

three-component interaction between T-antigen, *p53*, and *pRb* proteins can be considered as a crucial condition in the process of brain tumor formation.

Also it can be observed from the two cross-spectral functions of agnoprotein and T-antigen proteins (Figs. 10.4*a* and 10.4*b*) that characteristic frequencies of these protein groups are different, leading to the conclusion that agnoprotein and T-antigen protein will not interact with each other (Table 10.5). Interestingly to, the same conclusion was withdrawn after finalizing the experimental study using the T-antigen and agnoprotein [24]. Results of our computational analysis suggest that only a four-component interaction between agnoprotein, T-antigen, *p53*, and *pRb* proteins (Fig. 10.4*e*) gives the same characteristic frequency at $f = 0.2021$ (characteristic feature of tumor development). Therefore it can be postulated that a possible interaction between agnoprotein and *pRb* tumor suppressor proteins may lead to the inactivation of *pRb* functionality that can initiate a process of a tumor formation.

10.3.3. HSP and Tumour Suppressor Protein Interactions

Here 30 HSP sequences, 13 *p53* protein sequences, and 9 *pRb* protein sequences were investigated concerning the understanding of the structure–function relationship within these proteins. A multiple cross-spectral analysis was performed for each selected protein group as well as for their mutual combination using the EIIP values (Figs. 10.7*a* to 10.7*f*). As a result, characteristic frequencies of analyzed protein groups were obtained and are shown in Table 10.6. Also Continuous Wavelet Transform (CWT) was used for the determination of functional active sites of mouse HSP70/HSP90 protein (Fig. 10.8).

Initially the structure–function analysis was applied to a group of 30 HSP sequences, and two peak frequencies were identified in their multiple cross-spectrum: $f_1 = 0.0195 \pm 0.033$ characterizing HSP's immunoregulatory activity and $f_2 = 0.4248 \pm 0.033$ (the same frequency was identified for Fibroblast Growth Factor (FGF) proteins in our previous studies [9, 19]) that characterize the ability to regulate cell growth and differentiation (Fig. 10.7*a*).

The same procedure was repeated with *p53* and *pRb* tumor suppressor proteins (Figs. 10.7*b* and 10.7*d*). The *p53* tumor suppressor gene is one of the genes most often mutated in human cancers. Its mainly point mutations lead to amino acid substitutions in the central region of the protein and thus cause its abnormal function. The prominent characteristic frequencies of *p53* and *pRb* tumor suppressor proteins were identified at $f = 0.4326 \pm 0.077$ and $f = 0.4316 \pm 0.111$, respectively.

The similarity of characteristic frequencies of *p53* and *pRb* proteins (Figs. 10.7*b* and 10.7*d*) is expected as both *p53* and *pRb* proteins have the same biological function—tumor suppression. Thus, it has been suggested that the frequency $f = 0.4316$ identified within the RRM analysis be considered as a characteristic feature of the specific biological activity of *p53* and *pRb* proteins—regulation of cells growth and proliferation—and consequently ability to prevent tumor formation.

As was mentioned above in the consensus spectrum of HSPs (Fig. 10.7*a*), we can observe two peaks with different amplitude ratios. One prominent peak at

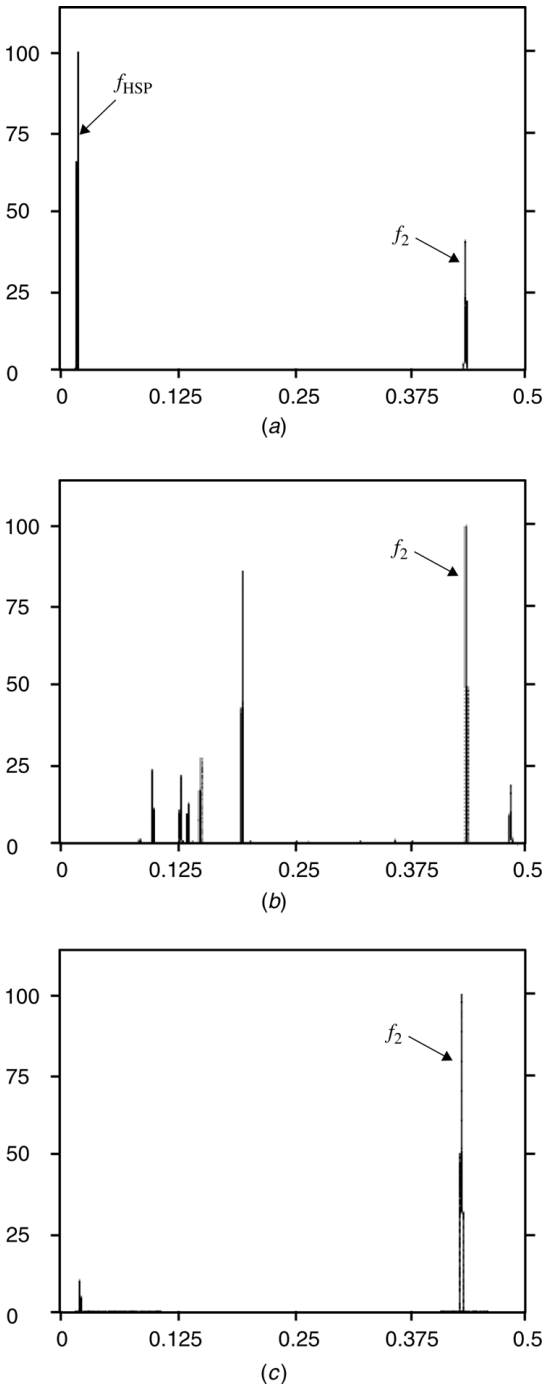
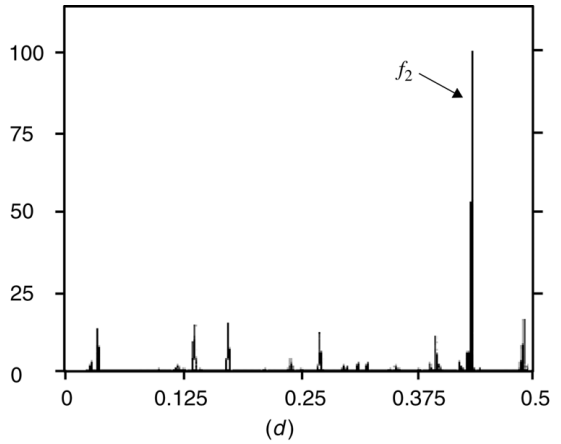
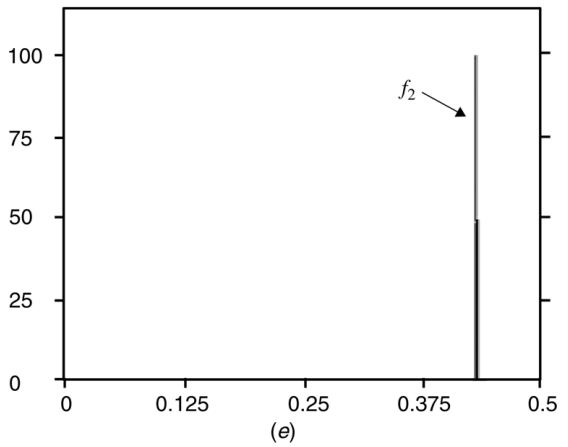


FIGURE 10.7. Multiple cross-spectral function of protein groups: (*a*) HSP proteins, (*b*) *p53* proteins, (*c*) HSP and *p53* proteins, (*d*) *pRb* proteins, (*e*) HSP and *pRb* proteins, and (*f*) HSP, *p53* and *pRb* proteins. The prominent peaks were found for HSPs at $f_{HSP} = 0.0195$ and HSP, *p53*, *pRb* at $f_2 = 0.4316$.





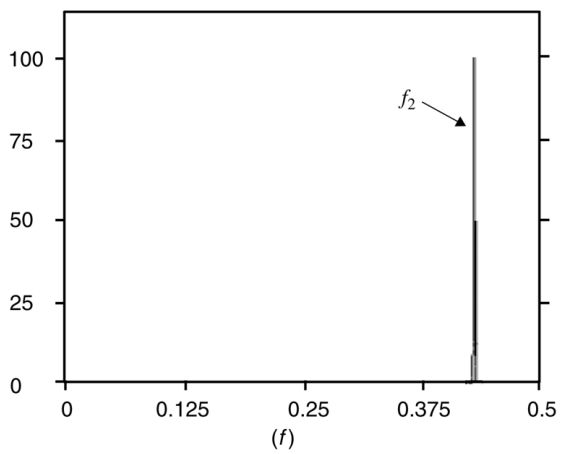


FIGURE 10.7. *Continued.*

TABLE 10.6 Peak frequency and Signal-to-Noise Values of Proteins Analysed

Protein Group	Frequency	S/N	No. Sequence	Standard Error, 1/No.
HSP	0.0195	277.24	30	0.033
p53	0.4326	159.97	13	0.077
HSP, *p53*	0.4277	295.95	43	0.023
pRb	0.4316	164.28	9	0.111
HSP, *pRb*	0.4287	505.94	39	0.026
HSP, *p53*, *pRb*	0.4287	438.19	51	0.020

$f = 0.0195 \pm 0.033$ corresponds to the HSP's common biological activity—immunological ability to defend cells against environmental stresses. However, the existence of another less significant peak at $f = 0.4248 \pm 0.033$ reveals that HSPs can participate in more than one biological process (interact with other proteins). This frequency identified for HSPs is of great importance as it is the same (within the calculation error) as was determined for *p53* and *pRb* tumor suppressors (Figs. 10.7*b* and 10.7*d*). This would confirm the RRM main postulate that proteins and their targets recognize/interact with each other on the basis of the same (similar) characteristic frequency underlying the possibility for HSPs and *p53* and *pRb* proteins to be involved in the same biological process (interact with each other). Analyzing the mutual interactions between HSPs and *p53* and *pRb*, respectively

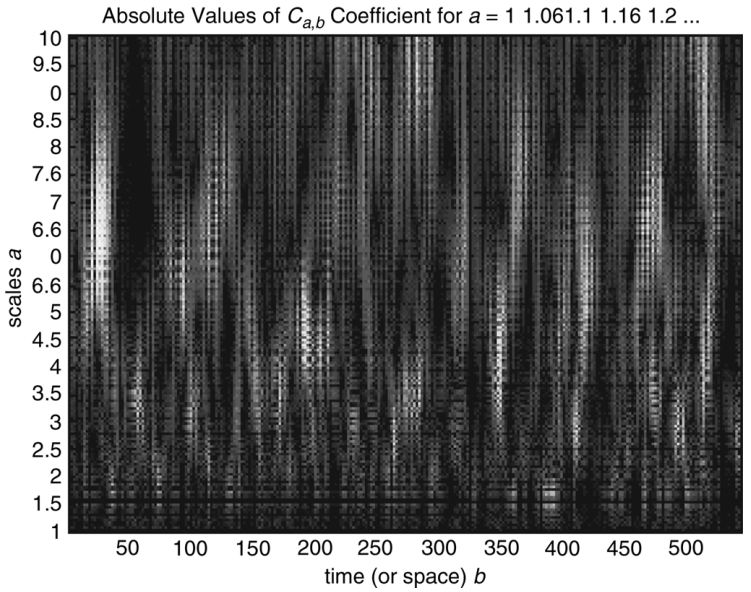


FIGURE 10.8. CWT scalogram of *Hsp70/Hsp90* protein (mouse) using Morlet function. The abscissa represents the position of amino acids in the protein molecule; ordinate is the continuous scale (from 1 to 10).

(Table 10.6), we obtain the same characteristic frequency $f = 0.4287$ (within the calculation error) for these analyzed proteins. From Figures 10.7*c*, 10.7*e*, and 10.7*f* we can observe one dominant peak in each cross-spectral function, and all analyzed sequences within the group have this frequency component in common (share the common biological activity).

Therefore, we can conclude that this identified frequency $f = 0.4287$ can be considered a characteristic feature of the three-component mutual interactions between HSPs and *p53* and *pRb* proteins, revealing the ability of these proteins to defend the cell against cancer by suppressing tumor formation. Consequently, HSPs may play a key role in the development of antitumor vaccines for very specific types of cancer.

10.3.4. Application of Wavelet Transform to Predict Protein Functional Epitopes

It is known that a protein active site is usually built up of domain(s) within the protein sequence. In our previous work [7–10] we demonstrated that by applying the wavelet transform to the particular protein molecule we are able to observe a whole frequency/spatial distribution and thus identify domains of high energy for the particular RRM frequency along the protein sequence. These energy concentrated regions in the CWT scalogram were proposed as the most critical locations of the protein's biological functions. Results of our previous work suggested that the Morlet wavelet is the most suitable wavelet function in the analysis of protein active sites or domains [10–13, 21].

Here we applied the CWT approach to identify the functional epitopes (active sites) of mouse HSP70/HSP90 protein (NP_058017, Entrez-Protein Database). From Figure 10.8 we can observe high-energy locations for two characteristic frequencies identified for HSPs using the RRM: for $f_1 = 0.0195$ the regions of amino acids are at 225–235, 275–295, and 525–535; for $f_2 = 0.4248$ they are at 260–280, 410–415, and 500–505. These computationally predicted locations correspond to the tetratricopeptide repeat (TPR) domain and binding motif identified experimentally by other authors [7].

This study extends the application of the RRM approach to analysis of the possible effect of HSPs on the biological functionality of *p53* and *pRb* tumor suppressor proteins. The common frequency identified for HSPs and *p53* and *pRb* proteins implies that according to the RRM concepts HSPs can be involved in the interactive biological process with tumor suppressor proteins. Thus, it indicates that HSPs may play a key role in the process of tumor growth arrest and consequently may be a crucial component for antitumor vaccine development. Also the wavelet transform incorporated into the RRM presents an invaluable tool for computational allocation of protein active/or binding sites.

10.4. CONCLUSION

We have shown previously that digital signal-processing methods can be used to analyze linear sequences of amino acids to reveal the informational content of proteins [8, 9]. This study extends the utility of the RRM procedures to the

structure–function analysis of viral proteins, HSPs, and tumour suppressor *p53* and *pRb* proteins.

The results of our computational analysis clearly indicate that the RRM can determine the protein characteristic frequencies crucial for biological activity/interaction of analyzed proteins. Hence, we can suggest that the RRM presents an engineering tool based on digital signal processing that is able to identify the protein characteristic patterns in protein sequences related to the common biological function/interaction of studied proteins. Consequently, knowing the protein characteristic frequency allows us to allocate the protein's biological active site(s) and design new peptides with the desired biological function. This novel prediction scheme can be used to facilitate the structure–function studies under different protein families and thus save the experimental cost greatly.

According to the RRM postulates, macromolecular interactions present the transfer of resonant energy between interacting molecules. These energies are electromagnetic in nature. It is known that sunlight is the main cause and support of life. Heat from the sun and sunlight are the main catalysts for the transformation of inorganic substances into organic ones [9]. However, macromolecules, mainly proteins, through their highly selective interactions with different targets, drive most of the processes of life. If the first organic molecules performed their biological function within the energy range of sunlight at the dawn of life, we can expect that more complicated organic molecules, though with the same bioactivity, will maintain their functioning within the same energy (frequency) range.

All the results obtained so far with the RRM applications lead to the conclusion that each macromolecular biological process inside a living cell or tissue is characterized by one frequency within a very wide frequency range from the extremely low infrared to the ultraviolet. Heat from the sun and sunlight are electromagnetic radiation emitted in a range of frequencies from the extremely low infrared to the ultraviolet, the same range of frequencies predicted by the RRM to be responsible for protein–protein and protein–DNA interactions, that is, the main intermolecular processes of life. We can speculate that perhaps life's intermolecular processes are carried out at their own frequencies and these frequencies lie within the sunlight frequency range [9].

The concepts presented here might lead to a completely new perspective on life processes at the molecular level. Consequently, they represent a step toward a better understanding of biological processes and biomolecular interactions. This new approach has enormous implications in the fields of molecular biology, biotechnology, medicine, agriculture, and the pharmacology industry. The ability to improve the predicted activity rates, alter substrate specificity, and design activity-modulating peptides purely from mathematical modeling via rational mechanism forms the innovative basis of the RRM methodology.

REFERENCES

1. S. Croul, F. D. Lublin, L. Del Valle, R. J. Oshinsky, A. Giordano, K. Khalili, and C. K. Ritchie, "The cellular response of JC virus T-antigen induced brain tumor implants to a murine intra-ocular model," *J. Neuroimmunol.*, 106: 181–188, 2000.

2. B. Krynska, L. Del Valle, J. Gordon, J. Otte, S. Croul, and K. Khalili, "Identification of a novel p53 mutation in JCV-induced mouse medulloblastoma," *Virology*, 274: 65–74, 2000.
3. L. Del Valle, S. A. Azizi, B. Krynska, S. Enam, S. E. Croul, and K. Khalili, "Reactivation of human neurotropic JC virus expressing oncogenic protein in a recurrent glioblastoma multiforme," *Ann. Neurol.*, 48: 932–936, 2000.
4. P. David, S. Lane, and K. Lain, "Therapeutic exploitation of the p53 pathway [Review]," *Trends Mol. Med.*, 8/4: 38–42, 2002.
5. C. Jolly and R. I. Morimoto, "Role of the heat shock response and molecular chaperones in oncogenesis and cell death," *J. Natl. Cancer Inst.*, 92/19: 1564–1572, 2000.
6. Y. Tamura, "Immunotherapy of tumors with autologous tumor-derived heat shock protein preparations," *Science*, 278: 117–120, 1997.
7. O. O. Odunugu, J. A. Hornby, C. Bies, R. Zimmermann, D. J. Pugh, and G. L. Blatch, "Tetratricopeptide repeat motif-mediated Hsc70-mSTI1 interaction. Molecular characterization of the critical contacts for successful binding and specificity," *J. Biol. Chem.*, 278/9: 6896–6904, 2003.
8. I. Cosic, "Macromolecular bioactivity: Is it resonant interaction between molecules? Theory and applications," *IEEE Trans. Biomed. Eng.*, 41: 1101–1114, 1994.
9. I. Cosic, *The Resonant Recognition Model of Macromolecular Bioactivity*, Birkhauser Verlag, Basel, Switzerland, 1997.
10. I. Cosic, C. H. De Trad, Q. Fang, and M. Akay, "Protein sequences analysis using the RRM model and wavelet transform methods: a comparative study analysis," in *Proceedings of the IEEE-EMBS Asia-Pacific Conference on Biomedical Engineering*, Hangzhou, China, 2000, pp. 405–406.
11. I. Cosic and Q. Fang, "Evaluation of different wavelet constructions (designs) for analysis of protein sequences," in *Proceedings of the Fourteenth International Conference on DSP*, Santorini, Greece, 2000.
12. C. H. De Trad, Q. Fang, and I. Cosic, "The resonant recognition model (RRM) predicts amino acid residues in highly conservative regions of the hormone prolactin (PRL)," *Biophys. Chem.*, 84/2: 149–157, 2000.
13. C. H. De Trad, Q. Fang, and I. Cosic, "Protein sequences comparison based on the wavelet transform approach," *Protein Eng.*, 15/3: 193–203, 2002.
14. I. Cosic, M. Pavlovic, and V. Vojisavljevic, "Prediction of 'hot spots' in Il-2 cased on information spectrum characterictics of growth regaling factors," *Biochemie*, 71: 333–342, 1989.
15. I. Cosic and M. T. W. Hearn, "'Hot spot' amino acid distribution in Ha-ras oncogene product p21: Relationship to guanine binding site," *J. Mol. Recognition*, 4: 57–62, 1991.
16. I. Cosic, A. N. Hodder, M. A. Aguilar, and M. T. W. Hearn, "Resonant recognition model and protein topography: Model studies with myoglobin, hemoglobin and lysozyme," *Eur. J. Biochem*, 198: 113–119, 1991.
17. I. Cosic, "Virtual spectroscopy for fun and profit," *Biotechnology,* 13: 236–238, 1995.
18. V. Krsmanovic, J. M. Biquard, M. Sikorska-Walker, I. Cosic, C. Desgranges, M. A. Trabaud, J. F. Whitfield, J. P. Durkin, A. Achour, and M.T.W. Hearn, "Investigation into the cross-reactivity of rabbit antibodies raised against nonhomologous pairs of synthetic peptides derived from HIV-1 gp120 proteins," *J. Peptide Res.*, 52(5): 4110–4120, 1998.

19. I. Cosic, A. E. Drummond, J. R. Underwood, and M. T. W. Hearn, "In vitro inhibition of the actions of basic FGF by a novel 16 amino cid peptide," *Mol. and Cell. Biochem.*, 130: 1–9, 1994.
20. L. Veljkovic and M. Slavic, "General model of pseudopotentials," *Phys. Rev. Lett.*, 29: 105–108, 1972.
21. E. Pirogova, Q. Fang, M. Akay, and I. Cosic, "Investigation of the structure and function relationships of oncogene proteins," *Proc. IEEE*, 90/12: 1859–1867, 2002.
22. E. Pirogova and I. Cosic, "The use of ionisation constants of amino acids for protein signal analysis within the resonant recognition model—Application to proteases," *Mol. Simulation*, 28(8–9): 845–851, 2002.
23. E. Pirogova, G. P. Simon, and I. Cosic, "Investigation of the applicability of dielectric relaxation properties of amino acid solutions within the resonant recognition model," *IEEE Trans. NanoBiosci.*, 2/2: 63–69, 2003.
24. L. Del Valle, J. Gordon, S. Enam, S. Delbue, S. Croul, S. Abraham, S. Radhakrishnan, M. Assimakopoulou, C. D. Katsetos, and K. Khalili, "Expression of human neurotropic polyomavirus JCV late gene product agnoprotein in human medulloblastoma," *J. Nat. Cancer Inst.*, 94(4): 240–273, 2002.
25. P. Lissoni, "Therapeutic potential of melatonin," *Frontiers Hormone Res.*, 23: 1997.
26. S. M. Webb and M. Puig-Domingo, "Role of melatonin in health and disease," *Clin. Endocrinol.*, 42: 221–234, 1995.
27. P. Lissoni, S. Barni, G. Tancini, E. Mainini, F. Piglia, G. J. Maestroni, and A. Lewinski, "Immunoendocrine therapy with low-dose subcutaneous interleukin-2 plus melatonin of locally advanced or metastatic endocrine tumours," *Oncology*, 52/2: 163–166, 1995.
28. P. Lissoni, S. Barni, M. Cazzaniga, A. Ardizzoia, F. Rovelli, F. Brivio, and G. Tancini, "Efficacy of the concomitant administration of the pineal hormone melatonin in cancer immunotherapy with low dose IL-2 in patients with advanced solid tumours who had progressed on IL-2 alone," *Oncology*, 51/4: 344–347, 1994.

INDEX

Acute lymphoblastic leukemia (ALL) *vs.* AML, 124f
Acute myeloid leukemia (AML) *vs.* ALL, 124f
  predicting treatment response, 36
Adaptive quality-based clustering (AQBC), 55, 69–70, 74, 75t
Additive white Gaussian noise (AWGN), 175
Affymetrix U94A microarrays, 102f
Aldesleukin, 269
Algorithms, 73
AlignACE, 83–84, 87t
ALL. *See* Acute lymphoblastic leukemia (ALL)
All-zero syndrome
  average syndrome distance, 201f
Amino acids
  electron–ion interaction potential, 263t
AML. *See* Acute myeloid leukemia (AML)
Analyzing microarrays
  VxInsight, 117–118
ANN. *See* Artificial neural network (ANN); Artificial neural networks (ANN)
AQBC. *See* Adaptive quality-based clustering (AQBC)
Artificial neural network (ANN), 35, 248f
  feed-forward, 247
Artificial neural networks (ANN), 247
AWGN. *See* Additive white Gaussian noise (AWGN)

BACIIS. *See* Biological and Chemical Information Integration System (BACIIS)
Basic sequence model, 82f
Baum–Welch algorithm, 243
Bayesian approach
  model-based clustering, 68
Bayesian classification scheme, 244, 245
Bayesian information criterion (BIC), 68
BIC. *See* Bayesian information criterion (BIC)
Binary block code, 186f
Binary random sequence, 187f
Binary table-based convolutional code models
  translation initiation, 202f
Binding sites, 56, 58
  common coregulated genes, 76–87
Binding vectors
  to code words
    inverse ECC systems, 199
BioCyc, 6
BioKleisli, 210, 222
Biological and Chemical Information Integration System (BACIIS), 211–212
  knowledge base, 222
    architecture, 212–213, 212f
    cost model, 223
Biological coding theory applications, 203–204
Biological data sources
  functional genomics and proteomics, 5–6
Biological queries, 16t–17t
BioProspector, 85, 87t
BLAST, 236
Blocks substitution matrix (BLOSUM), 237, 239f
BLOSUM. *See* Blocks substitution matrix (BLOSUM)

Genomics and Proteomics Engineering in Medicine and Biology. Edited by Metin Akay

Bootstrap method, 119f
Boundedness, 20
Bursty channel, 175

Cancer simulation, 131–132
CAST, 64
CATH. *See* Class Architecture Topology Homology (CATH)
cDNA. *See* Complementary DNA (cDNA)
Cell cycle, 259f
  constant radiosensitivity throughout, 144
Cell nucleus, 258f
Channel transition probability, 191t
CHIME, 242
Cis-regulatory elements
  searching for modules, 85–86
Class Architecture Topology Homology (CATH), 231
Class predictors
  constructing, 34–35
CLUSTALX, 236
Cluster analysis, 64
Clustering, 56
  algorithms, 65t, 70
  expression profiles, 64–65
  finding most representative, 111, 112f
  gene expression profiles, 63–70
  microarray data, 103–104
  parameters, 108–109
  quality-based, 64, 68–69
  significance, 110–111
  validation, 70–75
  VxOrd, 104
CLUSTLAW, 236
Coding theory, 203–204
Combinatorial approach
  oligonucleotide frequency analysis, 79–80
Common binding sites
  coregulated genes, 76–87
Communication channel, 175
Communication system, 174f
  characteristics, 186–187
Comparative modeling, 228
Complementary DNA (cDNA), 60
  schematic overview, 61f
Complex life science multidatabase queries, 209–223
  architecture, 212–213
  multidatabase queries
    future trends, 222
Component ontologies
  functional genomics and proteomics, 3–5
Compound channel, 175
Computational analysis of proteins, 227–252
  classification and prediction, 242–247
  databases, 229–232
  future trends, 252
  modeling, 241–242
  natural language processing, 248–250
  sequence alignment, 235–240
  sequence motifs and domains, 232–235
Configuration, 4
CONSENSUS, 81, 87t
Continuous Wavelet Transform (CWT), 280
  Hsp70/Hsp90 protein, 283f, 284–285
  predicting protein functional epitopes, 284
Convolutional codes, 178
Coregulated genes
  common binding sites, 76–87
  upstream region, 77f
CPN tools, 5
CWT. *See* Continuous Wavelet Transform (CWT)

Databases
  federated
    cost models for query planning, 222
    integrate life science Web databases, 214
    weaknesses, 222
  integrate life science Web
    federated database, 214
  list generation, 216–217
  MIPS, 74
  sequence, 230
Data mart, 219
Data-warehousing, 209–210, 219
  response times, 222
Decoder, 175–176
Decoding methodology, 178–179
Definition of secondary structure of proteins (DSSP), 244
  secondary-structure formations, 229f

Deoxyribonucleic acid (DNA), 25, 257
  computing, 203–204
  double-helix hybridization, 59
Deterministic motifs, 234
Dictionary model
  genomewide motif identification, 157–172
DiscoveryLink, 210, 222
Discrete memoryless channel (DMC), 191
DMC. *See* Discrete memoryless channel (DMC)
DNA. *See* Deoxyribonucleic acid (DNA)
Domains, 228
  sequence
    computational analysis of proteins, 232–235
  structural, 233
Double-helix hybridization
  DNA, 59
DSSP. *See* Definition of secondary structure of proteins (DSSP)
Dunn's validity index, 71
Dynamic programming algorithms, 236–237

EBI. *See* European Bioinformatics Institute (EBI)
Efficiency query execution plan
  early intersection, 220f
EM. *See* Expectation–maximization (EM)
Empirical risk minimization (ERM), 248
Encoder, 174
Encoding methodology, 176, 179–180
Enhancers, 58
Entrez, 6, 232
Enumeration approach
  oligonucleotide frequency analysis, 78–79
Enzyme Nomenclature, 217
Equivalence classes, 149
ERM. *See* Empirical risk minimization (ERM)
Error control
  codes and genome, 173–206
  coding methods for genomic sequence and system analysis, 204
  and communication, 173–202
  in genomic sequence, 185–186
Error-correcting codes, 174
Escherichia coli K-12
  initiation regions, 189f
  noninitiating intergenic regions, 188f
EST. *See* Expressed sequence tags (EST)
Eukaryotic genes
  mutation rate comparison to genome size, 190f
  structure, 58f
Eukaryotic replication channels
  capacity, 192f
European Bioinformatics Institute (EBI), 230
E-value. *See* Expected value (E-value)
Expectation–maximization (EM), 56, 82–83, 235
Expected value (E-value), 237
Expressed sequence tags (EST), 60
Extended Gibbs sampling methods, 83–84

FastA, 236
Fast orthogonal algorithm (FOA), 28
Fast orthogonal search (FSO), 25
Feature selection, 118
Federated databases
  cost models for query planning, 222
  integrate life science Web databases, 214
  weaknesses, 222
Feed-forward artificial neural networks, 247
Figure of merit (FOM), 72
Filtering, 63
Fingerprint, 233, 235
First-generation algorithms, 64
FOA. *See* Fast orthogonal algorithm (FOA)
Fold recognition, 244
FOM. *See* Figure of merit (FOM)
Footprinting
  phylogenetic, 86
Fractionation scheme
  standard, 145f
Frame shifting, 2
FSO. *See* Fast orthogonal search (FSO)
Functional categories
  enrichment, 73–75

Functional genomics and proteomics, 1–21
biological data sources, 5–6
component ontologies, 3–5
methods and tools, 3–5
modeling approach and results, 6–18
qualitative knowledge models, 1–21
querying the model, 16–17
representing abnormal functions and processes, 9–10
representing high-level clinical phenotypes, 15
representing levels of evidence for modeled facts, 16
representing molecular complexes, 9
representing mutations, 6–7
representing nucleic acid structure, 7
simulating the model, 19
Functional inversion, 196–197
Fuzzy C-means, 64

GA. *See* Genetic algorithm (GA)
Gapped motifs, 78
Garlic, 222
Gatlin's communication model, 181
Gaussian infinite mixture model (GIMM), 68
GBM. *See* Glioblastoma multiforme (GBM)
GDMB. *See* Genome Database (GDMB)
GeneCards, 222
Gene Expression Datamart, 219
Gene List Exploration Environment (GLEE), 127
Genes
chips, 60
coregulated
common binding sites, 76–87
upstream region, 77f
eukaryotic
structure, 58f
expression profiles
clustering, 63–70
measuring, 59–61
prediction, 35–45
lists, 119f, 124f
comparing, 123
generating, 118
predicting medulloblastoma, 44t
regulation bioinformatics of microarrays, 55–90
shaving, 64
structure, 58
Genetic algorithm (GA), 200, 245–247, 246f
operator, 247f
Genetic error control system, 184–202
GENIES, 251
Genome Database (GDMB), 217
Genomewide motifs
identification, 157–158
dictionary model, 157–172
previous methods, 159–167
Genomics. *See* Functional genomics and proteomics
Genomics Unified Schema (GUS), 210, 219, 222
Geometric cell, 149
G-functionals, 27
Gibbs sampling, 56, 87t, 235
algorithms, 82–83
extended, 83–84
GIMM. *See* Gaussian infinite mixture model (GIMM)
GLEE. *See* Gene List Exploration Environment (GLEE)
Glioblastoma multiforme (GBM), 260
fractionation schemes, 142–144
Gram–Schmidt process
Volterra series, 26
Graph coarsening
adding to VxOrd, 112–117
Guilt-by-association, 55
GUS. *See* Genomics Unified Schema (GUS)

Halting, 2
Heat shock protein (HSP), 60, 260, 283
tumour suppressor protein interactions, 280–281
Heat shock proteins 70/90 (HSP70/90)
CWT, 283f, 284–285
Hidden Markov models (HMM), 234, 242, 243f
classifier, 245f
Hierarchical clustering, 64–65, 66f
High cell loss factor, 144

High-level clinical phenotypes
functional genomics and proteomics representing, 15
HMM. *See* Hidden Markov models (HMM)
Homology identification
natural language processing (NLP), 250
Homology modeling, 228
HSP. *See* Heat shock protein (HSP)

IBM SPLASH, 40
IL-2, 269, 272
INCLUSive Web tool, 56, 84, 87–88
Incyte, 219
Information integration layer
architecture, 213f
Initiation complex, 2
In silico radiation oncology, 131–150
Integral object composition, 4
Integrate life science Web databases
federated database, 214
Inverse ECC models, 193–196
Inverse ECC systems
from binding vectors to code words, 199
Inverse EC model II, 196–197
Inverse error control coding models, 193

Japanese International Protein Information Database (JIPID), 230
JC virus (JCV), 260, 265, 268
medulloblastomas, 268
JIPID. *See* Japanese International Protein Information Database (JIPID)

K-means, 64, 75t
clustering, 66
K-nearest neighbors (k-NN), 40, 47t

Laguerre functions, 28
Learning
supervised, 103
unsupervised, 103
Lee–Schetzen method, 27
Levels of evidence for modeled facts
functional genomics and proteomics representing, 16
Likelihood evaluation
algorithms, 164–167
Linear block codes, 176
Linear-quadratic (LQ) model, 140–141
LinkDB, 222
Link-driven systems, 222
Liveness, 20
LNL cascade, 30
LQ. *See* Linear-quadratic (LQ) model

May et al.'s communication model, 182, 183f
MCLUST software, 68
Mean-square error (MSE), 33
Mechanical wounding, 88t
microarray experiment, 88t
MEDLINE, 232
Medulloblastomas, 260
JCV, 268
predicting clinical outcome, 40–41
predicting metastasis, 43–46
Melatonin, 269
Member–bunch composition, 4
MEME. *See* Multiple Em for Motif Elicitation (MEME)
Memoryless channel, 175
Messenger ribonucleic acid (mRNA), 2
leader
functional definition, 196–197
structure, 58f
Microarrays
Affymetrix U94A, 102f
analyzing
VxInsight, 117–118
basic analysis, 103
clustering, 103–104
experiment
mechanical wounding, 88t
experiments, 100–101
gene regulation bioinformatics of, 55–90
nonlinear system identification, 25–48
constructing class predictors, 34–35
gene expression profiling prediction, 35–46
parallel cascade identification, 30–33
predictors comparison, 46–47
online integrated analysis, 87–88
preprocessing, 61–63
robust analysis, 99–128
Microbial genome mutation rate
comparison to genome size, 190f

Minorization–maximization algorithm
parameter estimation, 167–171
MIPS. *See* Munich Information Center for Protein Sequences (MIPS)
Misreading, 2
Missing-value replacement, 62–63
MITOMAP, 6
Model-based clustering, 64, 68
Bayesian approach, 68
Modeling
functional genomics and proteomics, 6–18
Molecular biology, 203
Molecular complexes
functional genomics and proteomics representing, 9
Motifs, 56, 233
deterministic, 234
discovery, 235
finding, 56, 57f
algorithms, 82–83, 87t
recent advances, 85–86
software, 87
gapped, 78
genomewide
identification, 157–158, 157–172, 159–167
identification, 234–235
probabilistic, 234
probabilistic model of sequence, 80–81
sequence
computational analysis of proteins, 232–235
structural, 234
MotifSampler, 56, 84, 87t
mRNA. *See* Messenger ribonucleic acid (mRNA)
MSE. *See* Mean-square error (MSE)
Multidatabase queries
complex life science, 209–223
architecture, 212–213
multidatabase queries, 222
Multiple Em for Motif Elicitation (MEME), 81, 84, 87t, 235, 243
MULTIPROFILER, 79, 87t
Munich Information Center for Protein Sequences (MIPS), 74, 230
Mutations
functional genomics and proteomics representing, 6–7

National Center for Biotechnology Information (NCBI), 210, 222, 232
Natural language processing (NLP), 127, 248–249
computational analysis of proteins, 248–250
gene function extraction, 252
homology identification, 250
protein identification, 249–250
protein–protein interaction identification, 251f
relation/pathway identification, 250
synonym identification, 250
NCBI. *See* National Center for Biotechnology Information (NCBI)
Neighbor GC tumor (NGCT), 142
NGCT. *See* Six-neighbor GC tumor (NGCT)
NLP. *See* Natural language processing (NLP)
Nonlinear system identification
microarray data, 25–48
constructing class predictors, 34–35
gene expression profiling prediction, 35–46
parallel cascade identification, 30–33
predictors comparison, 46–47
Nonlinear transformation, 62
Nonparametric methods, 26
Normalization, 62, 101–102
Nucleic acid structure
functional genomics and proteomics representing, 7
Nucleotides
electron–ion interaction potential, 263t

Oligonucleotides, 56
frequency analysis
combinatorial approach, 79–80
enumeration approach, 78–79
OMIM. *See* Online Mendelian Inheritance in Man (OMIM)
Oncogene, 269
Online integrated analysis
microarray data, 87–88

Online Mendelian Inheritance in Man (OMIM), 6, 105, 211, 217
Ontologies, 1
  component
    functional genomics and proteomics, 3–5
Open reading frames (ORF), 228
ORF. *See* Open reading frames (ORF)

P53, 144–148
Pairwise alignment, 232
PAM. *See* Point accepted mutation (PAM)
PAM250 substitution matrix, 240f
Parallel cascade identification (PCI), 30–33, 31f, 47t
Parallel cascade ranking
  test expression profiles, 39t
Participants/role model, 3
Path generation, 217–218
Pattern, 233
PCI. *See* Parallel cascade identification (PCI)
PDB. *See* Protein Data Bank (PDB)
Peak frequency
  protein groups, 268t
Penalties, 237
Petri Net, 3, 4, 18f
  translation into, 5
Philosphiae Tumoralis Principia Algorithmica, 131–132
Phylogenetic footprinting, 86
PIR. *See* Protein Information Resource (PIR)
Place–area composition, 4
PlantCARE, 89
Point accepted mutation (PAM), 237
Position weight matrix (PWM), 234
Prediction-based method, 245
Predictors
  data set, 46–48
Preprocessing, 101–102
Probabilistic methods, 80–81
Probabilistic model of sequence motifs, 80–81
Probabilistic motifs, 234
Probabilistic sequence models, 56
Probabilistic suffix trees, 235
Profiles, 233
Prokaryotic replication channels
  capacity, 192f
Promoter regions, 56, 58
PROSITE, 230–231, 234
Protégé-2000 knowledge-modeling tool, 4
Protein Data Bank (PDB), 210, 231
Protein Information Resource (PIR), 230
Proteins. *See also* Computational analysis of proteins
  classification, 235
  configuration, 227
  folding, 228
  homology, 228
  identification
    NLP, 249–250
  multiple cross-spectral functions of, 266f–267f, 269f–271f, 273f–274f, 276f–279f, 281f–282f
  peak frequency and SNR, 268t, 272t, 274t, 283t
Protein Sequence Database (PSD), 230
Proteomics. *See* Functional genomics and proteomics
PSD. *See* Protein Sequence Database (PSD)
P53 tumor suppressor gene, 265
PWM. *See* Position weight matrix (PWM)

Qualitative knowledge models
  functional genomics and proteomics, 1–21
Quality-based clustering, 64, 68–69
Query, 15f
  decomposition, 215–216
  decomposition tree, 215f
  execution plans, 214–215
  functional genomics and proteomics, 16–17

Radiation therapy, 131–150
  in vivo, 135–136
RAMSOL, 241f, 242
Rand index, 71–72
Realistic sequence models, 77–78
REDUCE, 87t
Relation/pathway identification
  NLP, 250
Replication, 257
Repressors, 58

Rescaling, 63
Resonant recognition model (RRM), 260–265, 284
  frequencies, 264t
Response times
  data warehouses, 222
Reverse engineering, 184–202
Ribosomal interaction
  functional definition, 196–197
Ribosome as block decoder, 193–196
Rose scale, 34
RRM. *See* Resonant recognition model (RRM)
RSA tools, 87t

SAM. *See* Sequence Alignment and Modeling (SAM)
SARAH. *See* Simultaneously axially and radially aligned hydrophobicities (SARAH) scales
SCOP. *See* Structural Classification of Proteins (SCOP)
Second-generation algorithms, 67
Self-organizing maps (SOM), 64, 67
Self-organizing tree algorithm (SOTA), 64, 67
Sensitivity analysis, 72–73
SeqStore, 219
Sequence Alignment and Modeling (SAM), 235, 243
Sequence-based methods, 244
Sequence databases, 230
Sequence motifs and domains
  computational analysis of proteins, 232–235
Sequence Retrieval System (SRS), 210, 222, 232
Shannon's channel coding theorem, 187
Signal-to-noise ratio (SNR)
  protein groups, 268t
Signature, 233
Silhouette coefficients, 71
Similarity measure
  choosing, 104
  postprocessing, 105
Simulated annealing, 64
Simulating cancer on computer
  algorithmic principles, 131–135
  literature review, 133–134
Simulation model
  functional genomics and proteomics, 19
  parametric testing, 142–144
Simulation outline, 140–142
Simultaneously axially and radially aligned hydrophobicities (SARAH) scales, 34
Single-nucleotide background model based on base frequency (SNF), 84
Six-neighbor GC tumor (NGCT), 142
SNF. *See* Single-nucleotide background model based on base frequency (SNF)
SNR. *See* Signal-to-noise ratio (SNR)
Solid tumor in vivo
  biology, 137–138
  radiobiology, 140–150
SOM. *See* Self-organizing maps (SOM)
SOTA. *See* Self-organizing tree algorithm (SOTA)
Space-filling models, 241, 241f
SPARC/osteonectin, 43
SP-STAR, 79
SRM. *See* Structural risk minimization (SRM)
SRS. *See* Sequence Retrieval System (SRS)
Standard fractionation scheme, 145f
Standardization, 63
STRIPS-like planning, 219
Structural Classification of Proteins (SCOP), 231–232
Structural domains, 233
Structural motifs, 234
Structural profile, 234
Structural risk minimization (SRM), 248
Structure-based methods, 244
Substitution matrix, 237
Suffix trees, 80
Sum-of-square criterion of K-means, 71
Supervised learning, 103
Supervised methods, 117–118
Support vector machine (SVM), 40, 248
Support vector machine recursive feature elimination (SVM RFE), 124
SVM. *See* Support vector machine (SVM)
SVM RFE. *See* Support vector machine recursive feature elimination (SVM RFE)
Swiss-PdbViewer, 242

Swiss-Prot, 230, 231
Symmetric channel, 175
Synonym identification
  natural language processing (NLP), 250

TAMBIS. *See* Transparent Access to Multiple Biological Information Sources (TAMBIS)
T-antigen, 260
Ternary complex, 2
Tertiary-structure components, 7f
Test expression profiles
  parallel cascade ranking, 39t
Testing cluster coherence, 70–71
TFBS. *See* Transcription factor binding sites (TFBS)
Threading, 228
Threshold sampler, 85
Tokens, 4
Transcription, 58, 257
Transcriptional regulation, 57–58
Transcription factor binding sites (TFBS), 59, 78f
Transcription factors, 56, 58
Transcription processes
  initiation, 59f
Transfer ribonucleic acid (tRNA), 2, 7f
Translation initiation
  binary table-based convolutional code models, 202f
Translation processes, 257
  process diagram, 11f, 12f, 13f
Transparent Access to Multiple Biological Information Sources (TAMBIS), 4, 7, 9, 210, 211, 222
tRNA, transfer ribonucleic acid (tRNA)
Tumor cells
  cytokinetic model, 138f
  three-dimensional visualization, 143f
Tumor growth
  data collection and preprocessing, 135
  data visualization, 135–136
  radiation therapy in vivo, 135–136
Tumor suppressor gene
  p53, 265
Tumor suppressor proteins
  interactions, 269
  viral interactions, 265
Tumor/tumor suppressor protein interactions computation analysis, 257–285
  methodology, 260–265
  results, 265–270

UEP. *See* Unequal error protection (UEP)
UMLS. *See* Unified Medical Language System (UMLS)
Unequal error protection (UEP), 200
Unified Medical Language System (UMLS), 4
Unified model, 160–164
UniProt/Swiss-Prot Protein knowledgebase, 6
Unsupervised learning, 103
Untranslated regions (UTR), 58
Upstream region
  coregulated genes, 77f
UTR. *See* Untranslated regions (UTR)

Variable selection, 118
Viral suppressor proteins
  interactions, 269
Volterra series
  Gram–Schmidt process, 26
VxInsight
  analyzing microarray data, 117–118
VxOrd
  adding graph coarsening to, 112–117
  clustering, 104

Weeder, 87t
Well clustered, 71
WINNOWER, 79
Wire-frame models, 241
Woflan Petri Net verification tool, 5
Word-counting methods, 56
Workflow model, 3

YMF, 87t
Yockey's communication model, 181, 182f

ABOUT THE EDITOR

Metin Akay is a professor of bioengineering and interim department chairman of the Harrington Department of Bioengineering at the Fulton School of Engineering, Arizona State University at Tempe. He received his Bachelor of Science and Master of Science in Electrical Engineering from the Bogazici University, Istanbul, Turkey in 1981 and 1984, respectively, and a Ph.D. from Rutgers University in 1990.

He is the author/coauthor/editor of 14 books and has given more than 50 keynote, plenary, and invited talks at international meetings including the first, second and third Latin American Conference on Biomedical Engineering in 1998, 2001, and 2004.

Dr. Akay is the founding chair of the Annual International Summer School on Biocomplexity from System to Gene sponsored by the NSF and Dartmouth College and technically cosponsored by the IEEE EMBS of the Satellite Conference on Emerging Technologies in Biomedical Engineering. In 2003, he was also the founding chair of the International IEEE Conference on Neural Engineering and the first chair of the steering committee of the *IEEE Transaction on Computational Biology and Bioinformatics* sponsored by the IEEE (CS, EMBS, NN, etc.) and non-IEEE societies. He was the invited guest editor for the special issues of *Bioinformatics: Proteomics* and *Genomics Engineering* of the *Proceedings of IEEE*, one of the most highly cited IEEE journal.

Prof. Akay is a recipient of the IEEE EMBS Service Award, an IEEE Third Millennium Medal, and the IEEE Engineering in Medicine and Biology Society Early Career Achievement Award 1997. He also received the Young Investigator Award of the Sigma Xi Society, Northeast Region in 1998 and 2000.

Dr. Akay is a fellow of Institute of Physics, senior member of the IEEE, a member of BMES, Eta Kappa, Sigma Xi, Tau Beta Pi. He also serves on the editorial or advisory board of several international journals including the IEEE T-BME, IEEE T-NSRE, IEEE T-ITIB, *Proceedings of IEEE*, *Journal of Neural Engineering*, NIH Neural Engineering and Bioengineering partnership study sections and several NSF review panels.