

STATISTICAL BIOINFORMATICS

*For Biomedical
and Life Science Researchers*



Edited by JAE K. LEE

*STATISTICAL
BIOINFORMATICS*

*STATISTICAL
BIOINFORMATICS*
*A Guide for Life and Biomedical
Science Researchers*

Edited by

JAE K. LEE

 **WILEY-BLACKWELL**

A JOHN WILEY & SONS, INC., PUBLICATION

The cover figure signifies the statistical analysis and biological interpretation of high-throughput molecular data. It is the logo of Dr. John N. Weinstein's former research group at the US National Cancer Institute (in which Dr. Jae Lee worked) and his current Genomics & Bioinformatics Group at the M. D. Anderson Cancer Center.

Copyright © 2010 Wiley-Blackwell. All rights reserved

Published by John Wiley & Sons, Inc., Hoboken, New Jersey

Published simultaneously in Canada

No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning, or otherwise, except as permitted under Section 107 or 108 of the 1976 United States Copyright Act, without either the prior written permission of the Publisher, or authorization through payment of the appropriate per-copy fee to the Copyright Clearance Center, Inc., 222 Rosewood Drive, Danvers, MA 01923, (978) 750-8400, fax (978) 750-4470, or on the web at www.copyright.com. Requests to the Publisher for permission should be addressed to the Permissions Department, John Wiley & Sons, Inc., 111 River Street, Hoboken, NJ 07030, (201) 748-6011, fax (201) 748-6008, or online at <http://www.wiley.com/go/permission>.

Limit of Liability/Disclaimer of Warranty: While the publisher and author have used their best efforts in preparing this book, they make no representations or warranties with respect to the accuracy or completeness of the contents of this book and specifically disclaim any implied warranties of merchantability or fitness for a particular purpose. No warranty may be created or extended by sales representatives or written sales materials. The advice and strategies contained herein may not be suitable for your situation. You should consult with a professional where appropriate. Neither the publisher nor author shall be liable for any loss of profit or any other commercial damages, including but not limited to special, incidental, consequential, or other damages.

For general information on our other products and services or for technical support, please contact our Customer Care Department within the United States at (800) 762-2974, outside the United States at (317) 572-3993 or fax (317) 572-4002.

Wiley also publishes its books in variety of electronic formats. Some content that appears in print may not be available in electronic format. For more information about Wiley products, visit our web site at www.wiley.com.

Library of Congress Cataloging-in-Publication Data:

Statistical bioinformatics: a guide for life and biomedical science researchers / edited by Jae K. Lee.

p. cm.

Includes bibliographical references and index.

ISBN 978-0-471-69272-0 (cloth)

1. Bioinformatics—Statistical methods. I. Lee, Jae K.

QH324.2.S725 2010

570.285—dc22

2009024890

Printed in the United States of America

10 9 8 7 6 5 4 3 2 1

For Sue, Irene, and Kevin

CONTENTS

<i>PREFACE</i>	<i>xi</i>
<hr/>	
<i>CONTRIBUTORS</i>	<i>xiii</i>
<hr/>	
1 <i>ROAD TO STATISTICAL BIOINFORMATICS</i>	<i>1</i>
<hr/>	
Challenge 1: Multiple-Comparisons Issue	1
Challenge 2: High-Dimensional Biological Data	2
Challenge 3: Small- <i>n</i> and Large- <i>p</i> Problem	3
Challenge 4: Noisy High-Throughput Biological Data	3
Challenge 5: Integration of Multiple, Heterogeneous Biological Data Information	3
References	5
2 <i>PROBABILITY CONCEPTS AND DISTRIBUTIONS FOR ANALYZING LARGE BIOLOGICAL DATA</i>	<i>7</i>
<hr/>	
2.1 Introduction	7
2.2 Basic Concepts	8
2.3 Conditional Probability and Independence	10
2.4 Random Variables	13
2.5 Expected Value and Variance	15
2.6 Distributions of Random Variables	19
2.7 Joint and Marginal Distribution	39
2.8 Multivariate Distribution	42
2.9 Sampling Distribution	46
2.10 Summary	54
3 <i>QUALITY CONTROL OF HIGH-THROUGHPUT BIOLOGICAL DATA</i>	<i>57</i>
<hr/>	
3.1 Sources of Error in High-Throughput Biological Experiments	57
3.2 Statistical Techniques for Quality Control	59
3.3 Issues Specific to Microarray Gene Expression Experiments	66
3.4 Conclusion	69
References	69
4 <i>STATISTICAL TESTING AND SIGNIFICANCE FOR LARGE BIOLOGICAL DATA ANALYSIS</i>	<i>71</i>
<hr/>	
4.1 Introduction	71
4.2 Statistical Testing	72
4.3 Error Controlling	78

viii CONTENTS

4.4	Real Data Analysis	81
4.5	Concluding Remarks	87
	Acknowledgments	87
	References	88
5	<i>CLUSTERING: UNSUPERVISED LEARNING IN LARGE BIOLOGICAL DATA</i>	89
<hr/>		
5.1	Measures of Similarity	90
5.2	Clustering	99
5.3	Assessment of Cluster Quality	115
5.4	Conclusion	123
	References	123
6	<i>CLASSIFICATION: SUPERVISED LEARNING WITH HIGH-DIMENSIONAL BIOLOGICAL DATA</i>	129
<hr/>		
6.1	Introduction	129
6.2	Classification and Prediction Methods	132
6.3	Feature Selection and Ranking	140
6.4	Cross-Validation	144
6.5	Enhancement of Class Prediction by Ensemble Voting Methods	145
6.6	Comparison of Classification Methods Using High-Dimensional Data	147
6.7	Software Examples for Classification Methods	150
	References	154
7	<i>MULTIDIMENSIONAL ANALYSIS AND VISUALIZATION ON LARGE BIOMEDICAL DATA</i>	157
<hr/>		
7.1	Introduction	157
7.2	Classical Multidimensional Visualization Techniques	158
7.3	Two-Dimensional Projections	161
7.4	Issues and Challenges	165
7.5	Systematic Exploration of Low-Dimensional Projections	166
7.6	One-Dimensional Histogram Ordering	170
7.7	Two-Dimensional Scatterplot Ordering	174
7.8	Conclusion	181
	References	182
8	<i>STATISTICAL MODELS, INFERENCE, AND ALGORITHMS FOR LARGE BIOLOGICAL DATA ANALYSIS</i>	185
<hr/>		
8.1	Introduction	185
8.2	Statistical/Probabilistic Models	187
8.3	Estimation Methods	189
8.4	Numerical Algorithms	191
8.5	Examples	192
8.6	Conclusion	198
	References	199

9	<i>EXPERIMENTAL DESIGNS ON HIGH-THROUGHPUT BIOLOGICAL EXPERIMENTS</i>	201
<hr/>		
9.1	Randomization	201
9.2	Replication	202
9.3	Pooling	209
9.4	Blocking	210
9.5	Design for Classifications	214
9.6	Design for Time Course Experiments	215
9.7	Design for eQTL Studies	215
	References	216
10	<i>STATISTICAL RESAMPLING TECHNIQUES FOR LARGE BIOLOGICAL DATA ANALYSIS</i>	219
<hr/>		
10.1	Introduction	219
10.2	Resampling Methods for Prediction Error Assessment and Model Selection	221
10.3	Feature Selection	225
10.4	Resampling-Based Classification Algorithms	226
10.5	Practical Example: Lymphoma	226
10.6	Resampling Methods	227
10.7	Bootstrap Methods	232
10.8	Sample Size Issues	233
10.9	Loss Functions	235
10.10	Bootstrap Resampling for Quantifying Uncertainty	236
10.11	Markov Chain Monte Carlo Methods	238
10.12	Conclusions	240
	References	247
11	<i>STATISTICAL NETWORK ANALYSIS FOR BIOLOGICAL SYSTEMS AND PATHWAYS</i>	249
<hr/>		
11.1	Introduction	249
11.2	Boolean Network Modeling	250
11.3	Bayesian Belief Network	259
11.4	Modeling of Metabolic Networks	273
	References	279
12	<i>TRENDS AND STATISTICAL CHALLENGES IN GENOMEWIDE ASSOCIATION STUDIES</i>	283
<hr/>		
12.1	Introduction	283
12.2	Alleles, Linkage Disequilibrium, and Haplotype	283
12.3	International HapMap Project	285
12.4	Genotyping Platforms	286
12.5	Overview of Current GWAS Results	287
12.6	Statistical Issues in GWAS	290
12.7	Haplotype Analysis	296
12.8	Homozygosity and Admixture Mapping	298
12.9	Gene \times Gene and Gene \times Environment Interactions	298
12.10	Gene and Pathway-Based Analysis	299

X CONTENTS

12.11 Disease Risk Estimates	301
12.12 Meta-Analysis	301
12.13 Rare Variants and Sequence-Based Analysis	302
12.14 Conclusions	302
Acknowledgments	303
References	303
13 <i>R AND BIOCONDUCTOR PACKAGES IN BIOINFORMATICS: TOWARDS SYSTEMS BIOLOGY</i>	<i>309</i>
<hr/>	
13.1 Introduction	309
13.2 Brief overview of the Bioconductor Project	310
13.3 Experimental Data	311
13.4 Annotation	318
13.5 Models of Biological Systems	328
13.6 Conclusion	335
13.7 Acknowledgments	336
References	336
<i>INDEX</i>	<i>339</i>
<hr/>	

PREFACE

This book has been constructed primarily as a textbook for a one- or two-semester course in statistical bioinformatics. We hope that it will first serve as a comprehensive introduction to a broad range of topics in this area for life science students and researchers who are motivated to learn statistical analysis concepts and techniques in bioinformatics. Statistical and quantitative science audiences who have not yet dealt with challenging statistical issues in recent information-rich biological and biomedical data analysis may also benefit from this book by efficiently reviewing the different statistical concepts and techniques in such analyses. In particular, the four separate blocks in this book—statistical foundation, high-dimensional analysis, advanced topics, and multigene systems analysis—can be somewhat independently studied and taught in a course based on relevant needs and time restrictions for an effective learning purpose.

A similar outline as organized in this book has been used for several years in a one-semester graduate course which is open to students with diverse backgrounds at the University of Virginia. By publishing this book, we felt that these contents could be significantly enhanced by direct contributions of more specialized experts in the broad field of bioinformatics. In this multiauthor book, we have yet tried to maintain the need of a high-level mathematical and statistical understanding at a minimum. Readers of this book are thus assumed to have not much more than a basic college calculus level of mathematical understanding. A knowledge of matrix algebra would also be useful but is not a prerequisite.

This book is certainly a tribute to the contributions and support of many people. Acknowledgments are first due to the succeeding life science editors of John Wiley & Sons: Luna Han, Thomas Moore, and Karen Chambers. Without their continuous support and encouragement, the publication of this book would not have been possible. Also, all the efforts made by the expert authors who were willing to participate in this book should be highly acknowledged for its successful completion. Finally, the students who have taken this initial course and provided valuable feedback on various topics in this area over the last several years are also significant contributors to the current form of the book. The preparation of this book was supported in part by the National Institutes of Health Research Grant R01 HL081690.

J. K. LEE

*Division of Biostatistics and Epidemiology
Charlottesville, Virginia*

CONTRIBUTORS

Hongshik Ahn, Department of Applied Mathematics and Statistics, SUNY at Stony Brook, Stony Brook, NY, USA

Merav Bar, Fred Hutchinson Cancer Research Center, Program in Computational Biology, Division of Public Health Sciences, Seattle, WA, USA

Nabil Belacel, National Research Council Canada, Institute for Information Technology, Moncton, NB, Canada

Sooyoung Cheon, KU Industry-Academy Cooperation Group Team of Economics and Statistics, Korea University, Jochiwon 339-700, Korea

Hyung Jun Cho, Department of Statistics, Korea University, Seoul, Korea

Xiangqin Cui, Department of Biostatistics, University of Alabama, Birmingham, AL, USA

Miroslava Cupelovic-Culf, National Research Council Canada, Institute for Information Technology, Moncton, NB, Canada

Ginger Davis, Department of Systems and Information Engineering, University of Virginia, Charlottesville, VA, USA

Robert Gentleman, Fred Hutchinson Cancer Research Center, Program in Computational Biology, Division of Public Health Sciences, Seattle, WA, USA

Debashis Ghosh, Departments of Statistics and Public Health Sciences, Penn State University, University Park, PA, USA

Haseong Kim, Intelligent Systems and Networks Group, Department of Electrical and Electronic Engineering, Imperial College, London, UK

Youngeul Kim, Division of Biostatistics and Epidemiology, University of Virginia, Charlottesville, VA, USA

Michael Lawrence, Fred Hutchinson Cancer Research Center, Program in Computational Biology, Division of Public Health Sciences, Seattle, WA, USA

Jae K. Lee, Division of Biostatistics and Epidemiology, University of Virginia, Charlottesville, VA, USA

Seungyeoun Lee, Department of Applied Statistics, Sejong University, Seoul, Korea

Nolweim LeMeur, Fred Hutchinson Cancer Research Center, Program in Computational Biology, Division of Public Health Sciences, Seattle, WA, USA

Karen Lostritto, Yale School of Public Health, Yale University, New Haven, CT, USA

Annette M. Molinaro, Yale School of Public Health, Yale University, New Haven, CT, USA

Hojin Moon, Department of Mathematics and Statistics, California State University, Long Beach, CA, USA

Annamalai Muthiah, Department of Systems and Information Science, University of Virginia, Charlottesville, VA, USA

Taesung Park, Department of Statistics, Seoul National University, Seoul, Korea

Jinwook Seo, Visual Processing Laboratory, School of Computer Science and Engineering, Seoul National University, Gwanak-gu, Seoul, Korea

Wonseok Seo, Department of Statistics, Korea University, Seoul, Korea

Ben Sheiderman, Human-Computer Interaction Lab & Department of Computer Science, University of Maryland, College Park, MD, USA

Ning Sun, Department of Epidemiology and Public Health, Yale University School of Medicine, New Haven, CT, USA

Muneesh Tewari, Fred Hutchinson Cancer Research Center, Program in Computational Biology, Division of Public Health Sciences, Seattle, WA, USA

Christa Wang, National Research Council Canada, Institute for Information Technology, Moncton, NB, Canada

Paul D. Williams, Department of Public Health Science, University of Virginia, Charlottesville, VA, USA

Hongyu Zhao, Department of Epidemiology and Public Health, Yale University School of Medicine, New Haven, CT, USA

ROAD TO STATISTICAL BIOINFORMATICS

Jae K. Lee

*Department of Public Health Science, University of Virginia,
Charlottesville, Virginia, USA*

There has been a great explosion of biological data and information in recent years, largely due to the advances of various high-throughput biotechnologies such as mass spectrometry, high throughput sequencing, and many genome-wide SNP profiling, RNA gene expression microarray, protein mass spectrometry, and many other recent high-throughput biotechniques (Weinstein et al., 2002). Furthermore, powerful computing systems and fast Internet connections to large worldwide biological databases enable individual laboratory researchers to easily access an unprecedentedly huge amount of biological data. Such enormous data are often too overwhelming to understand and extract the most relevant information to each researcher's investigation goals. In fact, these large biological data are information rich and often contain much more information than the researchers who have generated such data may have anticipated. This is why many major biomedical research institutes have made significant efforts to freely share such data with general public researchers. Bioinformatics is the emerging science field concerned with the development of various analysis methods and tools for investigating such large biological data efficiently and rigorously. This kind of development requires many different components: powerful computer systems to archive and process such data, effective database designs to extract and integrate information from various heterogeneous biological databases, and efficient analysis techniques to investigate and analyze these large databases. In particular, analysis of these massive biological data is extremely challenging for the following reasons.

CHALLENGE 1: MULTIPLE-COMPARISONS ISSUE

Analysis techniques on high-throughput biological data are required to carefully handle and investigate an astronomical number of candidate targets and possible mechanisms, most of which are false positives, from such massive data (Tusher

et al., 2001). For example, a traditional statistical testing criterion which allows 5% false-positive error (or significance level) would identify ~ 500 false positives from 10K microarray data between two biological conditions of interest even though no real biologically differentially regulated genes exist between the two. If a small number of, for example, 100, genes that are actually differentially regulated exist, such real differential expression patterns will be mixed with the above 500 false positives without any a priori information to discriminate the true positives from the false positives. Then, confidence on the 600 targets that were identified by such a statistical testing may not be high. Simply tightening such a statistical criterion will result in a high false-negative error rate, without being able to identify many important real biological targets. This kind of pitfall, the so-called *multiple-comparisons issue*, becomes even more serious when biological mechanisms such as certain signal transduction and regulation pathways that involve multiple targets are searched from such biological data; the number of candidate pathway mechanisms to be searched grows exponentially, for example, $10!$ for 10-gene sequential pathway mechanisms. Thus, no matter how powerful a computer system can handle a given computational task, it is prohibitive to tackle such problems by exhaustive computational search and comparison for these kinds of problems. Many current biological problems have been theoretically proven to be NP (nonpolynomial) hard in computer science, implying that no finite (polynomial) computational algorithm can search all possible solutions as the number of biological targets involved in such a solution becomes too large. More importantly, this kind of exhaustive search is simply prone to the risk of discovering numerous false positives. In fact, this is one of the most difficult challenges in investigating current large biological databases and is why only heuristic algorithms that tightly control such a high false positive error rate and investigate a very small portion of all possible solutions are often sought for many biological problems. Thus, the success of many bioinformatics studies critically depends on the construction and use of effective and efficient heuristic algorithms, most of which are based on probabilistic modeling and statistical inference techniques that can maximize the statistical power of identifying true positives while rigorously controlling their false positive error rates.

CHALLENGE 2: HIGH-DIMENSIONAL BIOLOGICAL DATA

The second challenge is the high-dimensional nature of biological data in many bioinformatics studies. When biological data are simultaneously generated with many gene targets, their data points become dramatically sparse in the corresponding high-dimensional data space. It is well known that mathematical and computational approaches often fail to capture such high-dimensional phenomena accurately (Tamayo et al., 1999). For example, many statistical algorithms cannot easily move between local maxima in a high-dimensional space. Also, inference by combining several disjoint lower dimensional phenomena may not provide the correct understanding on the real phenomena in their joint, high-dimensional space. It is therefore important to understand statistical dimension reduction techniques that

can reduce high-dimensional data problems into lower dimensional ones while the important variation of interest in biological data is preserved.

CHALLENGE 3: SMALL- n AND LARGE- p PROBLEM

The third challenge is the so-called “small- n and large- p ” problem. Desired performance of conventional statistical methods is achieved when the sample size, namely n , of the data, the number of independent observations of event, is much larger than the number of parameters, say p , which need to be inferred by statistical inference (Jain et al., 2003). In many bioinformatics problems, this situation is often completely reversed. For example, in a microarray study, tens of thousands of gene transcripts’ expression patterns may become candidate prediction factors for a biological phenomenon of interest (e.g., tumor sensitivity vs. resistance to a chemotherapeutic compound) but the number of independent observations (e.g., different patient biopsy samples) is often at most a few tens or smaller. Due to the experimental costs and limited biological materials, the number of independent replicated samples can be sometimes extremely small, for example, two or three, or unavailable. In these cases, most traditional statistical approaches often perform very poorly. Thus, it is also important to select statistical analysis tools that can provide both high specificity and high sensitivity under these circumstances.

CHALLENGE 4: NOISY HIGH-THROUGHPUT BIOLOGICAL DATA

The fourth challenge is due to the fact that high-throughput biotechnical data and large biological databases are inevitably noisy because biological information and signals of interest are often observed with many other random or biased factors that may obscure main signals and information of interest (Cho and Lee, 2004). Therefore, investigations on large biological data cannot be successfully performed unless rigorous statistical algorithms are developed and effectively utilized to reduce and decompose various sources of error. Also, careful assessment and quality control of initial data sets is critical for all subsequent bioinformatics analyses.

CHALLENGE 5: INTEGRATION OF MULTIPLE, HETEROGENEOUS BIOLOGICAL DATA INFORMATION

The last challenge is the integration of information often from multiple heterogeneous biological and clinical data sets, such as large gene functional and annotation databases, biological subjects’ phenotypes, and patient clinical information. One of the main goals in performing high-throughput biological experiments is to identify the most important critical biological targets and mechanisms highly associated with biological subjects’ phenotypes, such as patients’ prognosis and therapeutic response

(Pittman et al., 2004). In these cases, multiple large heterogeneous datasets need to be combined in order to discover the most relevant molecular targets. This requires combining multiple datasets with very different data characteristics and formats, some of which cannot easily be integrated by standard statistical inference techniques, for example, the information from genomic and proteomic expression data and reported pathway mechanisms in the literature. It will be extremely important to develop and use efficient yet rigorous analysis tools for integrative inference on such complex biological data information beyond the individual researcher's manual and subjective integration.

In this book, we introduce the statistical concepts and techniques that can overcome these challenges in studying various large biological datasets. Researchers with biological or biomedical backgrounds may not be able, or may not need, to learn advanced mathematical and statistical techniques beyond the intuitive understanding of such topics for their practical applications. Thus, we have organized this book for life science researchers to efficiently learn the most relevant statistical concepts and techniques for their specific biological problems. We believe that this composition of the book will help nonstatistical researchers to minimize unnecessary efforts in learning statistical topics that are less relevant to their specific biological questions, yet help them learn and utilize rigorous statistical methods directly relevant to those problems. Thus, while this book can serve as a general reference for various concepts and methods in statistical bioinformatics, it is also designed to be effectively used as a textbook for a semester or shorter length course as below. In particular, the chapters are divided into four blocks of different statistical issues in analyzing large biological datasets (Fig. 1.1):

- I. *Statistical Foundation* Probability theories (Chapter 2), statistical quality control (Chapter 3), statistical tests (Chapter 4)

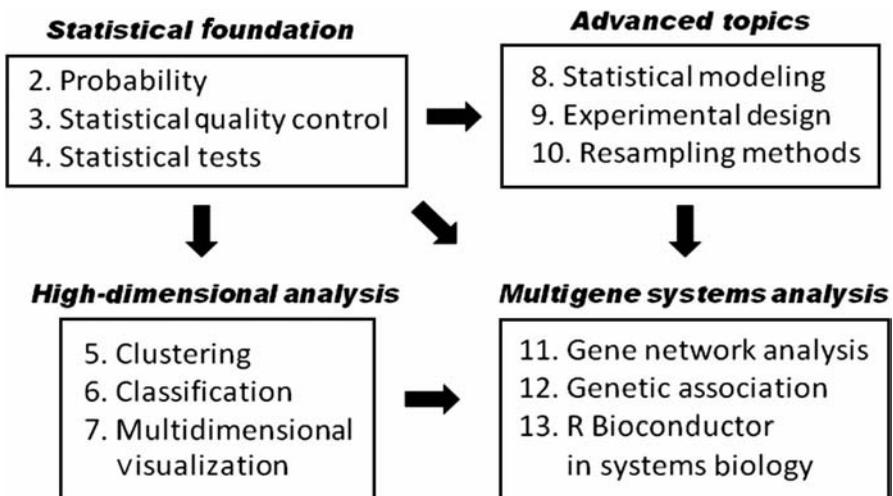


Figure 1.1 Possible course structure.

- II. *High-Dimensional Analysis* Clustering analysis (Chapter 5), classification analysis (Chapter 6), multidimensional visualization (Chapter 7)
- III. *Advanced Analysis Topics* Statistical modeling (Chapter 8), experimental design (Chapter 9), statistical resampling methods (Chapter 10)
- IV. *Multigene Analysis in Systems Biology* Genetic network analysis (Chapter 11), genetic association analysis (Chapter 12), R Bioconductor tools in systems biology (Chapter 13)

The first block of chapters will be important, especially for students who do not have a strong statistical background. These chapters will provide general backgrounds and terminologies to initiate rigorous statistical analysis on large biological datasets and to understand more advanced analysis topics later. Students with a good statistical understanding may also quickly review these chapters since there are certain key concepts and techniques (especially in Chapters 3 and 4) that are relatively new and specialized for analyzing large biological datasets.

The second block consists of analysis topics frequently used in investigating high-dimensional biological data. In particular, clustering and classification techniques, by far, are most commonly used in many practical applications of high-throughput data analysis. Various multidimensional visualization tools discussed in Chapter 7 will also be quite handy in such investigations.

The third block deals with more advanced topics in large biological data analysis, including advanced statistical modeling for complex biological problems, statistical resampling techniques that can be conveniently used with the combination of classification (Chapter 6) and statistical modeling (Chapter 8) techniques, and experimental design issues in high-throughput microarray studies.

The final block contains concise description of the analysis topics in several active research areas of multigene network and genetic association analysis as well as the R Bioconductor software in systems biology analysis. These will be quite useful for performing challenging gene network and multigene investigations in the fast-growing systems biology field.

These four blocks of chapters can be followed with the current order for a full semester-length course. However, except for the first block, the following three blocks are relatively independent of each other and can be covered (or skipped for specific needs and foci under a time constraint) in any order, as depicted in Figure 1.1. We hope that life science researchers who need to deal with challenging analysis issues in overwhelming large biological data in their specific investigations can effectively meet their learning goals in this way.

REFERENCES

- Cho, H., and Lee, J. K. (2004). Bayesian hierarchical error model for analysis of gene expression data. *Bioinformatics*, **20**(13): 2016–2025.
- Jain, N., et al. (2003). Local-pooled-error test for identifying differentially expressed genes with a small number of replicated microarrays. *Bioinformatics*, **19**(15): 1945–1951.

- Pittman, J., et al. (2004). Integrated modeling of clinical and gene expression information for personalized prediction of disease outcomes. *Proc. Natl. Acad. Sci. U.S.A.*, **101**(22): 8431–8436.
- Tamayo, P., et al. (1999). Interpreting patterns of gene expression with self-organizing maps: Methods and application to hematopoietic differentiation. *Proc. Natl. Acad. Sci. U.S.A.*, **96**(6): 2907–2912.
- Tusher, V. G., Tibshirani, R., and Chu, G. (2001). Significance analysis of microarrays applied to the ionizing radiation response. *Proc. Natl. Acad. Sci. U.S.A.*, **98**(9): 5116–5121.
- Weinstein, J. N., et al. (2002). The bioinformatics of microarray gene expression profiling. *Cytometry*, **47**(1): 46–49.

*PROBABILITY CONCEPTS
AND DISTRIBUTIONS
FOR ANALYZING LARGE
BIOLOGICAL DATA*

Sooyoung Cheon

*KU Industry-Academy Cooperation Group Team of Economics and Statistics,
Korea University, Jochiwon 339-700, Korea*

2.1 INTRODUCTION

In general, the results may vary when measurements are taken. Because of this variability, many decisions have to be made that involve uncertainties. For example, in medical research, interest may center on the effectiveness of a new vaccine for mumps or the estimation of the cure rate of treatment for breast cancer. In these situations, probabilistic foundation can help people make decisions to reduce uncertainty.

Experiments may be generally considered for events where the outcomes cannot be predicted with certainty; that is, each experiment ends in an outcome that cannot be determined with certainty before the performance of the experiment. Such experiments are called random experiments. Thus we consider probability in the study of randomness and uncertainty.

A physician tells a patient with breast cancer that there is a 90% cure rate with treatment. A sportscaster gives a baseball team a 20% chance of winning the next World Series. These two situations represent two different approaches, the frequentist and subjective approaches, respectively, to probability. The frequentist approach assigns a probability a long-term relative frequency. The cure rate is based on the observed outcomes of a large number of treated cases. When an experiment is repeated a large number of times under identical conditions, a regular pattern may emerge. In that case, the relative frequencies with which different outcomes occur will settle down to fixed proportions. These fixed proportions are defined as the probabilities of the corresponding outcomes. The subjective approach is a personal assessment of the chance that a given outcome will occur. The situations in this case are one-time phenomena that are not repeatable. The accuracy of this probability depends on the

information available about the particular situation and a person's interpretation of that information. A probability can be assigned, but it may vary from person to person.

This chapter reviews basic probability concepts and common distributions used for analyzing large biological data, in particular bioinformatics. First, basic concepts and some laws of rules of probability are introduced. In many cases, we are often in a situation to update our knowledge based on new information. In that case we need a concept of conditional probability. General definitions of conditional probability and the Bayes theorem are introduced. The theorem is the basis of statistical methodology called Bayesian statistics. Then random variables are discussed, along with the expected value and variance that are used to summarize its behavior. Six discrete and five continuous probability distributions are described that occur most frequently in bioinformatics and computational biology. We conclude with the description of an empirical distribution and sampling strategy related to resampling and bootstrapping techniques.

2.2 BASIC CONCEPTS

We begin with the notion of a random experiment, which is a procedure or an operation whose outcome is uncertain, and consider some aspects of events themselves before considering the probability theory associated with events.

2.2.1 Sample Spaces and Events

The collection of all possible outcomes of a particular experiment is called a *sample space*. We will use the letter S to denote a sample space. An *event* is any collection of possible outcomes of an experiment, that is, any subset of S . For example, if we plan to roll a die twice, the experiment is actual rolling of the die two times, and none, one, or both of these two events will have occurred after the experiment is carried out. In the first five examples, we have discrete sample spaces; in the last two, we have continuous sample spaces.

Example 2.1 Sample Space and Events from Random Experiments and Real Data. The following are some examples of random experiments and the associated sample spaces and events:

1. Assume you toss a coin once. The sample space is $S = \{H, T\}$, where H = head and T = tail and the event of a head is $\{H\}$.
2. Assume you toss a coin twice. The sample space is $S = \{(H, H), (H, T), (T, H), (T, T)\}$, and the event of obtaining exactly one head is $\{(H, T), (T, H)\}$.
3. Assume you roll a single die. The sample space is $S = \{1, 2, 3, 4, 5, 6\}$, and the event that the outcome is even is $\{2, 4, 6\}$.
4. Assume you roll two dice. The sample space is $S = \{(1, 1), (1, 2), \dots, (6, 6)\}$. The event that the sum of the numbers on the two dice equals 5 is $\{(1, 4), (2, 3), (3, 2), (4, 1)\}$.

5. Assume you count the number of defective welds in a car body. The sample space is $S = \{0, 1, \dots, N\}$, where $N =$ total number of welds. The event that the number of defective welds is no more than two is $\{0, 1, 2\}$.
6. Assume you measure the relative humidity of air. The sample space is $S = [0, 100]$. The event that the relative humidity is at least 90% is $[90, 100]$.
7. Assume you observe the lifetime of a car battery. The sample space is $S = [0, +\infty)$. The event that the car battery fails before 12 months is $[0, 12)$.

Event algebra is a mathematical language to express relationships among events and combinations of events. Below are the most common terms and ones that will be used in the remainder of this chapter.

Union The union of two events A and B , denoted by $A \cup B$, is the event consisting of all outcomes that belong to A or B or both:

$$A \cup B = \{x: x \in A \quad \text{or} \quad x \in B\}$$

Intersection The intersection of events A and B , denoted by $A \cap B$ or simply by AB , is the event consisting of all outcomes common to both A and B :

$$A \cap B = \{x: x \in A \quad \text{and} \quad x \in B\}$$

Complementation The complement of an event A , denoted by A^c , is the event consisting of all outcomes *not* in A .

$$A^c = \{x: x \notin A\}$$

Disjoint If A and B have no outcomes in common, that is, $A \cap B = \emptyset$, where \emptyset is an empty set, then A and B are called disjoint or mutually exclusive events.

Example 2.2 Event Algebra from Random Experiments

$$A = \{\text{Sum of two dice is a multiple of 3}\} = \{3, 6, 9, 12\}$$

$$B = \{\text{Sum of two dice is a multiple of 4}\} = \{4, 8, 12\}$$

$$C = \{\text{Sum of two dice is even}\} = \{2, 4, 6, 8, 10, 12\}$$

$$D = \{\text{Sum of two dice is odd}\} = \{3, 5, 7, 9, 11\}$$

1. The union of A and B is $A \cup B = \{3, 4, 6, 8, 9, 12\}$.
2. The intersection of A and B is $A \cap B = \{12\}$.
3. The complementation of A is $A^c = \{2, 4, 5, 7, 8, 10, 11\}$.
4. Since $C \cap D = \emptyset$, C and D are mutually exclusive events.

2.2.2 Probability

When an experiment is performed, an outcome from the sample space occurs. If the same experiment is performed a number of times, different outcomes may occur

each time or some outcomes may repeat. The relative frequency of occurrence of an outcome in a large number of identical experiments can be thought of as a probability. More probable outcomes are those that occur more frequently. If the outcomes of an experiment can be described probabilistically, the data from an experiment can be analyzed statistically.

The probability usually assigns a real number to each event. Let S be the sample space of an experiment. Each event A in S has a number, $P(A)$, called the *probability* of A , which satisfied the following three axioms:

Axiom 1 $P(A) \geq 0$.

Axiom 2 $P(S) = 1$.

Axiom 3 If A and B are mutually exclusive events, then $P(A \cup B) = P(A) + P(B)$.

These three axioms form the basis of all probability theory. Any function of P that satisfies the axioms of probability is called a probability function. For any sample space many different probability functions can be defined. The following basic results can be derived using event algebra:

- $P(A^c) = 1 - P(A)$.
- For any two events A and B , $P(A \cup B) = P(A) + P(B) - P(A \cap B)$.
- For any two events A and B , $P(A) = P(A \cap B) + P(A \cap B^c)$.
- If B is included in A , then $A \cap B = B$.
- Therefore $P(A) - P(B) = P(A \cap B^c)$ and $P(A) \geq P(B)$.

Example 2.3 Probabilities of Events from Random Experiments. The following are some examples of the probability of an outcome from experiments given in Example 2.1.

1. $P(\{H\}) = \frac{1}{2}$.
2. $P(\{(T, T)\}) = \frac{1}{4}$.
3. $P(\{1, 2\}) = \frac{2}{6} = \frac{1}{3}$.
4. $P(\{(1, 1), (2, 2), (3, 3), (4, 4), (5, 5), (6, 6)\}) = \frac{6}{36} = \frac{1}{6}$.

2.3 CONDITIONAL PROBABILITY AND INDEPENDENCE

A sample space is generally defined and all probabilities are calculated with respect to that sample space. In many cases, however, we are in a position to update the sample space based on new information. For example, like the fourth example of Example 2.3, if we just consider the case that two outcomes from rolling a die twice are the same, the size of the sample space is reduced from 36 to 6. General definitions of conditional probability and independence are introduced. The Bayes theorem is also introduced, which is the basis of a statistical methodology called Bayesian statistics.

2.3.1 Conditional Probability

Consider two events, A and B . Suppose that an event B has occurred. This occurrence may change the probability of A . We denote this by $P(A|B)$, the conditional probability of event A given that B has occurred.

As an example, suppose a card is randomly dealt from a well-shuffled deck of 52 cards. The probability of this card being a heart is $\frac{1}{4}$, since there are 13 hearts in a deck. However, if you are told that the dealt card is red, then the probability of a heart changes to $\frac{1}{2}$, because the sample space is reduced to just 26 cards. Likewise, if you are told that the dealt card is black, then $P(\text{Heart} | \text{Black}) = 0$.

If A and B are events in S and $P(B) > 0$, then $P(A|B)$ is called the *conditional probability of A given B* if the following axiom is satisfied:

$$\text{Axiom } P(A|B) = P(A \cap B)/P(B).$$

Example 2.4 Conditional Probability from Tossing Two Dice. An experiment consists of tossing two fair dice with a sample space of $6 \times 6 = 36$ outcomes. Consider two events:

$$A = \{\text{difference of the numbers on the dice is } 3\}$$

$$B = \{\text{sum of the numbers on the dice is a multiple of } 3\}$$

What is the conditional probability of A given B ?

Solution

$$A = \{(1, 4), (2, 5), (3, 6), (4, 1), (5, 2), (6, 3)\},$$

$$B = \{(1, 2), (1, 5), (2, 1), (2, 4), (3, 3), (3, 6), (4, 2), (4, 5), (5, 1), (5, 4), (6, 3), (6, 6)\}$$

and thus $A \cap B = \{(3, 6), (6, 3)\}$.

Thus B and $A \cap B$ consist of 12 and 2 outcomes, respectively. Assuming that all outcomes are equally likely, the conditional probability of A given B is

$$P(A|B) = \frac{P(A \cap B)}{P(B)} = \frac{2/36}{12/36} = \frac{2}{12} = \frac{1}{6}.$$

That means that the probability of the event with 2 outcomes that difference of numbers on two dice is 3, in the sample space of 12 outcomes that sum of numbers on two dice is a multiple of 3, is $1/6$.

2.3.2 Independence

Sometimes there are some situations that the knowledge of an event B does not give us any more information about A than what we already had. Like this situation the event A is called *independent* of the event B if $P(A|B) = P(A)$. In this case, we have the following axiom:

$$\text{Axiom } P(A \cap B) = P(A|B)P(B) = P(A)P(B).$$

Since $P(A \cap B)$ can also be expressed as $P(B | A) P(A)$, this shows that B is independent of A . Thus A and B are mutually independent.

Independent events are not same as disjoint events. In fact, there is a strong dependence between disjoint events. If $A \cap B = \emptyset$ and if A occurs, then B cannot occur. Similarly, if B occurs, then A cannot occur.

The advantage of the above axiom is that it treats the events symmetrically and will be easier to generalize to more than two events. Many gambling games provide models of independent events. The spins of a roulette wheel and the tosses of a pair of dice are both series of independent events.

Example 2.5 Independence from Tossing One Die. Suppose that a fair die is to be rolled. Let the event A be that the number is even and the event B that the number is greater than or equal to 3. Is event A independent of B ?

Here $A = \{2, 4, 6\}$ and $B = \{3, 4, 5, 6\}$. Thus $P(A) = \frac{1}{2}$ and $P(B) = \frac{2}{3}$ since S has six outcomes all equally likely. Given the event A , we know that it is 2, 4, or 6, and two of these three possibilities are greater than or equal to 3. Therefore, the probability the number is greater than or equal to 3 given that it is even is $\frac{2}{3}$:

$$P(B | A) = \frac{2}{3} = P(B)$$

These events are independent of each other. Thus the probability that it is greater than or equal to 3 is the same whether or not we know that the number is even.

2.3.3 Bayes's Theorem

For any two events A and B in a sample space, we can write

$$P(A \cap B) = P(A | B)P(B) = P(B | A)P(A)$$

This leads to a simple formula:

$$P(B | A) = \frac{P(A | B)P(B)}{P(A)}$$

This is known as *Bayes's theorem* or the inverse probability law. It forms the basis of a statistical methodology called Bayesian statistics. In Bayesian statistics, $P(B)$ is called the prior probability of B , which refers to the probability of B prior to the knowledge of the occurrence of A . We call $P(B | A)$ the posterior probability of B , which refers to the probability of B after observing A . Thus Bayes's theorem can be viewed as a way of updating the probability of B in light of the knowledge about A .

We can write the above formula in a more useful form.

Bayes's rule: Let B_1, B_2, \dots be a partition of the sample space. In other words, B_1, B_2, \dots are mutually disjoint and $B_1 \cup B_2 \cup \dots = S$. Let A be any set. Then, for each $i = 1, 2, \dots$,

$$P(B_i | A) = \frac{P(A | B_i)P(B_i)}{P(A)} = \frac{P(A | B_i)P(B_i)}{\sum_{j=1}^{\infty} P(A | B_j)P(B_j)}$$

This result can be used to update the prior probabilities of mutually exclusive events B_1, B_2, \dots in light of the new information that A has occurred. The following example illustrates an interesting application of Bayes's theorem.

Example 2.6 Bayes's Rule. A novel biomarker (or a combination of biomarkers) diagnosis assay is 95% effective in detecting a certain disease when it is present. The test also yields 1% false-positive result. If 0.5% of the population has the disease, what is the probability a person with a positive test result actually has the disease?

Solution Let $A = \{\text{a person's biomarker assay test result is positive}\}$ and $B = \{\text{a person has the disease}\}$. Then $P(B) = 0.005$, $P(A | B) = 0.95$, and $P(A | B^c) = 0.01$:

$$\begin{aligned} P(B | A) &= \frac{P(A | B)P(B)}{P(A | B)P(B) + P(A | B^c)P(B^c)} \\ &= \frac{0.95 \times 0.005}{0.95 \times 0.005 + 0.01 \times (1 - 0.005)} = \frac{475}{1470} \approx 0.323 \end{aligned}$$

Thus, the probability that a person really has the disease is only 32.3%!

In Example 2.6, how can we improve the odds of detecting real positives? We can improve by using multiple independent diagnosis assays, for example, A , B , and C , all with 32.3% detection probabilities. Then, the probability that a person with positive results from all three assays has the disease will be

$$\begin{aligned} P(A \cup B \cup C) &= 1 - P(A^c \cap B^c \cap C^c) \\ &= 1 - P(A^c) \times P(B^c) \times P(C^c) \\ &= 1 - (1 - 0.323)^3 = 1 - 0.677^3 \\ &\approx 68.97\% \end{aligned}$$

2.4 RANDOM VARIABLES

A random variable (abbreviated as r.v.) associates a unique numerical value with each outcome in the sample space. Formally, a r.v. is a real-valued function from a sample space S into the real numbers. We denote a r.v. by an uppercase letter (e.g., X or Y) and a particular value taken by a r.v. by the corresponding lowercase letter (e.g., x or y).

Example 2.7 Random Variables. Here are some examples of random variables:

1. Run an assay test. Then a r.v. is $X = \begin{cases} 1 & \text{if the result is positive} \\ 0 & \text{if the result is negative} \end{cases}$
2. Toss two dice. Then a r.v. is $X = \text{sum of the numbers on the dice.}$
3. Observe a transistor until it lasts. Then a r.v. is $X = \text{lifetime of the transistor in days.}$

A r.v. X may be discrete or continuous. The r.v.'s in the first two examples above are discrete, while the third one is continuous.

2.4.1 Discrete Random Variables

A r.v. is *discrete* if the number of possible values it can take is finite (e.g., $\{0, 1\}$) or countably infinite (e.g., all nonnegative integers $\{0, 1, 2, \dots\}$). Thus the possible values of a discrete r.v. can be listed as x_1, x_2, \dots . Suppose that we can calculate $P(X = x)$ for every value x . The collection of these probabilities can be viewed as a function of X . The *probability mass function (p.m.f.)* of a discrete r.v. X is given by

$$f_X(x) = P(X = x) \quad \text{for all } x$$

Example 2.8 Distribution of Discrete r.v.'s from Tossing Two Dice. Assume you toss two dice. What is the distribution of the sum?

Solution Let X be the sum of the numbers on two fair tossed dice. The p.m.f. of X can be derived by listing all 36 possible outcomes, which are equally likely, and counting the outcomes that result in $X = x$ for $x = 2, 3, \dots, 12$. Then

$$f(x) = P(X = x) = \frac{\# \text{ of outcomes with } X = x}{36}$$

TABLE 2.1 The p.m.f. of X , Sum of Numbers on Two Fair Dice

X	2	3	4	5	6	7	8	9	10	11	12
$f(x)$	$\frac{1}{36}$	$\frac{2}{36}$	$\frac{3}{36}$	$\frac{4}{36}$	$\frac{5}{36}$	$\frac{6}{36}$	$\frac{5}{36}$	$\frac{4}{36}$	$\frac{3}{36}$	$\frac{2}{36}$	$\frac{1}{36}$

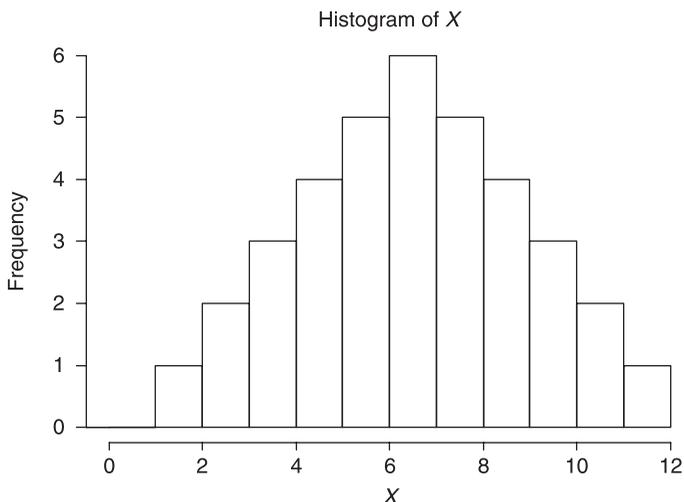


Figure 2.1 Histogram of X , the sum of the numbers on two fair dice.

For example, there are 4 outcomes that result in $X = 5$: (1, 4), (2, 3), (3, 2), (4, 1). Therefore,

$$f(5) = P(X = 5) = \frac{4}{36}$$

The p.m.f. is tabulated in Table 2.1 and plotted in Figure 2.1.

2.4.2 Continuous Random Variables

A r.v. X is *continuous* if it can take any value from one or more intervals of real numbers. We cannot use a p.m.f. to describe the probability distribution of X , because its possible values are uncountably infinite. We use a new notion called the *probability density function* (p.d.f.) such that areas under the $f(x)$ curve represent probabilities.

The p.d.f. of a continuous r.v. X is the function that satisfies

$$F_X(x) = \int_{-\infty}^x f_X(t) dt \quad \text{for all } x$$

where $F_X(x)$ is the *cumulative distribution function* (c.d.f.) of a r.v. X defined by

$$F_X(x) = P_X(X \leq x) \quad \text{for all } x$$

Example 2.9 Probability Calculation from Exponential Distribution. The simplest distribution used to model the times to failure (lifetime) of items or survival times of patients is the exponential distribution. The p.d.f. of the exponential distribution is given by

$$f(x) = \lambda e^{-\lambda x} \quad \text{for } x \geq 0$$

where λ is the failure rate. Suppose a certain type of computer chip has a failure rate of once every 15 years ($\lambda = \frac{1}{15}$), and the time to failure is exponentially distributed. What is the probability that a chip would last 5–10 years?

Solution Let X be the lifetime of a chip. The desired probability is

$$\begin{aligned} P(5 \leq x \leq 10) &= \int_5^{10} \lambda e^{-\lambda x} dx = [-e^{-\lambda x}]_5^{10} \\ &= e^{-5\lambda} - e^{-10\lambda} = 0.4866 - 0.2835 = 0.2031 \end{aligned}$$

A simple R code example is:

```
R command:  pexp(1/15, 10) - pexp(1/15, 5)
Output: 0.4866 - 0.2835 = 0.2031
```

2.5 EXPECTED VALUE AND VARIANCE

The p.m.f. or p.d.f. completely describes the probabilistic behavior of a r.v. However, certain numerical measures computed from the distribution provide useful summaries.

Two common measures that summarize the behavior of a r.v., called parameters, are its expected value (or mean) and its variance. The expected value of a r.v. is merely its average value weighted according to the probability distribution. The expected value of a distribution is a measure of center. By weighting the values of the r.v. according to the probability distribution, we are finding the number that is the average from numerous experiments. The variance gives a measure of the degree of spread of a distribution around its mean.

2.5.1 Expected Value

The *expected value* or the *mean* of a discrete r.v. X , denoted by $E(X)$, μ_X , or simply μ , is defined as

$$E(X) = \mu = \sum_x xf(x) = x_1f(x_1) + x_2f(x_2) + \cdots$$

This is a sum of possible values, x_1, x_2, \dots , taken by the r.v. X weighted by their probabilities.

The expected value of a continuous r.v. X is defined as

$$E(X) = \mu = \int_x xf(x) dx$$

$E(X)$ can be thought of as the center of gravity of the distribution of X .

If the probability distribution of a r.v. X is known, then the expected value of a function of X , say $g(X)$ [e.g., $g(X) = X^2$], equals

$$E[g(X)] = \begin{cases} \sum_x g(x)f(x) & \text{if } X \text{ is discrete} \\ \int_x g(x)f(x) & \text{if } X \text{ is continuous} \end{cases}$$

provided that the sum of the integral exists. If $E|g(X)| = \infty$, we say that $E[g(X)]$ does not exist.

Example 2.10 Expectation of Discrete Random Variable. Suppose two dice are tossed (Example 2.8). Let X_1 be the sum of two dice and X_2 and X_3 be the values of the first and second tossings, respectively. What are the expectations of $X_1, X_2 + X_3$?

Solution

$$E(X_1) = 2 \times \frac{1}{36} + 3 \times \frac{2}{36} + \cdots + 12 \times \frac{1}{36} = 7$$

$$E(X_2 + X_3) = E(X_2) + E(X_3) = 2 \times (1 + 2 + 3 + 4 + 5 + 6) \times \frac{1}{6} = 2 \times 21 \times \frac{1}{6} = 7$$

Example 2.11 Expectation of Continuous Random Variable. Let the density of X be $f(x) = \frac{1}{2}$ for $0 < x < 2$. What is the expectation of X ?

Solution

$$E(X) = \int_0^2 \frac{1}{2}x \, dx = \left[\frac{x^2}{4} \right]_0^2 = \frac{2^2}{4} = 1$$

REMARK Changing the order of summation (or subtraction) and expectation does not affect the result (invariant to summation/subtraction).

2.5.2 Variance and Standard Deviation

The *variance* of a r.v. X , denoted by $\text{Var}(X)$, σ_X^2 , or simply σ^2 , is defined as

$$\text{Var}(X) = E(X - \mu)^2$$

The variance is a square sum of the differences between each possible value and mean μ weighted by their probabilities and a measure of the dispersion of a r.v. about its mean.

The variance of a constant is zero.

An alternative expression for $\text{Var}(X)$ is given by

$$\begin{aligned} E(X - \mu)^2 &= E(X^2 - 2\mu X + \mu^2) = E(X^2) - 2\mu E(X) + \mu^2 \\ &= E(X^2) - 2\mu^2 + \mu^2 = E(X^2) - \mu^2 \end{aligned}$$

The *standard deviation* (*SD*) is the positive square root of the variance:

$$\text{SD}(X) = \sqrt{\text{Var}(X)}$$

The interpretation attached to the variance is that a small value of the variance means that all X s are very likely to be close to $E(X)$ and a larger value of variance means that all X s are more spread out. If $\text{Var}(X) = 0$, then X is equal to $E(X)$, with probability 1, and there is no variation in X . The standard deviation has the same qualitative interpretation. The standard deviation is easier to interpret in that the measurement unit on the standard deviation is the same as that for the original variable X . The measurement unit on the variance is the square of the original unit.

Example 2.12 Variance from Tossing Two Dice. Let X be the sum of the values from tossing two dice. What is the variance of X ?

Solution

$$\begin{aligned} \text{Var}(X) &= E(X^2) - \mu^2 = 2^2 \times \frac{1}{36} + 3^2 \times \frac{2}{36} + \dots + 12^2 \times \frac{1}{36} - 7^2 \\ &= 54.833 - 49 = 5.833 \end{aligned}$$

2.5.3 Covariance and Correlation

In earlier sections, we discussed the independence in a relationship between two r.v.'s. If there is a relationship, it may be strong or weak. In this section we discuss two numerical measures of the strength of a relationship between two r.v.'s, the covariance and correlation.

For instance, consider an experiment in which r.v.'s X and Y are measured, where X is the weight of a sample of water and Y is the volume of the same sample of water. Clearly there is a strong relationship between X and Y . If (X, Y) pairs with such a strong relationship are measured on a lot of samples and plotted, they should fall on a straight line. Now consider another experiment where X is the body weight of a human and Y is the same human's height. We can clearly find that they have a relationship, but it is not nearly as strong. In this case we would not expect that the observed data points fall on a straight line. These relationships are measured by covariance and correlation that quantify the strength of a relationship between two r.v.'s.

The *covariance* of two r.v.'s, X and Y , measures the joint dispersion from their respective means, given by

$$\begin{aligned}\text{Cov}(X, Y) &= E[(X - \mu_X)(Y - \mu_Y)] \\ &= E(XY) - \mu_X\mu_Y = E(XY) - E(X)E(Y)\end{aligned}$$

Note that $\text{Cov}(X, Y)$ can be positive or negative. Positive covariance implies that large (small) values of X are associated with large (small) values of Y ; negative covariance implies that large (small) values of X are associated with small (large) values of Y .

If X and Y are independent, then $E(XY) = E(X)E(Y)$ and hence $\text{Cov}(X, Y) = 0$. However, the converse of this is not true in general (e.g., normal). In other words, X and Y may be dependent and yet their covariance may be zero. This is illustrated by the following example.

Example 2.13 Example of Dependence with Zero Covariance. Define Y in terms of X such that

$$Y = \begin{cases} X & \text{with } P[Y] = \frac{1}{2} \\ -X & \text{with } P[Y] = \frac{1}{2} \end{cases}$$

Are X and Y independent?

Solution Obviously, Y depends on X ; however, $\text{Cov}(X, Y) = 0$, which can be verified as follows:

$$E(XY) = \frac{1}{2}E(X^2) - \frac{1}{2}E(X^2) = 0 \quad E(Y) = \frac{1}{2}E(X) - \frac{1}{2}E(X) = 0$$

Hence, $\text{Cov}(X, Y) = E(XY) - E(X)E(Y) = 0$.

Example 2.14 Covariance from Probability and Statistics Grades. The joint distribution of the probability and statistics grades, X and Y , is given later in Table 2.6 and their marginal distributions are given also later in Table 2.7. What is the covariance of X and Y ?

Solution

$$E(XY) = (4 \times 4 \times 0.160) + (4 \times 3 \times 0.110) + \cdots + (1 \times 1 \times 0.025) = 8.660$$

$$E(X) = (4 \times 0.315) + \cdots + (1 \times 0.075) = 2.915$$

$$E(Y) = (4 \times 0.230) + \cdots + (1 \times 0.045) = 2.850$$

Then,

$$\text{Cov}(X, Y) = E(XY) - E(X)E(Y) = 8.660 - 2.915 \times 2.850 = 0.352$$

To judge the extent of dependence or association between two r.v.'s, we need to standardize their covariance. The correlation (or correlation coefficient) is simply the covariance standardized so that its range is $[-1, 1]$. The *correlation* between X and Y is defined by

$$\rho_{XY} = \text{Corr}(X, Y) = \frac{\text{Cov}(X, Y)}{\sqrt{\text{Var}(X)\text{Var}(Y)}} = \frac{\sigma_{XY}}{\sigma_X\sigma_Y}$$

Note that ρ_{XY} is a unitless quantity, while $\text{Cov}(X, Y)$ is not. It follows from the covariance relationship that if X and Y are independent, then $\rho_{XY} = 0$; however, $\rho_{XY} = 0$ does not imply that they are independent.

Example 2.15 Correlation Coefficient from Probability and Statistics Grades. We saw that the covariance between the probability and statistics grades is 0.352 in Example 2.14. Although this tells us that two grades are positively associated, it does not tell us the strength of linear association, since the covariance is not a standardized measure. For this purpose, we calculate the correlation coefficient. Check that $\text{Var}(X) = 0.858$ and $\text{Var}(Y) = 0.678$. Then

$$\rho_{XY} = \frac{\text{Cov}(X, Y)}{\sqrt{\text{Var}(X)\text{Var}(Y)}} = \frac{0.352}{\sqrt{0.858 \times 0.678}} = 0.462$$

This correlation is not very close to 1, which implies that there is not a strong linear relationship between X and Y .

2.6 DISTRIBUTIONS OF RANDOM VARIABLES

In this section we describe the six discrete probability distributions and five continuous probability distributions that occur most frequently in bioinformatics and computational biology. These are called *univariate* models. In the last three sections, we discuss probability models that involve more than one random variable called *multivariate* models.

TABLE 2.2 R Function Names and Parameters for Standard Probability Distributions

Distribution	R name	Parameters
Beta	beta	shape 1, shape 2
Binomial	binom	size, prob
Cauchy	cauchy	location, scale
Chi squared	chisq	df
Exponential	exp	rate
<i>F</i>	f	df1, df2
Gamma	gamma	shape, rate
Geometric	geom	prob
Hypergeometric	hyper	m, n, k
Lognormal	lnorm	meanlog, sdlog
Logistic	logis	location, scale
Negative binomial	nbinom	size, prob
Normal	norm	mean, sd
Poisson	pois	lambda
<i>T</i>	t	df
Uniform	unif	min, max
Weibull	weibull	shape, scale
Wilcoxon	wilcox	m, n

Table 2.2 describes the R function names and parameters for a number of standard probability distributions for general use. The first letter of the function name indicates which of the probability functions it describes. For example:

- `dnorm` is the density function (p.d.f.) of the normal distribution
- `pnorm` is the cumulative distribution function (c.d.f.) of the normal distribution
- `qnorm` is the quantile function of the normal distribution

These functions can be used to replace statistical tables. For example, the 5% critical value for a (two-sided) *t* test on 11 degrees of freedom is given by `qt(0.975, 11)`, and the *P* value associated with a `Poisson(25)`-distributed count of 32 is given (by convention) by `1 - ppois(31, 25)`. The functions can be given vector arguments to calculate several *P* values or quantiles. R codes for generating random samples from each selected distribution and drawing graphs of each p.d.f. and c.d.f. are provided in this section.

2.6.1 Selected Discrete Distributions

In this section we describe the six important discrete probability distributions in bioinformatics and computational biology. The means and variances of the distributions discussed below are listed in Table 2.3.

TABLE 2.3 Means and Variances of Discrete Random Variables

Distribution	Mean	Variance
Bernoulli (p)	p	$p(1 - p)$
Binomial (n, p)	np	$np(1 - p)$
Uniform (a, b)	$a + (b - 1)/2$	$(b^2 - 1)/12$
Geometric (p)	$p/(1 - p)$	$p/(1 - p)^2$
Negative binomial (m, p)	$m(1 - p)/p$	$m(1 - p)/p^2$
Poisson (λ)	λ	λ

2.6.1.1 Bernoulli Distribution A r.v. that can take only two values, say 0 and 1, is called a Bernoulli r.v. The Bernoulli distribution is a useful model for dichotomous outcomes. An experiment with a dichotomous outcome is called a *Bernoulli trial*.

Suppose that an item drawn at random from a production process can be either defective or nondefective. Let p be the fraction of the defective items produced by the process. Then the probabilities of the possible outcomes for an item drawn randomly from this process are $P(\text{Defective}) = p$ and $P(\text{Nondefective}) = 1 - p$. A Bernoulli r.v. can be defined as $X = 1$ if the item is defective and 0 if nondefective with the following *Bernoulli distribution*:

$$f(x) = P(X = x) = \begin{cases} p & \text{if } x = 1 \\ 1 - p & \text{if } x = 0 \end{cases}$$

The mean and variance of a Bernoulli r.v. can be found from their respective definitions:

$$E(X) = 0 \times P(X = 0) + 1 \times P(X = 1) = 0(1 - p) + 1(p) = p$$

$$\text{Var}(X) = E(X^2) - (EX)^2 = [0^2(1 - p) + 1^2(p)] - p^2 = p - p^2 = p(1 - p)$$

Example 2.16 Bernoulli Trials. Many experiments can be modeled as a sequence of Bernoulli trials, the simplest being repeated tossing of a coin; p = probability of a head, $X = 1$ if the coin shows a head. Other examples include gambling games (e.g., in roulette let $X = 1$ if red occurs, so p = probability of red), election polls ($X = 1$ if candidate A gets a vote), and incidence of a disease (p = probability that a random person gets infected). Suppose $X \sim \text{Bernoulli}(0.7)$. $P(X = 1) = 0.7$. The R code to compute $P(X = 1)$ is as follows:

```
R command: dbern(1, 0.7)
```

```
Output: 0.7
```

2.6.1.2 Binomial Distribution Some experiments can be viewed as a sequence of independent and identically distributed (i.i.d.) Bernoulli trials, where each outcome is a “success” or a “failure.” The total number of successes from such an experiment is

often of interest rather than the individual outcomes. If there is a fixed number n of trials that are independent and each trial has the same probability p of success, then the sum of these i.i.d. Bernoulli r.v.'s is referred to as a binomial r.v. The number of heads in some fixed number of tosses of a coin is an example of a binomial r.v.

The general form of the p.m.f. of a binomial r.v. X with parameters n and p [denoted by $X \sim \text{Bin}(n, p)$] is derived as follows: The probability of obtaining x successes and $n - x$ failures in a particular way (e.g., the first x trials resulting in successes and the last $n - x$ trials resulting in failures) is $p^x(1 - p)^{n-x}$, because the trials are independent. There are a total of nCx ways of distributing x successes and failures among n trials. Therefore the *binomial distribution* is given by

$$f(x) = P(X = x) = \binom{n}{x} p^x (1 - p)^{n-x} \quad \text{for } x = 0, 1, \dots, n$$

Note that $f(0) = P(X = 0) = (1 - p)^n$ and $f(n) = P(X = n) = p^n$.

To find the mean and variance of $X \sim \text{Bin}(n, p)$, note that $X = Y_1 + Y_2 + \dots + Y_n$, where the Y_i 's are i.i.d. Bernoulli r.v.'s with parameter p . From this representation we obtain

$$E(X) = E[Y_1] + E[Y_2] + \dots + E[Y_n] = np$$

and

$$\text{Var}(X) = \text{Var}[Y_1] + \text{Var}[Y_2] + \dots + \text{Var}[Y_n] = np(1 - p)$$

We will generate first 10,000 samples from the binomial for $n = 10$ and $p = 0.5$, $\text{Bin}(10, 0.5)$ and then 10,000 samples from the binomial for $n = 20$ and $p = 0.25$, $\text{Bin}(20, 0.25)$, using the following R code for the binomial plot:

```
bin1=rbinom(10000,10,0.5); bin2=rbinom(10000,20,0.25)
win.metafile('binom.emf')
par(mfrow=c(1,2))
hist(bin1, nclass=12, right=FALSE,
     probability=TRUE, sub='(a)',
     main="", xlab='X', ylab='P(X)')
hist(bin2, nclass=12, right=FALSE,
     probability=TRUE, sub='(b)',
     main="", xlab='X', ylab='P(X)')
dev.off()
```

Figure 2.2 shows the probability distribution for two different binomials generated by the above setting. The first (a) is symmetric while the second (b) is not. The probabilities in (b) are nonzero up to and including $x = 14$, but the probabilities for values of x exceeding 12 are too small to be visible on the graph.

Example 2.17 Binomial Probability from Tooth Cavities. Suppose that the incidence of tooth cavities in a population of children is known to be 10%. A toothpaste manufacturer

tests a newly formulated toothpaste on a random sample of 20 children. The children are examined one year after using the new toothpaste. Only one child in the sample (i.e., 5% of the sample) is found to have new cavities. Based on this test, the manufacturer advertises that the new formula reduces the incidence of tooth cavities by half. Is this claim supported by the test results? Superficially the claim appears to be valid. However, due to the small sample size of 20, this result could occur simply by chance. How likely is it to find no more than one child with cavities if the new toothpaste has in fact no effect (i.e., the incidence rate remains at 10%)?

Solution Under the assumption of no effect, $X =$ number of children in a sample of 20 who develop cavities has the $\text{Bin}(n = 20, p = 0.1)$ distribution and this probability is

$$\begin{aligned} P(X \leq 1) &= P[X = 0] + P[X = 1] = (0.90)^{20} + \binom{20}{1}(0.10)(0.90)^{19} \\ &= 0.122 + 0.270 = 0.392 \end{aligned}$$

Thus, test results as good as or better than what were actually found would have occurred almost 40% of the time even if the toothpaste was ineffective. Therefore the manufacturer's claim is not conclusively supported by the data. The R code for this example is as follows:

```
R command: pbinom(1, 20, 0.1)
Output: 0.3917
```

2.6.1.3 Uniform Distribution The simplest discrete probability distribution is the uniform distribution. A r.v. X has the discrete uniform distribution if the possible values of X are $a, a + 1, \dots, a + b - 1$, for two integer constants a and b , and the probability that X takes any specified one of these b possible values is $1/b$.

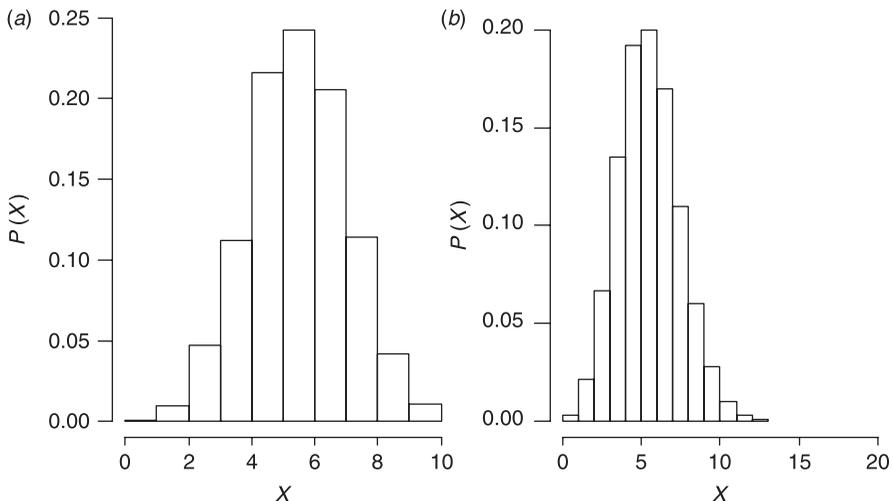


Figure 2.2 Two binomial distributions: (a) $\text{Bin}(10, 0.5)$; (b) $\text{Bin}(20, 0.25)$.

Therefore the *discrete uniform distribution* is given by

$$f(x) = P(X = x) = \frac{1}{b} \quad \text{for } x = a, a + 1, \dots, a + b - 1$$

We will generate 10,000,000 samples from Unif(5, 10) using the following R code:

```
unif=sample(5:10,size=10000000,replace=TRUE,
  prob=rep(1/6,6))
win.metafile('dunif.emf')
hist(unif,breaks=5:11,right=FALSE,probability=TRUE,
  main="",xlab='X',ylab='P(X)')
dev.off()
```

Figure 2.3 shows the probability distribution for uniform distribution with $a = 5$ and $b = 10$.

Example 2.18 Discrete Uniform Distribution from a Fair Die. A simple example of the discrete uniform distribution is throwing a fair die. The possible values of k are 1, 2, 3, 4, 5, 6; each time the die is thrown, the probability of a given score is $\frac{1}{6}$.

2.6.1.4 Geometric Distribution The geometric distribution models the number of i.i.d. Bernoulli trials needed to obtain the first success. It is the simplest of the waiting time distributions, that is, the distribution of discrete time to an event, and is a special case of the negative binomial distribution (we will show this distribution in the next section). Here the number of required trials is the r.v. of interest. As an

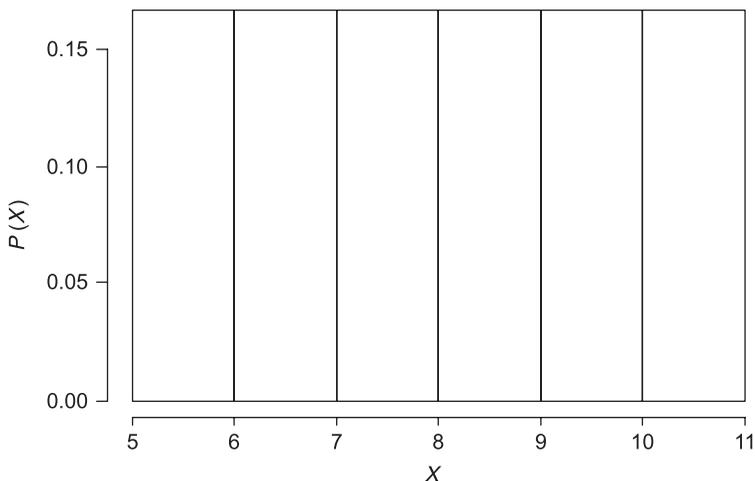


Figure 2.3 Density function of uniform distribution, Unif(5, 10).

example, consider playing on a slot machine until hitting the jackpot. Let the probability of hitting the jackpot on any attempt be p . The sample space is

$$\{S, FS, FFS, FFFS, FFFFS, \dots\}$$

where S denotes a “success” and F denotes a “failure.” Let X be the number of trials required to hit the first jackpot. Assuming that the attempts are independent and p remains constant, the probability of hitting the jackpot on the x th attempt is

$$\begin{aligned} f(x) &= P(X = x) = \underbrace{P(F) \times P(F) \times \dots \times P(F)}_{x-1 \text{ unsuccessful attempts}} \times P(S) \\ &= (1 - p)^{x-1} p \quad x = 1, 2, \dots \end{aligned}$$

This is a *geometric distribution* with parameter p . The geometric distribution with $p = 0.7$ is shown in Figure 2.4. We will generate 10,000 samples from $\text{Geo}(0.7)$ using the following R code (in R, $f(x) = p(1 - p)^x$, $x = 0, 1, 2, \dots$):

```
geo=rgeom(10000,0.7)
win.metafile('geo.emf')
hist(geo, nclass=6, right=FALSE, probability=TRUE, main="",
     xlab='X', ylab='P (X)')
dev.off()
```

The formulas for the mean and variance of a geometric r.v. are as follows: Let

$$Y = \begin{cases} 0 & \text{if the first attempt is a failure} \\ 1 & \text{if the first attempt is a success} \end{cases}$$

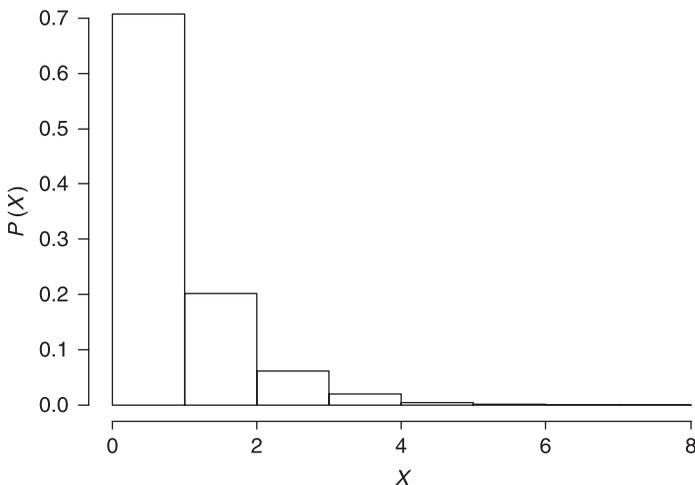


Figure 2.4 Geometric distribution with $p = 0.7$.

and X be the number of attempts required. The distribution of X is known as the geometric distribution:

$$\begin{aligned} E(X) &= E(X | Y = 0)P(Y = 0) + E(X | Y = 1)P(Y = 1) \\ &= E(X | Y = 0)(1 - p) + E(X | Y = 1)p \end{aligned}$$

Now, if the first attempt is a failure, then $E(X | Y = 0) = 1 + E(X)$ because one attempt is already made and the expected additional number of attempts is just $E(X)$, since future attempts are independent of the past and follow the same distribution. On the other hand, if the first attempt is a success, then $E(X | Y = 1) = 1$. Therefore,

$$\begin{aligned} E(X) &= [1 + E(X)](1 - p) + 1 \times p \\ &= 1/p \end{aligned}$$

and

$$E(X^2 | Y = 0) = E[(1 + X)^2] \quad \text{and} \quad E(X^2 | Y = 1) = 1$$

Hence,

$$\begin{aligned} E(X^2) &= E[(1 + X)^2](1 - p) + 1 \times p = [1 + 2E(X) + E(X^2)](1 - p) + 1 \times p \\ &= 1 - p + 2(1 - p) \times \frac{1}{p} + (1 - p)E(X^2) + p \\ E(X^2) &= \frac{2 - p}{p^2} \end{aligned}$$

Therefore,

$$\text{Var}(X) = E(X^2) - [E(X)]^2 = \frac{2 - p}{p^2} - \frac{1}{p^2} = \frac{1 - p}{p^2}$$

Memoryless Property A geometric r.v. has a property of forgetting the past in the following sense: Given that the first success has not occurred by the s th trial, the probability that it will not occur for at least another t trials is the same as if one were to start all over; that is, it is the probability that the first success will not occur for at least t trials starting at time zero. This property is easy to prove:

$$\begin{aligned} P(X > s + t | X > s) &= \frac{P[(X > s + t) \cap (X > s)]}{P[X > s]} = \frac{P[X > s + t]}{P[X > s]} \\ &= \frac{1 - P[X \leq s + t]}{1 - P[X \leq s]} = \frac{1 - \sum_{k=1}^{s+t} (1 - p)^{k-1} p}{1 - \sum_{k=1}^s (1 - p)^{k-1} p} \\ &= \frac{(1 - p)^{s+t}}{(1 - p)^s} = (1 - p)^t \end{aligned}$$

Example 2.19 Geometric Distribution from Fault Decision System. Suppose that a system is tested for a fault once every hour. The fault decision algorithm fails to detect an existing fault with probability 0.05. (a) What is the probability that a fault will be detected in exactly 3 hours? (b) In 3 hours or more? (c) In more than 3 hours, given that it has not been detected for 2 hours? (d) What is the average time to detection of a fault?

Solution Let X be the time to detection of a fault. Then X has a geometric distribution with $p = 1 - 0.05 = 0.95$.

(a) $P(X = 3) = (0.05)^2 (0.95) = 0.0024$.

(b) $P(X \geq 3) = P(X > 2) = 1 - P(X \leq 2) = 1 - 0.95 - (0.05)(0.95) = 0.0025$.

(c) $P(X > 3 | x > 2) = P(X > 1) = 0.05$.

(d) $E(X) = 1/0.95 = 1.053$ hours.

The R codes for this example are as follows:

```
R command: (a) dgeom(2, 0.95) (b) 1-pgeom(1, 0.95)
(c) 1-pgeom(0, 0.95)
Output: (a) 0.0024 (b) 0.0025 (c) 0.05
```

2.6.1.5 Negative Binomial Distribution The binomial distribution counts the number of successes in a fixed number of Bernoulli trials, each with a probability p of success. Instead, suppose that the number of successes m is fixed in advance, and the random variable X is the number of trials up to and including this m th success. The random variable X is then said to have the *negative binomial distribution*. The probability distribution of X is found as follows.

The $P(X = x)$ is the probability that the first $x - 1$ trials result in exactly $m - 1$ successes and $n - m$ failures (in some order) and that trial n results in success:

$$f(x) = P(X = x) = \binom{x-1}{m-1} p^m (1-p)^{x-m} \quad \text{for } x = m, m+1, m+2, \dots$$

An example for $p = 0.75$ and $m = 10$ is shown in Figure 2.5. We generate 10,000 samples from $NB(10, 0.75)$ using the following R code [in R, $f(x) =$

$$\binom{m+x-1}{x} p^m (1-p)^x, x = 0, 1, 2, \dots]:$$

```
nbin=rmnbinom(10000, 10, 0.75)
win.metafile('nbin.emf')
hist(nbin, nclass=12, right=FALSE, probability=TRUE, main=" ",
xlab='X', ylab='P (X) ')
dev.off()
```

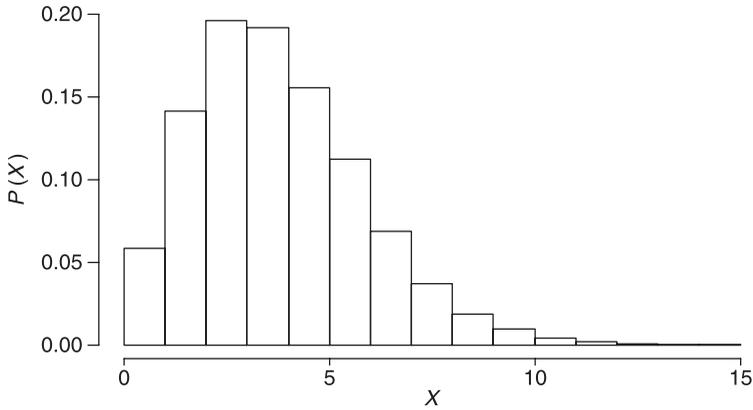


Figure 2.5 Negative binomial distribution with $p = 0.75$ and $m = 10$.

The mean and variance of X can be calculate as follows:

$$\begin{aligned}
 E(X) &= \sum_{x=m}^{\infty} x \binom{x-1}{m-1} p^m (1-p)^{x-m} = \sum_{x=0}^{\infty} x \binom{m+x-1}{x} p^m (1-p)^x \\
 &= \sum_{x=1}^{\infty} \frac{(m+x-1)!}{(x-1)!(m-1)!} p^m (1-p)^x = \sum_{x=1}^{\infty} m \binom{m+x-1}{x-1} p^m (1-p)^x \\
 &= \sum_{y=0}^{\infty} m \binom{m+y}{y} p^m (1-p)^{y+1} = m \frac{1-p}{p} \sum_{y=0}^{\infty} \binom{(m+1)+y-1}{y} p^{m+1} (1-p)^y \\
 &= \frac{m(1-p)}{p}
 \end{aligned}$$

A similar calculation will show

$$\text{Var}(X) = \frac{m(1-p)}{p^2}$$

Example 2.20 Fruit Flies; Negative Binomial Distribution. A technique known as inverse binomial sampling is useful in sampling biological populations. If the proportion of individuals possessing a certain characteristic is p and we sample until we see m such individuals, then the number of individuals sampled is a negative binomial random variable. Suppose that in a population of fruit flies we are interested in the proportion having vestigial wings and decide to sample until we have found 100 such flies. What is the probability that we will have to examine at least N flies?

Solution

$$P(X \geq N) = \sum_{x=N}^{\infty} \binom{x-1}{99} p^{100} (1-p)^{x-100} = 1 - \sum_{x=100}^{N-1} \binom{x-1}{99} p^{100} (1-p)^{x-100}$$

For given p and N , we can evaluate this expression to determine how many fruit flies we are likely to look at.

2.6.1.6 Poisson Distribution The *Poisson distribution* can serve as a model for a number of different types of experiments. For example, when the number of opportunities for the event is very large but the probability that the event occurs in any specific instance is very small, the number of occurrences of a rare event can sometimes be modeled by the Poisson distribution. Moreover, the occurrences are i.i.d. Bernoulli trials. Other examples are the number of earthquakes and the number of leukemia cases.

A r.v. X having this distribution is called a Poisson r.v. with parameter λ (denoted by $X \sim \text{Pois}(\lambda)$) given by

$$f(X) = P(X = x) = \frac{e^{-\lambda} \lambda^x}{x!} \quad x = 0, 1, 2, \dots$$

The Poisson distribution is a limiting form of the binomial distribution. When $n \rightarrow \infty$ and $p \rightarrow 0$ in such a way that np approaches a positive constant λ , the limiting binomial p.m.f. can be Poisson p.m.f. An example with $\lambda = 5$ is given in Figure 2.6. We

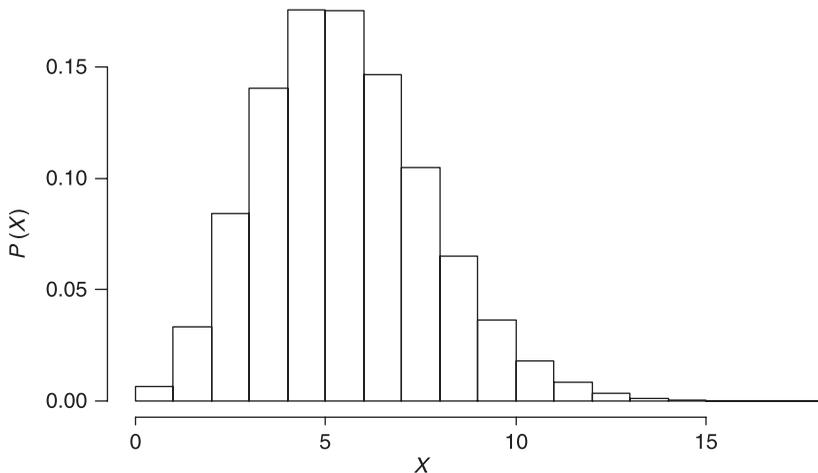


Figure 2.6 Poisson distribution with $\lambda = 5$.

generate 1,000,000 samples from POI(5) using the following R code:

```
poi=rpois(1000000,5)
win.metafile('poi.emf')
hist(poi, nclass=20, right=FALSE, probability=TRUE, main=" ",
     xlab='X', ylab='P (X) ')
dev.off()
```

The mean and variance of X are given by

$$E(X) = \text{Var}(X) = \lambda$$

Example 2.21 Poisson Approximation to Binomial from Tooth Cavities. In Example 2.17, let us see how well the Poisson distribution approximates the binomial distribution for the tooth cavity problem even though $n = 20$ is not very large and $p = 0.10$ is not very small.

Solution

$$\lambda = np = 20 \times 0.10 = 2.0$$

The Poisson approximation to $P(X = 0) + P(X = 1)$ is

$$\frac{e^{-2}2^0}{0!} + \frac{e^{-2}2^1}{1!} = 0.135 + 0.271 = 0.406$$

This is reasonably close to the binomial probability of 0.392 found in Example 2.17.

The R code for this example is as follows:

```
R command: ppois(1,2)
Output: 0.4060
```

2.6.2 Selected Continuous Distribution

Probabilities for continuous r.v.'s are not allocated to specific values but rather are allocated to intervals of values. Each r.v. X with range I has an associated density function $f_x(x)$, which is defined and positive for all x in the interval I , and the probability that the r.v. takes a value in some given interval is obtained by integrating the density function over that interval. Specifically,

$$P(a < X < b) = \int_a^b f_x(x) dx$$

This section discusses six important continuous distributions. The means and variances of the distributions discussed below are listed in Table 2.4.

2.6.2.1 Uniform Distribution A uniform distribution arises in situations where all values are “equally likely” over an interval. It is sometimes called a rectangular distribution. Specifically, the p.d.f. of a uniform distribution is constant over an interval.

TABLE 2.4 Means and Variances of Continuous Random Variables

Distribution	Mean	Variance
Uniform(a, b)	$(a + b)/2$	$(b - a)^2/12$
Normal(μ, σ^2)	μ	σ^2
Exponential(λ)	$1/\lambda$	$1/\lambda^2$
Gamma(α, β)	α/β	α/β^2
Beta(a, b)	$a/(a + b)$	$ab/[(a + b)^2(a + b + 1)]$

A r.v. X has a *uniform distribution* over the interval $[a, b]$ (denoted by $X \sim U[a, b]$) if the p.d.f. is given by

$$f(X) = \begin{cases} \frac{1}{b-a} & a \leq x \leq b \\ 0 & \text{otherwise} \end{cases}$$

The mean and variance of $X \sim U[a, b]$ are given by

$$E(X) = \frac{a+b}{2} \quad \text{and} \quad \text{Var}(X) = \frac{(b-a)^2}{12}$$

An example with $a = 2$ and $b = 5$ is given in Figure 2.7. The R code for the plot is as follows:

```
cunif=2:5
win.metafile('cunif.emf')
plot(cunif, dunif(cunif, 2, 5), type="l", xlab='X',
     ylab='f(x)', cex.lab=1.0)
dev.off()
```

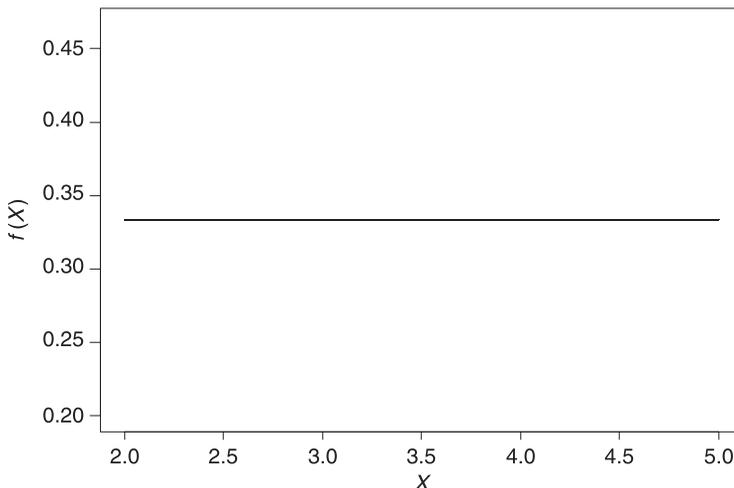


Figure 2.7 Density function of the uniform distribution with $U(2, 5)$.

Example 2.22 Uniform Distribution from a Fair Die. If a fair die is thrown, the probability of obtaining any one of the six possible outcomes is $\frac{1}{6}$. Since all outcomes are equally probable, the distribution is uniform. If a uniform distribution is divided into equally spaced intervals, there will be an equal number of members of the population in each interval.

2.6.2.2 Normal Distribution The most important continuous r.v. is one having the normal, or Gaussian, distribution. The normal distribution is used to model many real-life phenomena such as measurements of blood pressure and weight and dimensions. A large body of statistics is based on the assumption that the data follow the normal distribution.

A continuous r.v. X has a *normal distribution* with parameters μ and σ^2 [denoted by $X \sim N(\mu, \sigma^2)$], if its p.d.f. is given by

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x-\mu)^2/(2\sigma^2)} \quad -\infty < x < \infty$$

where $-\infty < \mu < \infty$ and $\sigma > 0$. The generated normal p.d.f. curve shown in Figure 2.8 is centered at $\mu = 1$, and $\sigma = 2$ is the distance from μ to the inflection point of the curve. The R code for the plot is as follows:

```
norm1=seq(-7,9,0.01)
win.metafile('norm.emf')
plot(norm1, dnorm(norm1, 1, 2), type="l", xlab='X',
      ylab='f (x)', cex.lab=1.2)
dev.off()
```

It can be shown that the mean and variance of $X \sim N(\mu, \sigma^2)$ are given by

$$E(X) = \mu \quad \text{and} \quad \text{Var}(X) = \sigma^2$$

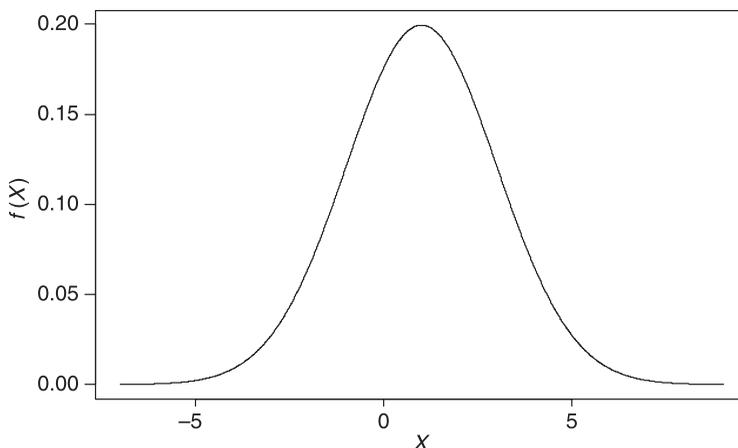


Figure 2.8 The p.d.f. of the normal distribution with $\mu = 1$ and $\sigma = 2$.

A particular important normal distribution is the one for which $\mu = 0$ and $\sigma^2 = 1$. This is sometimes called the *standard normal distribution*. Any normal r.v. can be transformed to a standard normal r.v. by subtracting its mean and then dividing by its standard deviation. This is called standardizing. If $X \sim N(\mu, \sigma^2)$, then

$$Z = \frac{X - \mu}{\sigma} \sim N(0, 1)$$

The observed value z of Z is often called a z score. This is a useful technique, since any probability involving $X \sim N(\mu, \sigma^2)$ can be expressed in terms of $Z \sim N(0, 1)$. We will use special notations for the p.d.f. and c.d.f. of Z since they occur frequently. The p.d.f. of Z is denoted by

$$\phi(z) = \frac{1}{\sqrt{2\pi}} e^{-z^2/2} \quad -\infty < z < \infty$$

and the c.d.f. of Z is denoted by

$$\Phi(z) = P(Z \leq z) = \int_{-\infty}^z \phi(y) dy$$

The standard normal p.d.f. curve is symmetric about zero. Since the total area under a p.d.f. curve is 1, the standard normal c.d.f. satisfies the condition

$$\Phi(-z) = 1 - \Phi(z)$$

For $z = 0$, we get $\Phi(0) = 1 - \Phi(0)$ or $\Phi(0) = \frac{1}{2}$. This result is, of course, obvious from the symmetry of this curve about zero.

Example 2.23 Cumulative Density Function of Normal Distribution. Suppose that the breaking strength (in pounds) of a rope has an $N(205, 5^2)$ distribution. A rope is considered defective if the breaking strength is less than 200 pounds. What proportions of ropes are defective?

Solution The proportion p is given by the probability that a randomly chosen rope has strength $X < 200$ pounds. Then

$$p = P(X < 200) = P\left(Z = \frac{X - 205}{5} < \frac{200 - 205}{5}\right) = \Phi(-1) = 0.1587$$

Thus nearly 16% of the ropes are defective. The R code for this example is as follows:

```
R command: pnorm(200, 205, 5)
Output: 0.1587
```

Example 2.24 Proportion Using Normal Distribution. When the data are normally distributed, σ is often used as a standard measure of distance from the mean μ of the distribution.

Calculate the proportion of the population that falls within one, two, and three standard deviations from μ .

Solution In general, when $X \sim N(\mu, \sigma^2)$ and a is a positive constant, the proportion of the population that falls within a standard deviations from μ is

$$P(\mu - a\sigma \leq X \leq \mu + a\sigma) = P\left(-a \leq Z = \frac{X - \mu}{\sigma} \leq a\right) = \Phi(a) - \Phi(-a) = 2\Phi(a) - 1$$

Using this relationship, we find

$$P(\mu - 1\sigma \leq X \leq \mu + 1\sigma) = 2\Phi(1) - 1 = 2(0.8413) - 1 = 0.6826$$

$$P(\mu - 2\sigma \leq X \leq \mu + 2\sigma) = 2\Phi(2) - 1 = 2(0.9772) - 1 = 0.9544$$

$$P(\mu - 3\sigma \leq X \leq \mu + 3\sigma) = 2\Phi(3) - 1 = 2(0.9987) - 1 = 0.9973$$

These calculations show that approximately 68% of a normal population lies within $\pm 1\sigma$ of μ , approximately 95% lies within $\pm 2\sigma$ of μ , and nearly 100% lies within $\pm 3\sigma$ of μ .

2.6.2.3 Exponential Distribution A third probability distribution arising often in computational biology is the *exponential distribution*. This distribution can be used to model lifetimes, analogous to the use of the geometric distribution in the discrete case; as such it is an example of a continuous waiting time distribution. A r.v. X with this distribution [denoted by $X \sim \exp(\lambda)$] has range $[0, +\infty]$ and density function

$$f(x) = \lambda e^{-\lambda} \quad x \geq 0$$

Here the single positive parameter λ characterizes the distribution. The R code for the plot is as follows:

```
expo=seq(0, 6, 0.01)
win.metafile('exp.emf')
plot(expo, dexp(expo, 2), type="l", xlab='X', ylab='f(x)',
      cex.lab=1.2)
dev.off()
```

The graph of the p.d.f. of the generated r.v. is shown in Figure 2.9.

The mean and variance of X are given by

$$E(X) = \frac{1}{\lambda} \quad \text{and} \quad \text{Var}(X) = \frac{1}{\lambda^2}$$

Since $1/\lambda$ is the average time between two events, λ can be interpreted as the rate at which events occur. It can be shown that if events occur according to a Poisson process with rate λ , then the interevent times are exponentially distributed with mean $1/\lambda$ and vice versa.

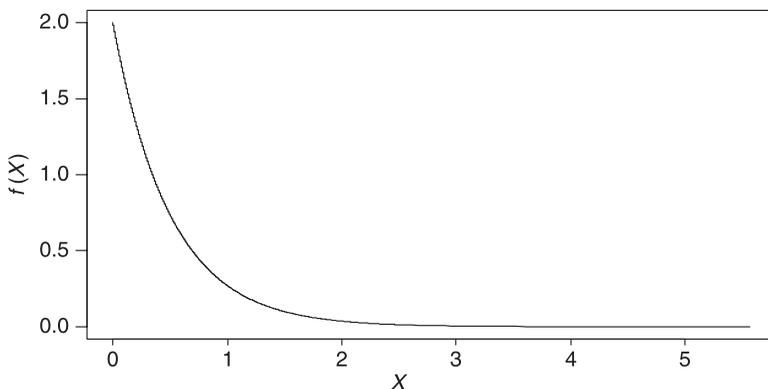


Figure 2.9 The p.d.f. of the exponential distribution with $f_X(x) = 2e^{-2x}$.

Memoryless Property The exponential distribution shares the memoryless property with the geometric distribution, and it is the only continuous distribution having this property. The proof of the property is simple:

$$\begin{aligned}
 P(X > s + t | X > s) &= \frac{P\{(X > s + t) \cap (X > s)\}}{P(X > s)} = \frac{P(X > s + t)}{P(X > s)} \\
 &= \frac{\int_{s+t}^{\infty} \lambda e^{-\lambda x} dx}{\int_s^{\infty} \lambda e^{-\lambda x} dx} = \frac{1 - \int_0^{s+t} \lambda e^{-\lambda x} dx}{1 - \int_0^s \lambda e^{-\lambda x} dx} \\
 &= \frac{1 - [1 - e^{-\lambda(s+t)}]}{1 - [1 - e^{-\lambda s}]} = e^{-\lambda t}
 \end{aligned}$$

Example 2.25 Exponential Distribution from Arrival Time of Shuttle Bus. Suppose that airport shuttle buses arrive at a terminal at the rate of one every 10 minutes with exponentially distributed interarrival times. If a person arrives at the bus stop and sees a bus leaving, what is the probability that he must wait for more than 10 minutes for the next bus? What if the person does not see a bus leaving?

Solution Let X be the time between arrivals of buses; $X \sim \exp(\lambda = \frac{1}{10})$. The first probability is

$$P(X > 10) = e^{-10/10} = 0.3679$$

One might think that the second probability should be smaller, since the person arrived after the previous bus had left and so there is a smaller chance that he will have to wait for more than 10 minutes. However, because of memoryless property of the exponential distribution

this probability is also 0.368. The R code for this example is as follows:

```
R command: 1-pexp(1/10,10)
Output: 0.3679
```

2.6.2.4 Gamma Distribution A r.v. X is said to have a *gamma distribution* with parameter $\alpha > 0$ and $\beta > 0$ [denoted by $X \sim \text{Gamma}(\alpha, \beta)$] if its p.d.f. is given by

$$f(x) = \frac{1}{\Gamma(\alpha)\beta^\alpha} x^{\alpha-1} e^{-\beta x} \quad x > 0$$

where the gamma function, $\Gamma(\alpha)$, is defined by

$$\Gamma(\alpha) = \int_0^{\infty} u^{\alpha-1} e^{-u} du \quad \alpha > 0$$

For positive integer values of α , it can be shown that $\Gamma(\alpha) = (\alpha - 1)!$. The parameter α is known as the shape parameter, since it most influences the peakedness of the distribution, while the parameter β is called the scale parameter, since most of its influence is on the spread of the distribution.

The density function can take several different shapes, depending on the value of the parameter α (see Fig. 2.10). The R code for the plot is as follows:

```
gam=seq(0.05,9,0.01); gam1=seq(0.01,9,0.01)
win.metafile('gam.emf')
```

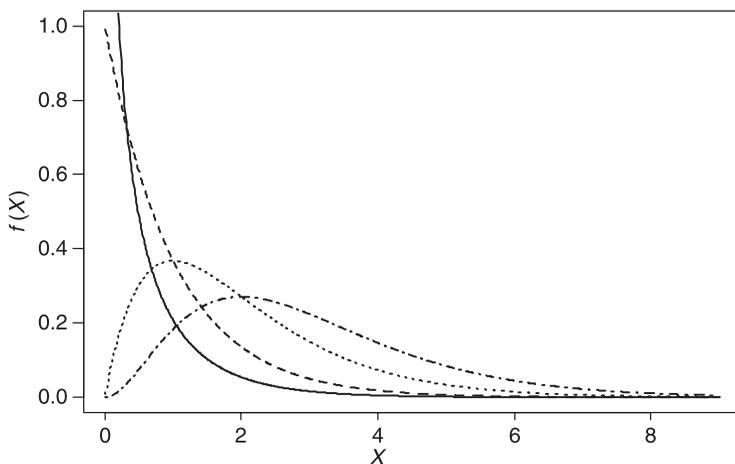


Figure 2.10 Examples of the density function of the gamma distribution with $\beta=1$, $\alpha = \frac{1}{2}$ (—), 1 (---), 2 (⋯), and 3 (-⋯).

```
plot(gam, dgamma(gam, 0.5, 1), ylim=c(0,1), type="l", lty=1,
      xlab='X', ylab='f(x)', cex.lab=1.2, lwd=2)
lines(gam1, dgamma(gam1, 1.0, 1), type='l', lty=2, lwd=2)
lines(gam1, dgamma(gam1, 2.0, 1), type='l', lty=3, lwd=2)
lines(gam1, dgamma(gam1, 3.0, 1), type='l', lty=4, lwd=2)
dev.off()
```

The mean and variance of X are given by

$$E(X) = \frac{\alpha}{\beta} \quad \text{and} \quad \text{Var}(X) = \frac{\alpha}{\beta^2}$$

The exponential distribution is a special case of the gamma distribution with $\alpha = 1$. The sum of n i.i.d. exponential r.v.'s, each with parameter β , can be shown to have the Gamma(β , n) distribution. As an illustration, consider a project that requires completing n independent successive tasks; each task has a processing time that is exponentially distributed with the same parameter β . Then the total processing time has the Gamma(β , n) distribution. Another important special case is where $\beta = \frac{1}{2}$ and $\alpha = \frac{1}{2}\nu$, with ν a positive integer, so that

$$f(x) = \frac{1}{2^{\nu/2}\Gamma[(1/2)\nu]} x^{(1/2)\nu-1} e^{-(1/2)x} \quad x > 0$$

This is called the chi-square distribution with ν degrees of freedom and is important for statistical hypothesis testing. It can be shown that if Z is a standard normal random variable, then Z^2 is a random variable having the chi-square distribution with one degree of freedom. Further, the exponential distribution with $\beta = \frac{1}{2}$ is the chi-square distribution with two degrees of freedom.

Example 2.26 Gamma Distribution from Arrival Time of Customer. Suppose that customers arrive at a bank teller's counter at the rate of 1 every 2 minutes on the average and their interarrival times are exponential distributed. What is the probability that at least five customers will arrive in 10 minutes?

Solution Let T_i denote the interarrival time between customer $i - 1$ and customer i ($i = 1, 2, 3, 4, 5$). Then $X = T_1 + \cdots + T_5$ has a gamma distribution with $n = 5$ and $\beta = \frac{1}{2}$. Thus $P(X \leq 10) = 0.5560$. The R code for this example is as follows:

```
R command: pgamma(10, 5, 1/2)
Output: 0.5595067
```

2.6.2.5 Beta Distribution The beta distribution provides a flexible way to model many types of measurements that have finite ranges and is one of the few common

“named” distributions that give probability 1 to a finite interval, taken to be $(0, 1)$. The beta distribution is often used to model proportions, which naturally lie between 0 and 1. A r.v. X has a *beta distribution* on the interval $[0, 1]$ with parameters a and b [denoted by $X \sim \text{Beta}(a, b)$] if its p.d.f. is given by

$$f(x) = \frac{1}{B(a, b)} x^{a-1} (1-x)^{b-1} \quad 0 \leq x \leq 1$$

where $B(a, b)$ is the beta distribution given as

$$B(a, b) = \frac{\Gamma(a)\Gamma(b)}{\Gamma(a+b)}$$

Note that a r.v. having a finite range other than $[0, 1]$ can always be transformed to the $[0, 1]$ range.

The $U[0, 1]$ distribution is a special case of the beta distribution when $a = b = 1$, in which case the above p.d.f. reduces to $f(x) = 1/B(1, 1) = 1$. The density function can take several different shapes, depending on the value of the parameter a and b (see Fig. 2.11). The R code for the plot is as follows:

```
bt=seq(0.01,0.99,0.01)
win.metafile('beta.emf')
plot(bt, dbeta(bt, 1, 1), ylim=c(0,10), type="l", lty=1,
      xlab='X', ylab='f (x)', cex.lab=1.2, lwd=2)
lines(bt,dbeta(bt, 1, 10),type='l', lty=2, lwd=2)
lines(bt,dbeta(bt, 10, 1),type='l', lty=3, lwd=2)
lines(bt,dbeta(bt, 10, 10),type='l', lty=4, lwd=2)
dev.off()
```

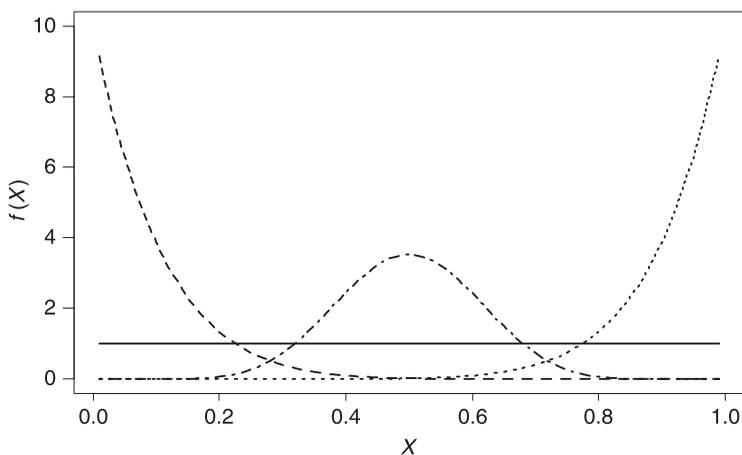


Figure 2.11 Examples of the density function of the beta distribution with $(a, b) = \{-(1, 1), \text{---}:(1, 10), \dots:(10, 1), \text{-}\cdot\text{-}\cdot:(10, 10)\}$.

The mean and variance of X are given by

$$E(X) = \frac{a}{a+b} \quad \text{and} \quad \text{Var}(X) = \frac{ab}{(a+b)^2(a+b+1)}$$

Example 2.27 Application of Beta Distribution. Beta distributions are used extensively in Bayesian statistics, since beta distributions provide a family of conjugate prior distributions for binomial (including Bernoulli) and geometric distributions. The Beta(0, 0) distribution is an improper prior and is sometimes used to represent ignorance of parameter values. This distribution can be used to model events which are constrained to take place within an interval defined by a minimum and maximum value. For this reason, the beta distribution—along with the triangular distribution—is used extensively in PERT, critical path method (CPM), and other project management control systems to describe the time to completion of a task. In project management, shorthand computations are widely used to estimate the mean and standard deviation of the beta distribution (Table 2.4).

2.7 JOINT AND MARGINAL DISTRIBUTION

In an experimental situation, it would be very unusual to observe only the value of one random variable, that is, the experiment that the total collected data consisted of just one numeric value. For example, consider an experiment designed to gain information about some health characteristics of a population of people. If we collect only one datum, say the height of one person, it would be a very modest experiment. Instead, the heights of several people in the population might be measured. These different heights would be observations on different random variables, one for each person measured. If several physical characteristics were measured on each person such as temperature, height, and blood pressure, multiple observations could also be obtained. These observations on different characteristics could also be modeled as observations on different random variables. Thus, we need to know how to describe and use probability models that deal with more than one random variable at a time. In this section, we will discuss joint and marginal distributions.

2.7.1 Jointly Distributed Random Variables

Frequently, more than one random variable is associated with an outcome. For example, an insurance file provides information on a customer's auto and homeowner policy. Vital signs monitored on a hospital patient include the systolic and diastolic blood pressure. The relationship between several variables can be investigated through their joint probability distribution.

To illustrate jointly distributed r.v.'s, we consider data on grades ($A = 4$, $B = 3$, $C = 2$, $D = 1$) in a probability and a statistics course for a sample of 200 Northwestern University undergraduate engineering students. A tabulation of these grades is shown in Table 2.5. Suppose the transcript of one of these students is drawn at random. What is the probability that the student received an A in probability and a B in statistics?

TABLE 2.5 Probability and Statistics Grades of 200 Students

		Statistics grade, Y				N_1
		A = 4	B = 3	C = 2	D = 1	
Probability grade, X	A = 4	32	22	8	1	63
	B = 3	11	35	25	1	72
	C = 2	3	25	20	2	50
	D = 1	0	5	5	5	15
	N_2	46	87	58	9	200

TABLE 2.6 Joint Distribution of Probability and Statistics Grades

		Statistics grade, Y				N_1
		A = 4	B = 3	C = 2	D = 1	
Probability grade, X	A = 4	0.160	0.110	0.040	0.005	0.315
	B = 3	0.055	0.175	0.125	0.005	0.360
	C = 2	0.015	0.125	0.100	0.010	0.250
	D = 1	0.000	0.025	0.025	0.025	0.075
	N_2	0.230	0.435	0.290	0.045	1.000

This probability is simply $22/200$, the proportion of students with this particular combination of grades. Thus $P(X = 4, Y = 3) = 0.110$. The joint p.m.f. of the discrete r.v.'s X and Y is given by Table 2.6, which shows the proportion of students with each combination of probability and statistics grades.

2.7.2 Joint Probability Mass or Density Function

If X and Y are discrete bivariate random vectors, their joint p.m.f. is $f(x, y) = P(X = x, Y = y)$ for all x and y . The joint p.m.f. satisfies

1. $f(x, y) \geq 0$ for all x and y .
2. $\sum_x \sum_y f(x, y) = 1$.

Similarly, the joint p.d.f. of two continuous bivariate random vectors (X, Y) is denoted by $f(X, Y)$, which satisfies

1. $f(x, y) \geq 0$ for all x and y .
2. $\int_x \int_y f(x, y) dx dy = 1$.

The joint p.m.f. of (X, Y) completely defines the probability distribution of the random vector (X, Y) , just as the p.m.f. of a discrete univariate random variable completely defines its distribution. Expectations of functions of random vectors are computed just as with univariate random variables. Let $g(x, y)$ be a real-valued function defined for all possible values (x, y) of the discrete random vector (X, Y) . Then $g(X, Y)$ is itself a random variable and its expected value $Eg(X, Y)$ is given by

$$Eg(X, Y) = \sum_{(x,y) \in \mathfrak{R}^2} g(x, y)f(x, y)$$

Example 2.28 Joint p.m.f. for Dice. Define $f(x, y)$ by

$$\begin{aligned} f(0, 0) &= f(0, 1) = \frac{1}{6} \\ f(1, 0) &= f(1, 1) = \frac{1}{3} \\ f(x, y) &= 0 \quad \text{for any other } (x, y) \end{aligned}$$

Then $f(x, y)$ is nonnegative and sums to 1, so $f(x, y)$ is the joint p.m.f. for some bivariate random vector (X, Y) . We can use $f(x, y)$ to compute probabilities such as

$$P(X = Y) = f(0, 0) + f(1, 1) = \frac{1}{2}$$

2.7.3 Marginal Distribution

Not only does the joint p.m.f. (or p.d.f.) give information about the behavior of a random vector (X, Y) together, it also contains information about the separate behaviors of the individual r.v.'s, X or Y . We now call $f_X(x)$ the *marginal p.m.f.* (or *p.d.f.*) of X to emphasize the fact that it is the p.m.f. (or p.d.f.) of X but in the context of the probability model that gives the joint distribution of the vector (X, Y) .

In the grades example, by computing the row and column totals of the joint probabilities from Table 2.6, we can find the p.m.f. of X and Y . The row totals give the p.m.f. of X and the column totals give the p.m.f. of Y . The resulting marginal distributions are shown in Table 2.7.

Given the joint p.m.f. of two discrete r.v.'s X and Y , the marginal p.m.f. of one r.v. can be found by summing over the values of the other:

$$g(x) = P(X = x) = \sum_y f(x, y) \quad \text{and} \quad h(y) = P(Y = y) = \sum_x f(x, y)$$

Similarly, given the joint p.d.f. of two continuous r.v.'s X and Y , the marginal p.d.f. of one r.v. can be found by integrating over the values of the other:

$$g(x) = P(X = x) = \int_{-\infty}^{\infty} f(x, y) dy \quad \text{and} \quad h(y) = P(Y = y) = \int_{-\infty}^{\infty} f(x, y) dx$$

TABLE 2.7 Marginal Distributions of Probability and Statistics Grades

<i>Marginal Distribution of X (Probability Grade)</i>				
x	1	2	3	4
$g(x) = P(X = x)$	0.075	0.250	0.360	0.315
<i>Marginal Distribution of Y (Statistics Grade)</i>				
y	1	2	3	4
$g(y) = P(Y = y)$	0.045	0.290	0.435	0.230

2.8 MULTIVARIATE DISTRIBUTION

In the previous sections our discussions has concentrated on just two r.v.'s. We refer to their joint distribution as a bivariate distribution. When more than two r.v.'s, say X_1, X_2, \dots, X_k , are jointly distributed, we can similarly define their joint p.m.f. (or p.d.f.), denoted by $f(x_1, x_2, \dots, x_n)$, referred to as a *multivariate distribution*. The properties of a multivariate distribution are natural extensions of the bivariate distribution properties. Examples of a multivariate distribution are multinomial, multivariate, normal, and Dirichlet distributions.

If $\mathbf{X} = (X_1, \dots, X_n)$ is a discrete random vector, then the joint p.m.f. of \mathbf{X} is the function defined by

$$f_{\mathbf{X}}(x) = f_{\mathbf{X}}(x_1, x_2, \dots, x_n) = P(X_1 = x_1, \dots, X_n = x_n) \quad \text{for each } (x_1, x_2, \dots, x_n)$$

Then for any the set A ,

$$P(\mathbf{X} \in A) = \sum_{x \in A} f(\mathbf{X})$$

If \mathbf{X} is a continuous random vector, then the joint p.d.f. of \mathbf{X} is a function $f_{\mathbf{X}}(x)$ satisfying $P(\mathbf{X} \in A) = \int \cdots \int_A f(\mathbf{X}) d\mathbf{x} = \int \cdots \int_A f(x_1, \dots, x_n) dx_1 \cdots dx_n$.

Let $g_{\mathbf{X}}(x) = g_{\mathbf{X}}(x_1, x_2, \dots, x_n)$ be a real-valued function defined on the sample space of \mathbf{X} . Then $g(\mathbf{X})$ is a random variable and the expected value of $g(\mathbf{X})$ is

$$Eg(\mathbf{X}) = \sum_{\mathbf{X}} g(\mathbf{X})f(\mathbf{X}) \quad \text{and} \quad Eg(\mathbf{X}) = \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} g(\mathbf{X})f(\mathbf{X}) d\mathbf{x}$$

in the discrete and continuous cases, respectively.

The marginal p.m.f. or p.d.f. can also be computed by summing or integrating the joint p.m.f. or p.d.f. over all possible values of the other coordinates. For example, the marginal distribution of X_1 is given by

$$f_1(x_1) = \begin{cases} \sum_{x_2} \cdots \sum_{x_k} f(x_1, x_2, \dots, x_k) & \text{in discrete case} \\ \int \int f(x_1, x_2, \dots, x_k) dx_2 \cdots dx_k & \text{in continuous case} \end{cases}$$

The conditional p.m.f. or p.d.f. given the values of the remaining coordinates is obtained by dividing the joint p.m.f. or p.d.f. by the marginal p.m.f. or p.d.f. of the remaining coordinates. For example, if $f(x_1, x_2, \dots, x_k) > 0$, the conditional p.m.f. or p.d.f. of (X_{k+1}, \dots, X_n) given $X_1 = x_1, \dots, X_n = x_n$ is the function of (x_1, \dots, x_n) defined by

$$f(x_{k+1}, \dots, x_n | x_1, \dots, x_k) = \frac{f(x_1, \dots, x_n)}{f(x_1, \dots, x_k)}$$

The r.v.'s X_1, X_2, \dots, X_k are said to be *mutually independent* if and only if their joint distribution factors into the product of their marginal distributions, that is,

$$f(x_1, x_2, \dots, x_k) = f(x_1)f(x_2), \dots, f(x_k) \quad \text{for all } x_1, x_2, \dots, x_k$$

Suppose that the r.v.'s X_1, X_2, \dots, X_n have variances $\sigma_1^2, \sigma_2^2, \dots, \sigma_n^2$, respectively, and pairwise covariances

$$\sigma_{ij} = \text{Cov}(X_i, X_j) \quad \text{for } 1 \leq i \neq j \leq k$$

If we form a $k \times k$ matrix Σ , which has the i th diagonal entry equal to $\text{Var}(X_i) = \sigma_i^2$ and the (i, j) th off-diagonal entry equal to $\text{Cov}(X_i, X_j) = \sigma_{ij}$, then

$$\Sigma = \begin{pmatrix} \sigma_1^2 & \sigma_{12} & \cdots & \sigma_{1k} \\ \sigma_{21} & \sigma_2^2 & \cdots & \sigma_{2k} \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_{k1} & \sigma_{k2} & \cdots & \sigma_k^2 \end{pmatrix} = \text{Cov}(X_i, X_j) \quad \text{for } 1 \leq i \neq j \leq k$$

is called the *variance-covariance matrix* (or just the *covariance matrix*) of the X_i 's. Note that Σ is a symmetric matrix, since $\sigma_{ij} = \sigma_{ji}$ for all $i \neq j$. Σ is positive definite if $a \Sigma a > 0$ for all a . If Σ is positive definite, there are orthogonal matrices U and V such that

$$U \Sigma V = D = \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_n) \quad \text{with } \lambda_i > 0$$

These λ_i 's are called eigenvalues, which represent the dispersion corresponding to each eigenvector direction.

Analogously, we can define the correlation matrix of the X_i 's. This matrix, defined by R , has all diagonal entries equal to 1 and the (i, j) th off-diagonal entry equal to $\rho_{ij} = \text{Corr}(X_i, X_j)$ ($1 \leq i \neq j \leq k$). Thus

$$R = \begin{pmatrix} 1 & \rho_{12} & \cdots & \rho_{1k} \\ \rho_{21} & 1 & \cdots & \rho_{2k} \\ \vdots & \vdots & \ddots & \vdots \\ \rho_{k1} & \rho_{k2} & \cdots & 1 \end{pmatrix}$$

where R is also a symmetric matrix, since $\rho_{ij} = \rho_{ji}$ for all $i \neq j$. It can be thought of as the covariance matrix of the standardized X_i 's, that is, the X_i 's divided by their standard deviation σ_i .

In this section, we introduce a multinomial distribution, an important family of discrete multivariate distributions. This family generalized the binomial family to the situation in which each trial has n (rather than two) distinct possible outcomes. Then multivariate normal and Dirichlet distributions will be discussed.

2.8.1 Multinomial Distribution

Let n and m be positive integers and let p_1, \dots, p_n be numbers satisfying $0 \leq p_i \leq 1$, $i = 1, \dots, n$, and $\sum_{i=1}^n p_i = 1$. Then the random vector (X_1, \dots, X_n) has a

multinomial distribution with m trials and cell probabilities p_1, \dots, p_n if the joint pmf of (X_1, \dots, X_n) is

$$f(x_{k+1}, \dots, x_n) = \frac{m!}{x_1! \cdots x_n!} p_1^{x_1} \cdots p_n^{x_n} = m! \prod_{i=1}^n \frac{p_i^{x_i}}{x_i!}$$

on the set of (x_1, \dots, x_n) such that each x_i is a nonnegative integer and $\sum_{i=1}^n x_i = m$.

The multinomial distribution is a model for the following kind of experiment. The experiment consists of m independent trials. Each trial results in one of n distinct possible outcomes. The probability of the i th outcome is p_i on every trial. And X_i is the count of the number of times the i th outcome occurred in the m trials. For $n = 2$, this is just a binomial experiment in which each trial has $n = 2$ possible outcomes and X_1 counts the number of “successes” and $X_2 = m - X_1$ counts the number of “failures” in m trials. In a general multinomial experiment, there are n different possible outcomes to count.

Example 2.29 Multivariate Distribution from Tossing a Die. Consider tossing a six-sided die 10 times. Suppose the die is unbalanced so that the probability of observing a 1 is $1/21$, the probability of observing a 2 is $2/21$, and, in general, the probability of observing an i is $1/21$. Now consider the random vector (X_1, \dots, X_6) , where X_i counts the number of times i comes up in the 10 tosses. Then what is the probability of rolling four 6s, three 5s, two 4s, and one 3?

Solution A r.v. vector (X_1, \dots, X_6) has a multinomial distribution with $m = 10$ trials, $n = 6$ possible outcomes, and cell probabilities $p_1 = 1/21$, $p_2 = 2/21, \dots, p_6 = 6/21$. Then

$$\begin{aligned} f(0, 0, 1, 2, 3, 4) &= \frac{10!}{0! 0! 1! 2! 3! 4!} \left(\frac{1}{21}\right)^0 \left(\frac{2}{21}\right)^0 \left(\frac{3}{21}\right)^1 \left(\frac{4}{21}\right)^2 \left(\frac{5}{21}\right)^3 \left(\frac{6}{21}\right)^4 \\ &= 0.0059 \end{aligned}$$

The R code for this example is as follows:

```
R command: dmultinom(c(0, 0, 1, 2, 3, 4),
  prob=c(1/21, 2/21, 3/21, 4/21, 5/21, 6/21))
Output: 0.0059
```

The factor $m!/(x_1! \cdots x_n!)$ is called a multinomial coefficient. It is the number of ways that m objects can be divided into n groups with x_1 in the first group, x_2 in the second group, and x_n in the n th group.

2.8.2 Multivariate Normal Distribution

A multivariate normal (or Gaussian) distribution is a generalization of the one-dimensional normal distribution (also called a *Gaussian distribution*) to higher

dimensions. It is also closely related to matrix normal distribution. If Σ is nonsingular, then the distribution may be described by the following p.d.f.:

$$f_X(x_1, \dots, x_N) = \frac{1}{(2\pi)^{N/2} |\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x - \mu)^T \Sigma^{-1}(x - \mu)\right)$$

where $|\Sigma|$ is the determinant of Σ . Note how the equation above reduces to that of the univariate normal distribution if Σ is a scalar (i.e., a 1×1 matrix). The vector μ in these conditions is the expected value of X and the matrix $\Sigma = AA^T$ is the covariance matrix of the components X_i . It is important to realize that the covariance matrix must be allowed to be singular (thus not described by the above formula for which Σ^{-1} is defined). That case arises frequently in statistics. Note also that the X_i 's are in general *not* independent; they can be seen as the result of applying the linear transformation A to a collection of independent Gaussian variables Z . The distribution of a random vector X is a multivariate normal distribution that can be written in the following notation:

$$X \sim N(\mu, \Sigma)$$

In the two-dimensional nonsingular case, the multivariate normal distribution reduces to the bivariate normal distribution. This bivariate normal distribution is a generalization of the familiar univariate normal distribution for a single r.v. X . Let μ_X and σ_X be the mean and standard deviation of X , μ_Y and σ_Y be the mean and standard deviation of Y , and

$$\rho = \text{Corr}(X, Y) = \frac{\text{Cov}(X, Y)}{\sqrt{\text{Var}(X)\text{Var}(Y)}}$$

be the correlation coefficient between X and Y . Then the bivariate normal probability density function of (X, Y) is given by

$$f(x, y) = \frac{1}{2\pi\sigma_x\sigma_y\sqrt{1-\rho^2}} \exp\left\{-\frac{1}{2(1-\rho^2)} \times \left[\left(\frac{x-\mu_x}{\sigma_x}\right)^2 - 2\rho\left(\frac{x-\mu_x}{\sigma_x}\right)\left(\frac{y-\mu_y}{\sigma_y}\right) + \left(\frac{y-\mu_y}{\sigma_y}\right)^2\right]\right\}$$

where ρ is the correlation between X and Y . In this case,

$$\Sigma = \begin{bmatrix} \sigma_x^2 & \rho\sigma_x\sigma_y \\ \rho\sigma_x\sigma_y & \sigma_y^2 \end{bmatrix}$$

2.8.3 Dirichlet Distribution

The *Dirichlet distribution*, often denoted $\text{Dir}(\alpha)$, is a family of continuous multivariate probability distributions parameterized by the vector α of positive real numbers. It is the multivariate generalization of the beta distribution and conjugate prior of the

multinomial distribution in Bayesian statistics. Given each event has been observed $\alpha_i - 1$ times, the probability of each event is x_i .

The p.d.f. of the Dirichlet distribution for variables $\mathbf{p} = (p_1, \dots, p_n)$ with parameters $\mathbf{u} = (u_1, \dots, u_n)$ is defined by

$$f(p) = \text{Dirichlet}(p, u) = \frac{1}{Z(u)} \prod_{i=1}^n p_i^{u_i-1}$$

where $p_1, \dots, p_n \geq 0$; $\sum_{i=1}^n p_i = 1$, $u_1, \dots, u_n > 0$, and $Z(u) = [\prod_{i=1}^n \Gamma(u_i)] / [\Gamma(\sum_{i=1}^n u_i)]$. The parameters u_i can be interpreted as “prior observation counts” for events governed by p_i .

Let $u_0 = \sum_{i=1}^n u_i$. Then the mean and variance of the distribution are

$$E[p_i] = \frac{u_i}{u_0} \quad \text{and} \quad \text{Var}[p_i] = \frac{u_i(u_0 - u_i)}{u_0^2(u_0 + 1)}$$

2.9 SAMPLING DISTRIBUTION

2.9.1 Definition of Sampling Distribution of a Statistic

After obtaining an estimator of a population parameter (μ , σ) or a parameter associated with a particular family of distributions such as λ in an exponential family, the *sampling distribution* of the estimator is the distribution of the estimator as its possible realizations vary across all possible samples that may arise from a given population. For example, let θ be a parameter for a population or for a family of distributions. Let Y_1, \dots, Y_n be i.i.d. r.v.’s with c.d.f. F completely unspecified or $F(\cdot, \theta)$, where θ is a vector of unknown parameters. Let $\hat{\theta} = \hat{\theta}(Y_1, \dots, Y_n)$ be an estimator of θ based on the observed data. We want to assess how well $\hat{\theta}$ estimates θ . In order to calculate some measures of this assessment such as mean-square error (MSE, $\text{MSE}(\hat{\theta}) = E[(\hat{\theta} - \theta)^2]$), we need to know the distribution of $\hat{\theta}$ over all possible samples from the population. This would be nearly impossible for a large population but we can envision the procedure as follows:

- Take M samples of size n from the population:
 - Sample 1: X_{11}, \dots, X_{1n} , then compute $\hat{\theta}_1$.
 - Sample 1: X_{21}, \dots, X_{2n} , then compute $\hat{\theta}_2$.
 - ...
 - Sample M : X_{M1}, \dots, X_{Mn} , then compute $\hat{\theta}_M$.
- Estimate the distribution of $\hat{\theta}$ using the M realizations of $\hat{\theta}$: $\hat{\theta}_1, \hat{\theta}_2, \dots, \hat{\theta}_M$.
- We could then estimate the c.d.f. of $\hat{\theta}$, its mean, variance, MSE, and so on.

In nearly all situations, this procedure is impossible because of cost, time, or the mere impossibility of every being able to take enough repeated samples from a fixed population. We overcome this problem in the next three sections.

2.9.2 Mathematical Derivations

First we can know the sampling distribution of a parameter considering mathematical derivations. Let X_1, X_2, \dots, X_n be a sequence of random variables. Consider the following examples:

1. If X_1, \dots, X_n are i.i.d. $N(\mu, \sigma^2)$, then the distribution of $\hat{\mu} = \bar{X}$ is $N(\mu, \sigma^2/n)$.
2. If X_1, \dots, X_n are i.i.d. $\exp(\beta)$, then the distribution of $n\hat{\beta} = \sum_{i=1}^n X_i$ is distributed Gamma (n, β) .
3. If X_1, \dots, X_n are i.i.d. Bernoulli with unknown value p , then the distribution of $\hat{p} = \bar{X}$ is obtained from the result that $n\hat{p} = \sum_{i=1}^n X_i$ is distributed Bin (n, p) .

2.9.3 Asymptotic Distribution

Asymptotic theory (large n) can also be applied in many situations. Let X_1, X_2, \dots be an infinite sequence of random variables, discrete or continuous, with respective distribution functions $F_1(x), F_2(x), \dots$. We consider the concept of *convergence in distribution* among various concepts of convergence for such a sequence. Suppose that X is a random variable whose distribution function is $F_X(x)$. Then if

$$\lim_{n \rightarrow \infty} F_n(x) = F_X(x)$$

for all points of continuity of $F_X(x)$, we say that the sequence $\{X_n\}$ converges in distributions to X . All of the convergence results in this book, and in particular the central limit theorem described below, refer to convergence in distribution.

Central Limit Theorem Assume that X_1, \dots, X_n are i.i.d., each with finite mean μ and finite variance σ^2 . The simplest and most important version of the central limit theorem states that as $n \rightarrow \infty$, the random variable $(\sum_{i=1}^n X_i - n\mu)/(\sqrt{n}\sigma)$ converges in distribution to a random variable having the standardized normal distribution, $N(0, 1)$.

Various versions of the central limit theorem give approximations to the sampling distributions of any parameter. If the r.v. X has a normal distribution, then we say that the sequence is asymptotically normal.

2.9.4 Simulation Studies

Simulation studies provide some insight to the sampling distribution but are limited in that we must specify the population distribution exactly in order to conduct the simulation. Consider the following example: Suppose we wanted to determine the sampling distribution of the estimators of its mean and variance, (θ_1, θ_2) , when sampling from a population distributed normally. We can simulate observations from a normal distribution using R but first we must design the simulation study and consider the following:

1. How many different values of the sample size n will be needed?
2. How many different values of the location and scale parameter θ_1 and θ_2 will be needed, respectively?

3. Which values of θ_1 and θ_2 should be selected?
4. How many replications of the simulation are needed for each choice of (n, θ_1, θ_2) ?
5. How can we infer the sampling distribution of $(\hat{\theta}_1, \hat{\theta}_2)$ for values of (n, θ_1, θ_2) not run in the simulation study?

Suppose we have a random sample of n units from a population with n of a modest size. We want to determine the sampling distribution of the sample median. The population distribution is completely unknown and hence a simulation study cannot be utilized. The sample size is too small to have much confidence in applying the central limit theorem for the median. A possible method for obtaining an approximation to the sampling distribution of the statistics is to use a *resampling* procedure. This involves taking numerous samples of size n (with replacement) from the actual observed data and computing the value of the statistic from each of these samples. One such procedure is called the *bootstrap sampling* procedure.

2.9.5 Empirical Distribution and Bootstrapping of Sampling Distribution

Let X_1, \dots, X_n be i.i.d. random variables with a common c.d.f. $F(\cdot)$. Let θ be a parameter which we wish to estimate using a function of the data $\hat{\theta} = \hat{\theta}(X_1, \dots, X_n)$. Suppose the c.d.f. $F(\cdot)$ is unknown and the sample size n is small or that the asymptotic distribution of $\hat{\theta}$ is intractable. We wish to determine the sampling distribution of $\hat{\theta}$ in order to be able to determine its bias as an estimator of θ or its variance or its percentiles in order to provide an assessment of how well it estimates θ . Suppose we cannot mathematically derive the true sampling distribution of $\hat{\theta}$ because the c.d.f. $F(\cdot)$ is unknown or the form of $\hat{\theta}$ may be too complex to obtain an exact result. The asymptotic distribution may not provide an adequate approximation of the true sampling distribution of $\hat{\theta}$ because n is too small.

An alternative to these two approaches is the *bootstrap procedure*, which will provide an approximation to the sampling distribution of $\hat{\theta}$ in the situation where we can write θ as a function of the c.d.f., that is, $\theta = g[F(\cdot)]$. For example:

- The population mean $\mu = \int_{-\infty}^{\infty} x dF(X)$.
- The population variance $\sigma^2 = \int_{-\infty}^{\infty} (x - \mu)^2 dF(X)$.
- The population median M can be defined by $0.5 = \int_{-\infty}^M dF(X)$.

To obtain the sample estimator, we simply replace the true c.d.f. $F(\cdot)$ with the empirical (sample) c.d.f.(e.d.f.)

$$\hat{F}(x) = \frac{1}{n} \sum_{i=1}^n I(X_i \leq x) = \frac{\#(X_i \leq x)}{n}$$

in $\theta = g[F(\cdot)]$ to obtain $\hat{\theta} = g[\hat{F}(\cdot)]$. For example, let $X_{(1,n)} \leq X_{(2,n)} \leq \dots \leq X_{(n,n)}$ be the n data values ordered from smallest to largest, called the *order statistics*. A graph of the e.d.f. for $n = 10$ with $X = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$ is given in Figure 2.12. Note, $\hat{F}(\cdot)$ is a piecewise constant function with jumps of height $1/n$ at each of the ordered

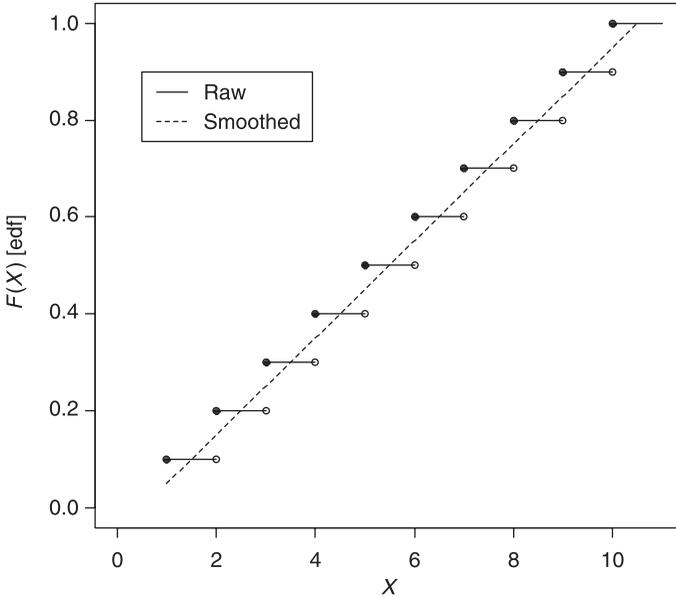


Figure 2.12 Empirical c.d.f. of the order statistics.

values $X_{(1,n)} \leq X_{(2,n)} \leq \dots \leq X_{(n,n)}$ unless there are ties (i.e., several X_i 's having the same value). In this case, the height of the jump at the tied value would be c/n , where c is the number of X_i 's having the tied value.

The “raw” e.d.f. is a step function for all datasets. If the data are from a discrete distribution, then the plot would be an appropriate plot. However, if we have observations from a continuous c.d.f., then the raw e.d.f., a step function, would not be an accurate portrayal of the population c.d.f., a continuous function. A very simple improvement to the raw e.d.f. is to simply connect the midpoints of each of the flat regions in the e.d.f., called a *smoothed e.d.f.* We can define this smoothed e.d.f. as follows: For $X_{(i,n)} < x \leq X_{(i+1,n)}$,

$$\bar{F}_n(x) = \left(\frac{i}{n}\right) \frac{X_{(i+1)} - x}{X_{(i+1)} - X_{(i)}} + \left(\frac{i+1}{n}\right) \frac{x - X_{(i)}}{X_{(i+1)} - X_{(i)}}$$

In order to obtain the sampling distribution of $\hat{\theta}$, we simulate data from the e.d.f. $\hat{F}(\cdot)$ in place of the true c.d.f. $F(\cdot)$. We will now consider the population to be the observed data having c.d.f. which places mass $1/n$ on each of the observed data values X_i . Thus, we select M random samples of size n (sampling with replacement) from this “new” population and compute $\hat{\theta}_1, \hat{\theta}_2, \dots, \hat{\theta}_M$. We now have M realizations of $\hat{\theta}$ from which we can estimate the p.d.f. (using a kernel density estimator), the quantile function, or specific parameters like its mean,

$$\mu_{\hat{\theta}} = E[\hat{\theta}] = \frac{1}{M} \sum_{i=1}^M \hat{\theta}_i$$

and its variance,

$$\text{Var}[\hat{\theta}] = \frac{1}{M} \sum_{i=1}^M (\hat{\theta}_i - \mu_{\hat{\theta}})^2$$

Similarly, we can compute its median or any other percentiles. When using a bootstrap procedure, we have two levels of approximation:

1. The estimation of the population c.d.f. F using the e.d.f. \hat{F} : accuracy of approximation controlled by n .
2. Repeatedly estimating the e.d.f. \hat{F} using the M bootstrap estimators \hat{F} : accuracy of approximation limited by the value of n .

A flow chart of the bootstrap procedure is given here:

1. Obtain data X_1, \dots, X_n i.i.d. with c.d.f. $F(\cdot)$.
2. Compute $\hat{\theta} = \hat{\theta}(X_1, \dots, X_n)$.
3. Select a random sample of size n with replacement from X_1, \dots, X_n (i.e., simulate n independent observations from the e.d.f. $\hat{F}(\cdot)$): denote by X_1^*, \dots, X_n^* .
4. Compute $\hat{\theta}^* = \hat{\theta}(X_1^*, \dots, X_n^*)$.
5. Repeat steps 3 and 4 M times, yielding $\hat{\theta}_1^*, \hat{\theta}_2^*, \dots, \hat{\theta}_M^*$.
6. Use these M realizations of $\hat{\theta}$ to construct the sampling distribution of $\hat{\theta}$: means, SDs, percentiles, p.d.f., and so on.

We will consider the following example to illustrate the application of the bootstrap procedure:

Example 2.30 Bootstrap Procedure. Suppose the life lengths of 11 engine parts are measured as 5700, 36,300, 12,400, 28,000, 19,300, 21,500, 12,900, 4100, 91,400, 7600, 1600. Estimate the median life length θ of the engine part.

From the data we compute $\hat{\theta} = X_{(6)} = 12,900$. To study the variation in this estimator we need to know its sampling distribution. We will use the bootstrap to approximate this distribution. We will generate first 200 bootstrap samples from $\hat{F}(\cdot)$ and then 20,000 bootstrap samples using the following R code:

```
Y=c(1600,4100,5700,7600,12400,12900,19300,21500,28000,
    36300,91400)
Mhat=median(y); M=20000; D=numeric(M)
for (i in 1:M) d[i]=median(sample(y,replace=T))
hist(d)
bootmean=mean(d); bootstd=sqrt(var(d));
bootquant=quantile(d)
pdf("bootexample20000.pdf")
Qd=quantile(d,probs<-seq(0,1,.01))
```

```

boxplot(d,main="Empirical Quantile for Sample Median",
ylab="Median Life Lengths of Engine Part",plot=T)
plot(probs,Qd,type="l",ylab="Q(u) for Median",xlab="u",
xlim=c(0,1),lab=c(10,11,7))
title("Empirical Quantile for Sample Median",cex=.75)
plot(density(d),type="l",xlab="Median Life
Lengths",ylab="PDF of Sample Median")
title("Empirical pdf for Sample Median",cex=.75)
qqnorm(d,main="Normal Prob Plot of Sample
Median",xlab="normal
quantiles",ylab="Sample Medians",lab=c(7,7,7),cex=.75)
qqline(d)
graphics.off()

```

In Table 2.8 the first five simulations are given with a plus indicating which of the original data values was sampled. Note that some values will be sampled multiple times and some values may not be included.

From the 200 realizations of $\hat{\theta}^*$, the following summary statistics were computed:

Average:

$$E^*[\hat{\theta}] = \frac{1}{200} \sum_{i=1}^{200} \hat{\theta}_i^* = 14,921.5$$

Standard deviation:

$$\sqrt{\text{Var}^*[\hat{\theta}]} = \sqrt{\frac{1}{200} \sum_{i=1}^{200} (\hat{\theta}_i^* - E^*[\hat{\theta}])^2} = 5784.13$$

TABLE 2.8 First Five Bootstrap Samples

Original data ordered	Bootstrap sample				
	1	2	3	4	5
1600			+		++
4100	+++	++	+	+	
5700	+	+	+	+++	+
7600				++	++
12,400	+	+	+		+
12,900	+		++		
19,300	+	+	+	++	
21,500		+++	+	+	+
28,500	+	++	+	++	+
36,300	+	+			+
91,400	++		++		++
$\hat{\theta}$	12,900	21,500	12,900	7600	12,400

Quantile				
0	0.25	0.50	0.75	1.0
4100	12,400	12,900	19,300	36,300

If we extended the simulation to 20,000 bootstrap samples, we obtain:

Average:

$$E^*[\hat{\theta}] = \frac{1}{20,000} \sum_{i=1}^{20,000} \hat{\theta}_i^* = 14,981.56$$

Standard deviation:

$$\sqrt{\text{Var}^*[\hat{\theta}]} = \sqrt{\frac{1}{20,000} \sum_{i=1}^{20,000} (\hat{\theta}_i^* - E^*[\hat{\theta}])^2} = 5850.69$$

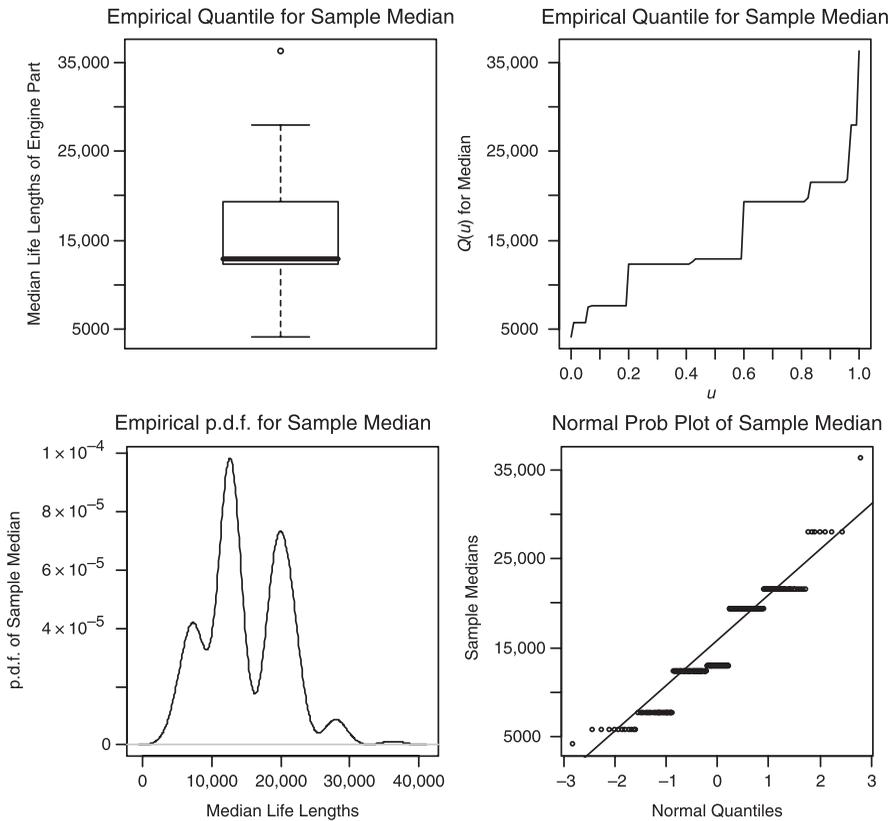


Figure 2.13 Plots of sampling distribution of sample median based on 200 resamples.

Quantile				
0	0.25	0.50	0.75	1.0
1600	12,400	12,900	19,300	91,400

Thus, there were only minor differences in the mean and standard deviation for the sampling distribution of the median when comparing 200 bootstrap samples to 20,000 bootstrap samples. However, note the big discrepancies between the quantiles. When generating 20,000 samples of size 11 from the original dataset, samples were obtained in which the median of the bootstrap sample was equal to the minimum value (1600) in the original dataset. Because the bootstrap median equals $\hat{\theta}^* = X^*_{(6)}$, this result implies that, in the bootstrap samples having median = 1600, at least 6 of the 11 data values must be equal to 1600. This seems very unlikely. However, if we calculate the expected number of samples in the 20,000 samples having exactly 6 of their 11 values equal to 1600, we find:

$$\begin{aligned} \text{Expected number} &= 20,000 \times \Pr[\text{exactly 6 of 11 values equal 1600}] \\ &= (20,000) \left[\binom{11}{6} (1)^6 (10)^5 \right] / (11)^{11} = 3.2 \end{aligned}$$

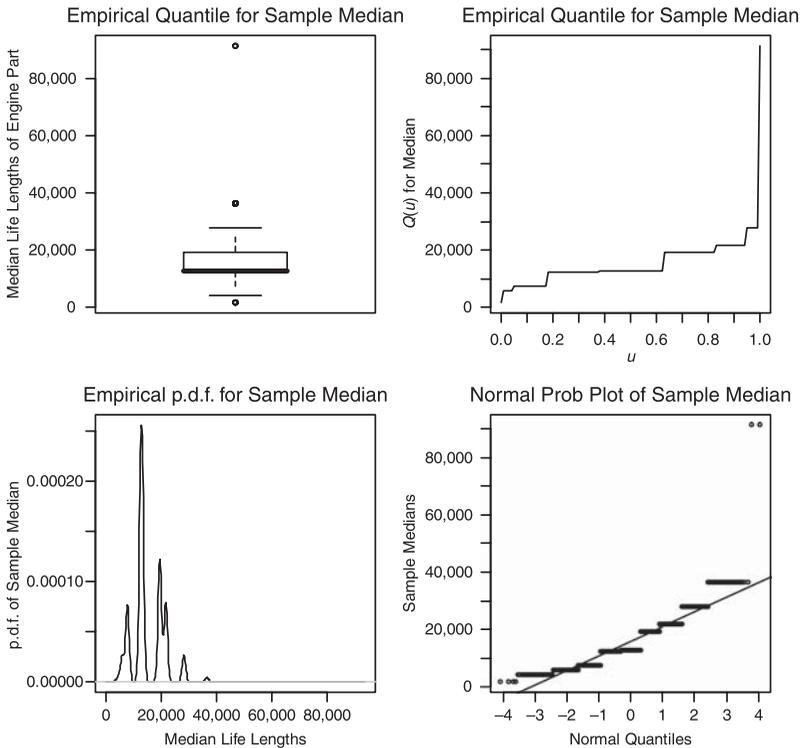


Figure 2.14 Plots of sampling distribution of sample median based on 20,000 resamples.

Therefore, on the average, we would expect 3.2 occurrences of the event that exactly 6 of the 11 data values were equal to 1600.

A plot of the quantile function, kernel density estimator of the p.d.f., a box plot, and a normal reference distribution plot for the sampling distribution of the sample quantile are given in Figures 2.13 and 2.14 for 200 and 20,000 bootstrap samples. We note that there are considerable differences in the plots. The plots for 20,000 bootstrap samples reveal the discreteness of the possible values for the median when the sample size ($n = 11$ in our case) is very small. Also, we note that $n = 11$ is too small for the sampling distribution for the median to achieve its asymptotic result (n large), an approximate normal distribution.

A good reference on bootstrapping is *Bootstrap Methods and Their Application* by D. V. Hinkley and A. C. Davison, Cambridge University Press, New York, 1997.

2.10 SUMMARY

This chapter reviews basic ideas of probability, random variables, probability distributions, and sampling distributions of a statistic. The axiomatic approach is followed to define probabilities of events in a sample space, which is the collection of all possible outcomes of an experiment. When the outcomes are equally likely, the probability of an event is just the ratio of the number of outcomes in the event to the number of outcomes in the sample space.

The conditional probability of an event A given that an event B has occurred, $P(A | B)$, defined as $P(A \cap B)/P(B)$. Events A and B are independent if $P(A \cap B) = P(A)P(B)$. The Bayes theorem forms an important basis of a Bayesian statistics as a reexpression of the conditional probability formula to find $P(B | A)$ in terms of $P(A | B)$, $P(A)$, and $P(B)$.

A r.v. X assigns a unique numerical value x to each outcome in the sample space. Random variables are classified as discrete or continuous. The probability distribution of a discrete or continuous r.v. is given by its p.m.f. $f(x) = P(X = x)$ or p.d.f. for all values x . The p.d.f. of a continuous r.v. has the property that $P(a \leq X \leq b)$ is given by the area under the p.d.f. curve between a and b for any interval $[a, b]$. The c.d.f., defined as $F(x) = P(X \leq x)$ for all x , unifies these two concepts.

The mean or the expected value is useful for summarizing a symmetrically distributed r.v. X ; that is, the mean, $E(X) = \mu$, is a measure of the center of the distribution, and the variance, $\sigma^2 = E[(X - \mu)^2]$, is a measure of dispersion of the distribution about its mean.

The above concepts can be generalized to two (or more) r.v.'s, say, X and Y . The joint probability distribution (p.d.f. or p.m.f.) of X and Y is denoted by $f(x, y)$. The marginal distribution $g(x)$ of X is just the p.m.f. (or p.d.f.) of X , which is obtained by summing (or integrating) $f(x, y)$ over y for fixed x . The covariance $\sigma_{XY} = E[(X - \mu_X)(Y - \mu_Y)]$ of X and Y measures their joint dispersion from their respective means. The correlation coefficient ρ_{XY} standardizes the covariance to the range $[-1, +1]$. Both are measures of the linear relationship between X and Y . This concept can be generalized to more than two r.v.'s, say, X_1, \dots, X_n . When they are jointly distributed, the multivariate distribution can be defined. A multivariate distribution is the natural extension of a bivariate distribution.

It is helpful to have standard probability models that are useful for analyzing large biological data, in particular bioinformatics. There are six standard distributions for discrete r.v.'s, that is, Bernoulli for binary r.v.'s, (e.g., success or failure), binomial for the number of successes in n independent Bernoulli trials with a common success probability p , uniform for model situations where all integer outcomes have the same probability over an interval $[a, b]$, geometric for the number of trials required to obtain the first success in a sequence of independent Bernoulli trials with a common success probability p , Poisson used to model the number of occurrences of rare events, and negative binomial for the number of successes in a fixed number of Bernoulli trials, each with a probability p of success.

Five standard distributions for continuous r.v.'s are uniform for model situations where all outcomes over an interval $[a, b]$ are equally likely, normal playing an important role in explaining many real-life phenomena, exponential for a continuous counterpart of the geometric distribution used to model the time to an event, gamma obtained by summing n i.i.d. exponential r.v.'s, and beta being a generalization of the uniform distribution.

The normal distribution is the most important and extensively used continuous distribution. The mean μ and variance σ^2 completely characterize a normal distribution, denoted by $N(\mu, \sigma^2)$. A normal distribution with $\mu = 0$ and $\sigma^2 = 1$ is called the standard normal distribution; that is, when measurements are made in the units of σ from the mean μ , all normal distributions are the same. Therefore the probabilities involving $X \sim N(\mu, \sigma^2)$ can be expressed and calculated in terms of $Z = (X - \mu)/\sigma \sim N(0, 1)$.

The sampling distribution of a statistic is its probability distribution, obtained by considering how the statistic value varies in repeated random samples drawn from the same population distribution. The sampling distribution of a statistic is useful for assessing the sampling error in the inferences based on that statistic about a population parameter.

For a random sample of size n (i.e., n i.i.d. observations) from any population with mean μ and variance σ^2 , the mean and variance of the distribution of the sample mean, \bar{X} , equal are given as

$$E(\bar{X}) = \mu \quad \text{and} \quad \text{Var}(\bar{X}) = \sigma^2/n.$$

The central limit theorem (CLT) states that, for large n , $\sum_{i=1}^n X_i$ is approximately distributed as $N(n\mu, n\sigma^2)$. If the population distribution is normal, then this result is exactly true for any n . One application of the CLT is to approximate the $\text{Bin}(n, p)$ distribution by a normal distribution with mean np and variance $np(1 - p)$. This approximation is made more accurate by applying the continuity correction.

Resampling methods draw repeated samples from the observed sample itself to generate the sampling distribution of a statistic. The permutation method draws samples without replacement while the bootstrap method draws samples with replacement. These methods are useful for assessing the accuracy (e.g., bias and standard error) of complex statistics.

QUALITY CONTROL OF HIGH-THROUGHPUT BIOLOGICAL DATA

Paul D. Williams

*Department of Public Health Science, University of Virginia,
Charlottesville, Virginia, USA*

Many potential sources of variability and error in high-throughput biological experiments can make it difficult to distinguish the causes of phenotypic differences. In this chapter we discuss methods of data transformation and normalization that can be used in order to distinguish meaningful biological variability from technical error.

3.1 SOURCES OF ERROR IN HIGH-THROUGHPUT BIOLOGICAL EXPERIMENTS

High-throughput biological experiments such as transcriptional genomic profiling or shotgun proteomics analyses provide simultaneous measurements of the relative levels of expression of thousands of messenger RNA (mRNA) transcripts or proteins in a biological sample of interest. A fundamental characteristic of such analyses is the comparison of relative expression levels in order to explore the molecular causes underlying phenotypic differences or to examine molecular responses to a given stimulus. For instance, to determine which genes are relevant to cancer progression and metastatic invasion, one might compare gene expression profiles of tumor samples taken from patients with stable, low-stage disease to those taken from patients with more aggressive tumors. To discover a blood biomarker to be used as a test for the presence of disease, one might perform tandem mass spectrometric analysis on processed blood samples to determine which proteins are consistently present in patients with the disease and consistently absent in healthy patients. In order for the conclusions of these studies to be meaningful, the measurements of gene or protein expression must be accurate, precise, and directly comparable.

Therefore, it is important to understand and attempt to minimize potential sources of error in measurement as the experiment is conducted, but it is often impossible to completely eliminate such errors. Mathematical techniques that allow more meaningful direct comparisons of gene or protein expression levels thus become highly important for correct interpretation of high-throughput biological data.

As an example, let us examine the potential sources of error in a hypothetical shotgun proteomics experiment. First, investigators collect samples of the biological material of interest from organisms classified in groups corresponding to different biological factors depending on the purpose of the experiment. Second, they process these samples such that the proteins are isolated and then digested into peptides; highly abundant proteins such as serum albumin may also be removed from the sample so that less abundant proteins of interest may be more easily detected. Then, using a liquid chromatography column, they partially separate the peptide mixture, which is then ionized for the initial spectroscopic step; different peptide ions will be selected for collisional fragmentation and daughter ion detection. Finally, different computational techniques will be applied to determine which peptides and proteins are originally in the sample and in what quantities.

Variability may arise from each of these steps. First, the amounts of the biomolecules are inherently variable between biological samples within the same group and between treatment groups. Second, the technical steps involved in sample processing introduce variation: Different sample collectors may have slightly different techniques; collection at different times of the day or year may affect the chemical properties of the sample. The amount of time samples sit between collection and processing may affect the chemical makeup; for instance, degradation by biologically active molecules within the sample may make it difficult to determine the presence of certain molecules within the sample. Removal of certain proteins from a sample may also inadvertently cause the loss of others. Different columns, chromatographic settings, ionization methods, spectroscopy methods, and computational protein identification techniques may also have effects on the measured abundances of particular proteins.

Obviously, experimental techniques should be standardized as much as possible within an experiment in order to avoid confounding biological and technological variability. Multiple biological replicates, or samples from different individuals, should also be sampled in order to measure the variability of expression given a particular biological condition. If the budget and amount of sample allow, technical replicates, or multiple analyses of different aliquots of the same sample, should also be performed, particularly for proteomics analyses where not all peptides in a sample may be detected in each experimental run. Technical replicates are somewhat less important for genomic analyses but can assist in assessing laboratory technique. Close collaboration with experimentalists can lead to optimal experimental design. Groundwork is being laid for the development of experimental standards in microarray experiments with the publication of robust external RNA controls (External RNA Controls Consortium, 2005) and data corresponding to highly standardized reference RNA hybridized to a wide variety of different types of chips (MAQC Consortium, 2006).

3.2 STATISTICAL TECHNIQUES FOR QUALITY CONTROL

Regardless of the provenance of a given high-throughput biological dataset, it is important to carefully examine distributional properties of the data and to assess their quality prior to any further analysis. The main purpose of such an initial investigation is twofold. The first goal is to examine whether the data are appropriate for any subsequent analysis. For instance, it is often desirable and may be required that the data have a well-behaved distribution, such as a Gaussian, for many statistical analysis approaches. For instance, even something as simple as a sample mean may be dominated by relatively few data points with very large values; if the dataset can be transformed such that the new distribution is well behaved, the sample mean of the transformed dataset will be less misleading and more meaningful. Examination of these distributional properties of the dataset can point the way toward appropriate quality control procedures, for example, data normalization, transformation, thresholding, or outlier elimination. The second goal is to investigate the relationships of the data corresponding to different samples and replicates. For example, are the data from particular samples considerably different from the others? Are some data from a group of patients highly correlated? Are there strong correlation effects between replicated samples or common to samples processed with certain biotechnical instrumentation settings?

Basic summary statistics and graphical assessments of the data distributions are often invaluable to this initial examination. For example, Table 3.1 shows summary statistics for the distribution of protein expression levels measured in a proteomics experiment involving the three patients sampled before and after a chemotherapeutic treatment (Cho et al., 2007).

3.2.1 Data Transformation

These summary statistics can already provide several important observations on the data. For example, the maximum values are very high compared with the rest of

TABLE 3.1 Summary Statistics of Protein Expression Levels

	Patient 1		Patient 2		Patient 3	
	Untreated	Treated	Untreated	Treated	Untreated	Treated
Minimum	12,300	15,200	477,000	482,000	505,000	396,000
First quartile	156,000	172,500	1,437,500	1,700,000	1,220,000	1,760,000
Median	216,500	238,000	200,000	3,080,000	1,760,000	2,480,000
Mean	583,234	959,898	3,481,000	5,557,333	2,824,576	4,508,758
Third quartile	332,250	469,250	3,835,000	4,232,500	2,580,000	4,440,000
Maximum	15,900,000	28,400,000	14,600,000	43,100,000	13,500,000	22,900,000
NAs	9	9	41	41	26	26

the observations, sometimes more than 50 times larger than the third quartiles. This indicates that these data are highly positively skewed, with a mixture of many low-intensity values and a small number of very large values, as is common with raw data from high-throughput biological experiments. If this kind of skewed data distribution is not appropriately adjusted, it can result in unwanted artifacts in many subsequent statistical analyses.

As an example, suppose there are two simulated protein expression vectors with randomly generated values between zero and unity. As shown in Figure 3.1*a*, these two vectors are poorly correlated, as expected. However, as seen in Figure 3.1*b*, the addition of an additional outlying point to each of these vectors changes the situation entirely—they become highly correlated as soon as the last outlying data point is added. In a search for proteins with associated expression patterns using a simple correlation coefficient to assess the association, these two genes would falsely be considered highly associated, due to one data point being much larger than the others. This underscores the need for appropriate transformation of highly skewed raw data prior to statistical analysis as well as the importance of graphical data analysis.

The goal of data transformation is to make the transformed distribution close to a Gaussian or another well-behaved statistical distribution. Logarithmic transformation is often used on a dataset with a highly positively skewed distribution. Figure 3.2 shows the effects of such a transformation. A histogram of the entire proteomics dataset summarized in Table 3.1 is shown in Figure 3.2*a*. Note that while the majority of the observed expression levels are low, there are a few highly expressed proteins. The histogram of the \log_2 -transformed data is shown in Figure 3.2*b*. This transformed distribution much more closely resembles a Gaussian curve and is more amenable to further statistical analysis. For biological experiments, binary logarithms (base 2) are often performed because fold changes can easily be determined from differences between transformed expression levels; a difference value of 1 is equal to a twofold

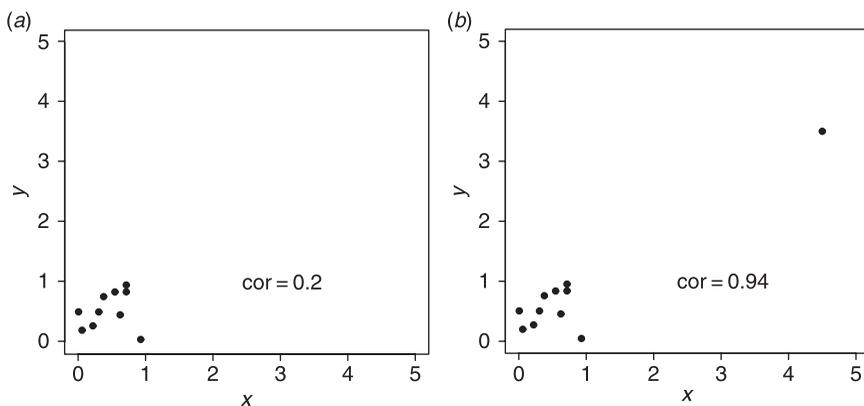


Figure 3.1 Effects of an outlier on correlation assessment. (a) Randomly distributed data have a low correlation coefficient. (b) The addition of an additional outlier results in a high correlation coefficient, although no relationship exists between the two samples.

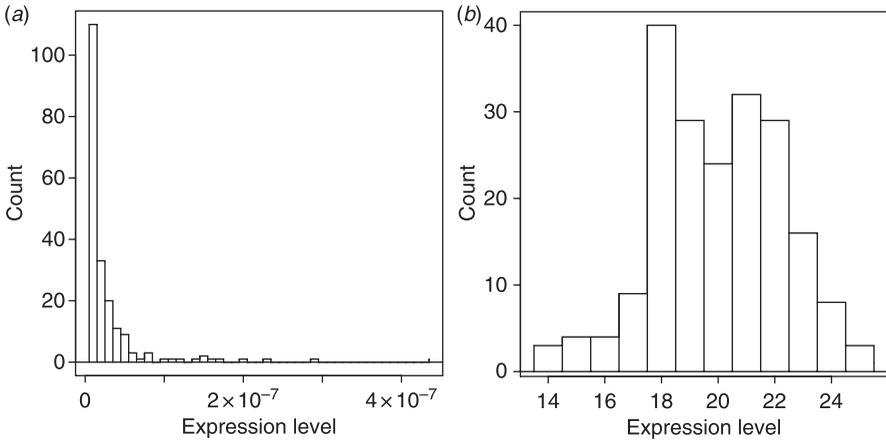


Figure 3.2 Effects of logarithmic transformation on the distribution of right-skewed data. (a) The distribution of all protein expression values in the proteomics experiment indicates many proteins with low expression but a long tail of few proteins with very high expression. (b) A histogram of the log-transformed values in this dataset shows a more Gaussian distribution.

change in expression, a difference value of 3 corresponds to an eightfold change ($2^3 = 8$), and so forth.

As the logarithm fails to produce numeric results for values less than or equal to zero and produces negative numbers for values less than 1, transformed datasets may contain undefined values or other values that behave poorly in subsequent statistical analyses. To exclude such values from the transformed dataset, it may be necessary to set all values below a certain threshold equal to that threshold value. This is called thresholding and is commonly performed with a threshold value of unity, so that all extremely small values become zero.

Although the logarithm is often a good choice for transforming the distribution of a dataset, more refined methods may also be used. The Box–Cox transformation (Box and Cox, 1964) can be used to improve the linear fit of two variables x and y by transforming x according to the following:

$$f(x; \lambda, a) = \begin{cases} [(x + a)^\lambda - 1]/\lambda & \text{for } \lambda \neq 0 \\ \log(x + a) & \text{for } \lambda = 0 \end{cases}$$

where $f(x; \lambda, a)$ is the transformed data and λ and a are tuning parameters that optimize the fit between $f(x; \lambda, a)$ and y .

The Fisher's z -transformation is a specialized procedure which can often make the distribution of correlation coefficients r close to a normal distribution:

$$f(r) = \frac{1}{2} [\ln(1 + r) - \ln(1 - r)]$$

It is important to remember that the goal of data transformation is to convert a poorly behaved distribution to a well-behaved one. If the distribution is already well behaved, further transformation may be unnecessary and may even lead to erroneous

conclusions in subsequent analyses. Datasets publicly available online may or may not have been processed or transformed, and it is important to ascertain if any transformation has already occurred through statistical and graphical analysis.

From the summary statistics above it is also possible to see that there was less variability in the ranges of data values for a given pair of patient samples than among the three treated or untreated samples. As the goal of such an experiment might be to compare proteins differentially expressed as a result of the treatment, the fact that there is more similarity in samples from a given patient than in samples given the same condition may indicate that extra care is required in comparisons. Since the treated and untreated samples for a given patient were labeled with different markers, mixed, and then this sample mixture was processed simultaneously, this implies that the data have been considerably affected by the liquid chromatography–tandem mass spectrometry (LC–MS/MS) experimental settings for each of the paired samples. If the three biological replicates for each condition are simply averaged

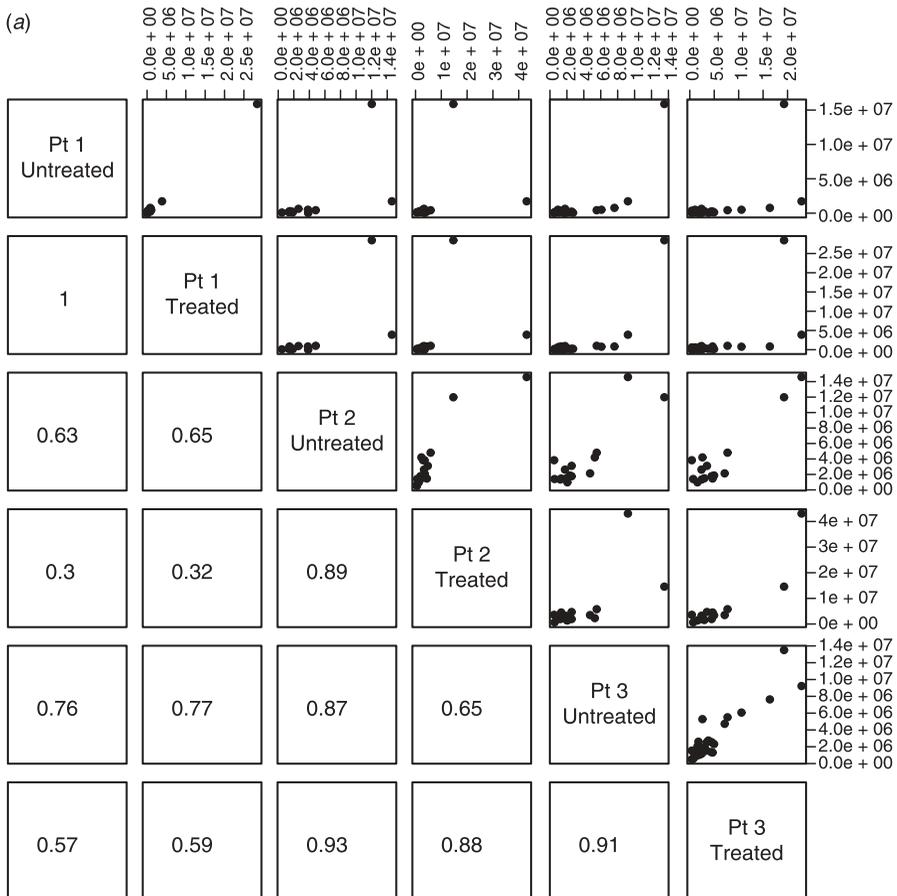


Figure 3.3 Pairwise scatter plots and correlation coefficients: (a) before and (b) after a log transformation.

without considering these effects, important biological signals of protein expression may be severely obscured by the experimental artifacts. Therefore, it is important to adjust the data for such bias. A more careful examination of this kind of experimental effect can be made graphically in pairwise scatter plots (Fig. 3.3). However, this kind of display can also be affected by the highly skewed data distribution, being dominated by a few large values (Fig. 3.3a). An appropriate data transformation (e.g., log transformation) may partially remedy this problem (Fig. 3.3b). However, Figure 3.3b shows that each of the three patient pairs were highly correlated (correlation coefficients 0.95, 0.9, 0.85), whereas the biological replicates—the untreated and treated—were less correlated (correlation coefficients ranging from 0.62 to 0.77), further suggesting some kind of experimental bias. Therefore, if such an artifact is observed, one may want to carefully choose appropriate analysis methods which can take into account and remedy such effects.

Graphical techniques are often invaluable to statistical analysis (Anscombe, 1973). Two useful techniques include the box plot and the QQ plot. While relatively simple, the box plot is useful for rapid examination of the distributional properties of

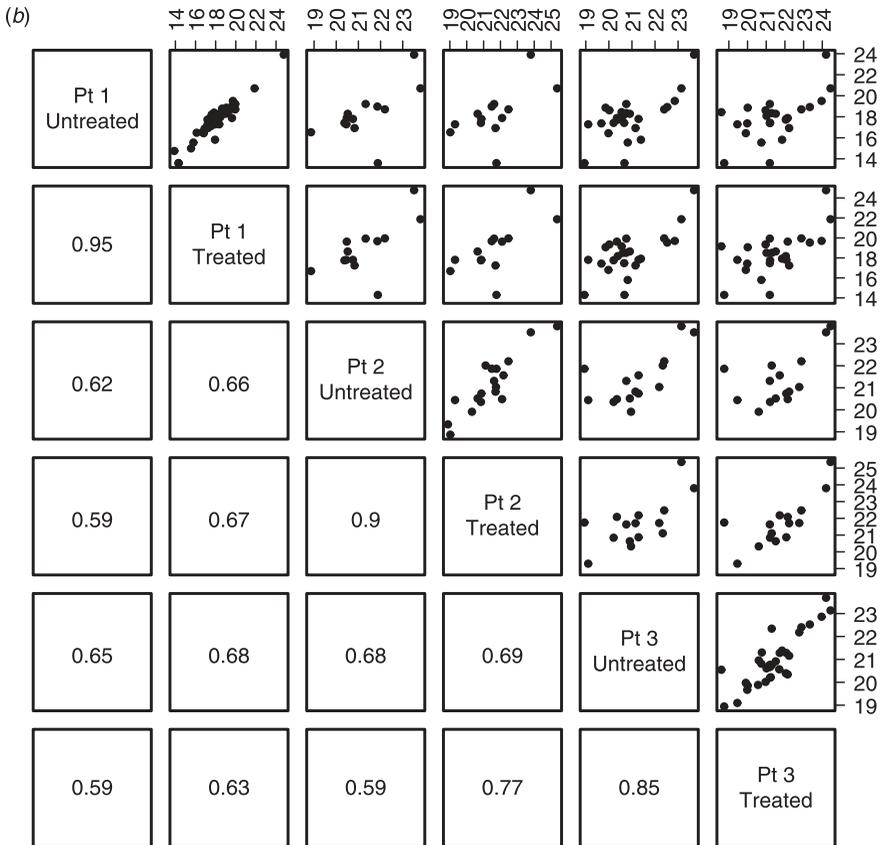


Figure 3.3 Continued.

samples, which can serve as a guide for transformation and normalization but can also be used to infer patterns in the data. For example, the graphical counterpart of Table 3.1 is shown in Figure 3.4*a*: median values, quartile ranges, extrema, and in this case outliers are clearly shown. In a QQ plot, the quantile values of one distribution are plotted against the corresponding quantiles of another. Frequently these are used to assess the amount of deviation from normality; a plot of an observed

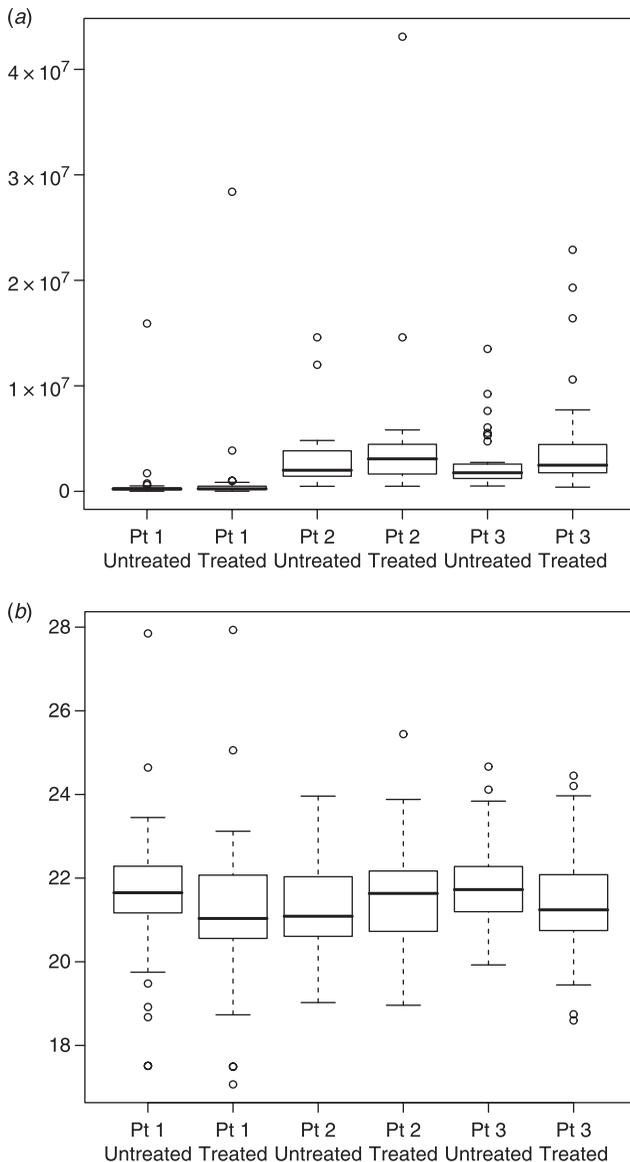


Figure 3.4 Boxplots of mass spectrometry data (a) before and (b) after log transformation.

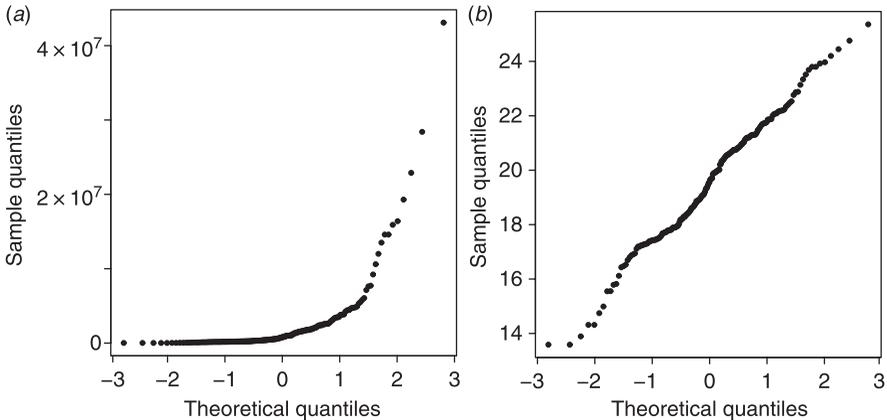


Figure 3.5 Normal QQ plots for mass spectrometry data (a) before and (b) after log transformation.

distribution against a normal distribution with the same mean and variance as the observed distribution will exhibit a straight line if the observed distribution is normal. QQ plots of the mass spectroscopy data before and after log transformation against theoretical normal distributions are shown in Figure 3.5. Again, Figure 3.5b shows that the transformed data more closely approximates a normal distribution.

3.2.2 Data Normalization

As we have just seen from Table 3.1, the ranges of protein expression in different samples are quite heterogeneous. These heterogeneous distributions are experimental artifacts that can often be found among different samples and replicates in various high-throughput biological experiments due to varying experimental parameters and instrumental settings. For instance, if a given mixture of peptides was too dilute, the measured expression amounts might be consistently lower than the true values. If this dilute sample were compared to a technical replicate with the correct concentration, many proteins might incorrectly be considered significantly differentially expressed. In general, differential expression of important biological genes or proteins can be completely obscured without proper normalization. Therefore, appropriate data normalization is required prior to subsequent statistical analysis. Several different approaches are commonly used based on assumptions about the characteristics of the data.

1. *Constant Shift* One simple method of sample normalization is to add or subtract a constant to all expression values in a sample to match the means (or other summary statistics) of different datasets. For instance, if the overall median expression value for a given sample is subtracted from all the expression values in that sample, the median expression value will be shifted to zero. When this procedure is performed on all samples in an experiment, the median values will all be equal. While this method has the benefit of preserving the relative

levels of expression, this minimally adjusts expression values and does not address differences in variance between samples.

2. *Scaling* This normalization procedure is performed to match the means (or other summary statistics, such as the median) and ranges of different datasets by multiplying the scaling factor ratios adjusting for their differences:

$$f(x_{ij}) = x_{ij} \times (\langle x_{\text{ref}} \rangle / \langle x_i \rangle)$$

where x_{ij} is the expression value for gene j in sample i , $\langle x_{\text{ref}} \rangle$ is the mean of the expression values for a reference sample, and $\langle x_i \rangle$ is the mean of the expression values in sample i . This technique is applied when the experimental bias between different replicates is only a constant multiplication factor due to, for example, slightly varying amounts of sample or different laser excitation intensities in different experiments.

3. *Interquartile Range Normalization* This procedure is used to match the variance of different samples in a dataset, assuming that the majority of the genes in the experiment are invariant in their expression values. For example, the middle halves of the distribution, between the 25th and 75th percentiles of expression values, may be matched by both multiplying scaling factors and subtracting their median differences:

$$f(x_{ij}) = (x_{ij} + m_x - m_i) \times \frac{D_x}{D_i}$$

where x_{ij} is the expression value of gene or protein j in sample i , m_x is the global median of all expression values over all samples, m_i is the chip median for all expression values in chip i , D_x represents the difference between the 75th and 25th percentiles over all expression values in all samples, and D_i represents the same value for chip i .

4. *Nonparametric Regression-Based Normalization* This normalization is performed when considerable nonlinear relationships are found among replicates across different intensity ranges. Nonparametric smoothing techniques such as LOWESS are often used for this normalization procedure (Cleveland, 1979). One drawback of this kind of normalization is that the intensity values are nonlinearly transformed after normalization, so that their linear relationships such as fold changes may not be completely preserved. Loess methods are frequently encountered in microarray analyses.

3.3 ISSUES SPECIFIC TO MICROARRAY GENE EXPRESSION EXPERIMENTS

Microarray gene expression analyses are a popular type of high-throughput biological experiment designed to simultaneously measure the expression values of thousands of genes. Although there are various types of microarray technologies, there are inherent commonalities in their design. Unlike proteomics experiments, the identities and design of the chips are known in advance, so there is much less uncertainty in the

identification of the molecules being measured, and the focus is concentrated on the precise quantitation of gene expression. Briefly, microarray expression analyses are designed to measure the relative abundances of mRNA transcripts in a given sample by competitively hybridizing their fluorescently labeled, reverse transcribed complementary DNA strands to a chip coated with oligonucleotide or complementary DNA (cDNA) sequences of genes of interest. The hybridized chips are then fluorescently scanned, and the abundance of the molecule corresponding to a given probe on the chip is proportional to the fluorescence intensity at the location on the chip.

Chip technologies differ in the type, number, and distribution of probe sequences across the chip as well as the number of samples that are hybridized to a chip. The cDNA microarrays (Schena et al., 1995) are constructed using a robotic printing device which deposits long cDNA sequences onto glass slides. The mRNA from a sample to be analyzed is reverse transcribed into cDNA, which is then labeled with a fluorescent dye. Often, two different samples are labeled with two different dyes, and the two samples are then hybridized to one chip. Expression ratios may thus be measured on the same chip, but subtle differences in the chemical properties of the dyes may lead to systematic error, which should be accounted for. A variety of experimental designs of varying complexity have been proposed (Churchill, 2002), including dye swaps, where two samples are hybridized to two chips, with the labels reversed on the second chip. Such designs must be fully balanced, meaning all samples are labeled with both dyes. Another simple but useful design involves a reference sample, labeled with one dye, hybridized to all chips in a cDNA experiment, while the samples of interest are labeled with the other dye.

In Affymetrix chips (Lockhart et al., 1996) genes are represented by multiple short (25-mer) oligomers corresponding to different sequence positions in the gene. Each probe set is comprised of 11 perfect-match and 11 mismatch probes, where the thirteenth base of the mismatch probe is mutated. These probes are distributed throughout the chip to minimize spatial effects. Only one dye is used for Affymetrix chips. Illumina chips (Fan et al., 2004) are a newer technology which involve a large number of beads coated with nonbiological DNA address sequences randomly distributed on a slide with regularly spaced wells. Before the chips are used for expression analysis, they are assayed to determine the location of beads with different address sequences on the chip. RNA from samples of interest is reverse transcribed into cDNA and then amplified using specially designed polymerase chain reaction (PCR) primers, so that fluorescent labels and sequences complementary to the bound address tags are incorporated into the amplified sequence. These amplified mixtures are then hybridized to the chips, so that the sequences with complementary address tags bind to their appropriate bead.

One useful graphical assessment of two different microarray samples is the M - A plot. In this plot, M , the difference in log-transformed intensity values of gene i on chips j and k [$\log(x_{ji}) - \log(x_{ki})$] is plotted against A , the average ($[\log(x_{ji}) + \log(x_{ki})]/2$). This is effectively a coordinate transformation such that the line of equality ($x_j = x_k$) becomes $M = 0$. Such a transformation has the benefit of making deviations from the median as well as differences in expression variance as a function of expression intensity more clear. An example of such a transformation is shown in Figure 3.6.

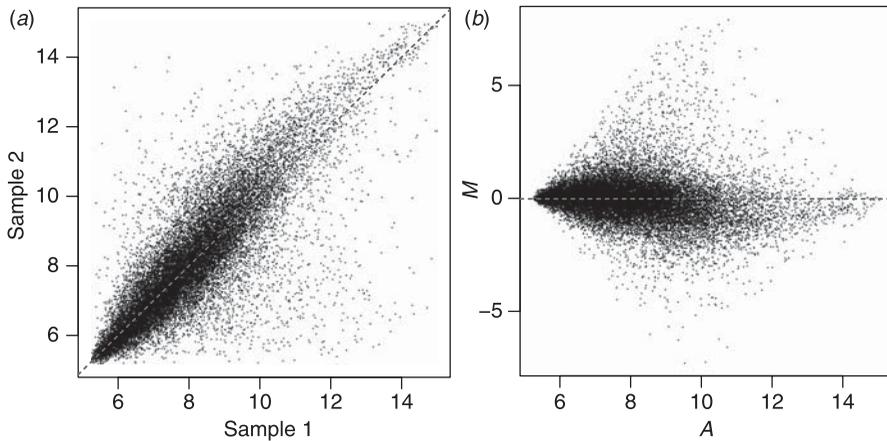


Figure 3.6 Graphical comparison of two microarray samples. (a) Expression values for Sample 2 are plotted against the expression values of Sample 1; the dashed line has slope 1 and intercept 0. (b) The MA plot for the same two samples; the difference in expression values, M , is plotted against the average of expression values, A . The dashed line represents the median of A and has slope 0.

A key assumption about microarray data is that the expression levels of the majority of genes are expected to remain minimally variant across all chips in an experiment. This means that the distributions of the expression values for the different samples are assumed to be very similar and individual gene expression values will generally have little effect on the distributional properties. The majority of microarray-specific normalization and transformation techniques are designed with this assumption in mind. Numerous statistical models have been developed to minimize systematic error resulting from a variety of sources depending on the type of chip.

For Affymetrix chips, after log transformation and background correction, there are a variety of normalization methods. The MAS 5 algorithm linearly scales expression intensities so that the trimmed mean expression values are consistent across an experiment. The model-based expression index of Li and Wong (2001a and b), released as dChip, models expression of probe pair j of gene n in sample i as

$$y_{ijn} = \text{PM}_{ijn} - \text{MM}_{ijn} = \theta_{in} f_{jn} + e$$

where PM and MM are the perfect-match and mismatch probes, θ_{in} is the expression index for gene n on chip i , f_{jn} is the effect of probe j , and e is the general error term. This method also makes use of nonlinear scaling techniques in order to bring the distributions of different chips in line with one another. In addition, this technique can also be used to locate outlier probes that may be a result of scratching or dust on the chips. A powerful technique that only involves the perfect-match probes is the robust multiarray average (RMA) model, which uses an additive model to model the expression:

$$Y_{ijn} = \theta_{in} + f_{jn} + e$$

Additionally, RMA uses quantile normalization (Bolstad et al., 2003), which normalizes the chip expression values such that quantiles are normalized against each other. RMA probe-level modeling also allows for pseudoimaging of chip residuals and other quality control metrics (Bolstad et al., 2005).

For cDNA arrays, the effects of spatial artifacts and print tips are often accounted for using statistical models. Loess normalization can be performed separately for the genes in each technical block, microarray print tip (print-tip normalization), and sometimes even with different scaling factors (scaled print-tip normalization) (Yang et al., 2002).

3.4 CONCLUSION

Quality control techniques are crucial components of any high-throughput biological analysis workflow. In summary, quality control procedures should include:

- Examination of the distributional properties of expression values
- Application of appropriate transformations to ensure that the expression value distributions are well behaved
- Application of scaling or shifting techniques to reduce systematic biases
- Graphical analysis to discover outlying samples or expression values, to test the goodness of fit to a distribution, or to discover any other irregularities in the data.

Depending on the source of the data, care may be required in the application of these various techniques. Fortunately, many of these techniques are already implemented in a variety of statistical software programs, such as R, so that data can be quickly and appropriately transformed. More effort can then be spent in order to answer the fundamental biological questions of interest.

REFERENCES

- Anscombe, F. J. (1973). Graphs in statistical analysis. *Am. Stat.*, **27**(1): 17–21.
- Bolstad, B. M., et al. (2005). Quality assessment of Affymetrix GeneChip data. In *Bioinformatics and Computational Biology Solutions Using R and Bioconductor*, R. Gentleman, R. A. Irizarry, V. J. Carey, S. Dudoit, and W. Huber (Eds.). New York: Springer.
- Bolstad, B. M., et al. (2003). A comparison of normalization methods for high density oligonucleotide array data based on variance and bias. *Bioinformatics*, **19**(2): 185–193.
- Box, G. E. P., and Cox, D. R. (1964). An analysis of transformations. *J. R. Stat. Soc. B (Methodol.)*, **26**(2): 211–252.
- Cho, H., et al. (2007). Statistical identification of differentially labeled peptides from liquid chromatography tandem mass spectrometry. *Proteomics*, **7**(20): 3681–3692.
- Churchill, G. (2002). Fundamentals of experimental design for cDNA microarrays. *Nat. Genet.*, **32**: 490–495.
- Cleveland, W. S. (1979). Robust locally weighted regression and smoothing scatterplots. *J. Am. Stat. Assoc.*, **74**(368): 829–836.
- External RNA Controls Consortium (2005). The External RNA Controls Consortium: A progress report. *Nat. Methods*, **2**(10): 731–734.

- Fan, J.-B., et al. (2004). A versatile assay for high-throughput gene expression profiling on universal array matrices. *Genome Res.*, **14**: 878–885.
- Li, C., and Wong, W. H. (2001a). Model-based analysis of oligonucleotide arrays: Model validation, design issues and standard error application. *Genome Biol.*, **2**(8): research0032.1–0032.11.
- Li, C., and Wong, W. H. (2001b). Model-based analysis of oligonucleotide arrays: Expression index computation and outlier detection. *Proc. Natl. Acad. Sci. U.S.A.*, **98**(1): 31–36.
- Lockhart, D. J., et al. (1996). Expression monitoring by hybridization to high-density oligonucleotide arrays. *Nat. Biotechnol.*, **14**: 1675–1680.
- MAQC Consortium (2006). The MicroArray Quality Control (MAQC) project shows inter- and intraplatform reproducibility of gene expression measurements. *Nat. Biotechnol.*, **24**(9): 1151–1161.
- Schena, M., et al. (1995). Quantitative monitoring of gene expression patterns with a complimentary DNA microarray. *Science*, **270**: 467–470.
- Yang, Y. H., et al. (2002). Normalization for cDNA microarray data: A robust composite method addressing single and multiple slide systematic variation. *Nucleic Acids Res.*, **30**(4): e15.

STATISTICAL TESTING AND SIGNIFICANCE FOR LARGE BIOLOGICAL DATA ANALYSIS

Hyung Jun Cho and Wonseok Seo

Department of Statistics, Korea University, Seoul, Korea

This chapter describes statistical tests and significance evaluation for an analysis of large biological data. These data are often generated with a large number of biomarker candidates and a small number of samples, resulting in making any relevant statistical analysis extremely difficult. Furthermore, heterogeneous variability may hinder identifying true positives and negatives in biomarker discovery. It usually takes significant amounts of time and effort to confirm experimentally the identified biomarker candidates. Thus, it is important to strictly control error rates by careful significance evaluation.

4.1 INTRODUCTION

It is fundamental in comparing measurements of two (or more) conditions to discover biomarkers among many candidates with large biological data. For instance, consider the *T-cell immune response data* (Jain et al., 2003), generated from microarray experiments with the Affymetrix MG-U74Av2 chip containing 12,488 probe sets. Triplicate microarrays were used for each of the three populations of immune exposure: *naive* (no exposure), *48-h activated*, and *CD8+ T-cell clone D4* (long-term mild exposure). The primary goal of this microarray study was to discover biomarker genes that were expressed differentially among the conditions. To achieve this goal, differential expression discovery can be performed by a traditional statistical test. However, low replication and large heterogeneous variation can reduce the statistical power for identifying differentially expressed genes. Thus, a number of new testing methods have been developed to obtain more reliable results by accounting for the characteristics of large biological data. The characteristics include the existence of many biomarker candidates, the limited supply of samples, and the heterogeneity of large variability. For differential expression discovery, biological investigators often use a fold change approach. This approach ignores the heterogeneity of variability.

In contrast, most statistical approaches evaluate signal differences relative to noise magnitudes that account for the heterogeneity of variability. We describe various parametric and nonparametric statistics tests, including resampling tests, in Section 4.2. We also introduce new approaches for large biological data.

For biomarker discovery with large biological data, we evaluate expression values of many candidates under two (or more) conditions. Statistically speaking, we perform significance tests for testing many hypotheses. Statistical significance for each hypothesis is evaluated based on its corresponding p -value that arises from applying a statistical test. For example, we claim that candidates having p -values of less than a significance level of 0.05 are significant, that is, differentially expressed between the conditions. However, significance evaluation based on naive p -values may generate many false discoveries. These false discoveries result in wasting a lot of time and effort for confirmation. Thus, it is important to carefully evaluate statistical significance based on appropriately adjusted p -values. The well-known procedures of many error-controlling procedures are introduced in Section 4.3. Various statistical tests and significance evaluation are illustrated with real data in Section 4.4 and programming examples with real data are included in the appendix. Two real datasets for illustration are the microarray data for a T-cell immune response study (Jain et al., 2003) and the mass spectrometry data for a platelet study (Garcia et al., 2005). In the *platelet study data*, *treated* and *untreated* samples were introduced into the same mass spectrometer and this was repeated three times.

4.2 STATISTICAL TESTING

Suppose we have K conditions and n_k samples in the k th condition with $N = n_1 + n_2 + \dots + n_K$. Let Y_{ik} be an indicator of conditions and X_{ijk} be an expression value, where sample $i = 1, 2, \dots, n_k$, candidate $j = 1, 2, \dots, m$, and condition $k = 1, 2, \dots, K$. We assume that the expression values are normalized by an appropriate method (Bolstad et al., 2003; Yang et al., 2002) prior to statistical testing.

4.2.1 Fold Change Approach

A fold change approach is often used by biological investigators to evaluate the degree of differential expression because this approach is simple and straightforward. In fact, this approach is not a statistical test. It defines the ratio of expression values under two conditions. If the ratio is greater than (or less than) a certain value c , for example, $c = 2$ (or $c = 0.5$), then we say the candidate is differentially expressed with a c (or $1/c$) fold change under two conditions. For symmetry, a log transformation (with base 2) is often taken; thus, the judgment is made in the same way when the absolute value of the log ratio is greater than $\log_2(c)$. However, the fold change approach may generate many false positives and false negatives in differential expression discovery with high-throughput data because this approach ignores the magnitude of variability that is heterogeneous in high-throughput data, for instance, as seen in Figure 4.1. We can observe large variation in low-intensity levels and small variation in high-intensity levels within the same condition (naive) in Figure 4.1a. The differential expression

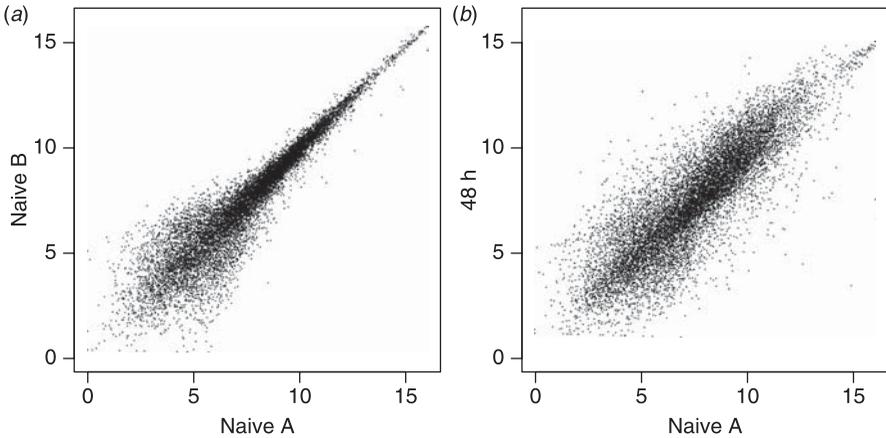


Figure 4.1 Scatter plots (log 2 scale) of (a) two replicated samples from condition naive and (b) two samples from two conditions naive and 48 h activated.

between the different conditions (naive versus 48 h in Figure 4.1b) should be distinguished from the within-condition variation, which is not true by a fold change approach because of ignoring the magnitude of the variation. In contrast, statistical methods allow us to evaluate the difference of signals compared to the magnitude of variability, that is, signal to noise, resulting in the reduction of false decisions.

4.2.2 Parametric Statistical Tests

The *two-sample t-test* (or *Student's t-test*) is the most widely used parametric statistical test. This test compares the means of two populations that should be normally distributed when a sample size is small. The test statistic is formed as the mean difference divided by its standard error, that is, the difference of measured expressions normalized by the magnitude of noises. If the difference of the measured expressions is very large relative to its noise, it is claimed as being significant. Formally, suppose we want to test null hypotheses, $H_j: \mu_{j1} = \mu_{j2}$, against alternative hypotheses, $H_j^a: \mu_{j1} \neq \mu_{j2}$, for $j = 1, 2, \dots, m$. The test statistic for each j is

$$t_j = \frac{\bar{X}_{j1} - \bar{X}_{j2}}{s_j}$$

where

$$s_j = \sqrt{\left(\frac{1}{n_1} + \frac{1}{n_2}\right) \frac{(n_1 - 1)s_{j1}^2 + (n_2 - 1)s_{j2}^2}{n_1 + n_2 - 2}}$$

and

$$s_{jk}^2 = \frac{\sum_{i=1}^{n_{jk}} (X_{ijk} - \bar{X}_{jk})^2}{n_k - 1}$$

is the sample variance for each j under the k th condition. This follows from the t -distribution with a degree of freedom $(n_1 + n_2 - 2)$ when the null hypothesis for j is true. The t -distribution is bell shaped like the normal distribution but thicker in the tail parts. A p -value can be found using a table of values from the t -distribution. For testing hypotheses for m candidates in large biological data, the t -test is conducted and a p -value is obtained for each candidate. Then we claim that the candidates with p -values less than a prespecified significance level, for example, $\alpha = 0.05$, are statistically significant. In fact, we assume that the variances for two conditions are equal. If the variances are not equal, a slightly modified t -test (called *Welch's t -test*) is used. The test statistic for each j is

$$t_j = \frac{\bar{X}_{j1} - \bar{X}_{j2}}{s_j}$$

where

$$s_j = \sqrt{\frac{s_{j1}^2}{n_1} + \frac{s_{j2}^2}{n_2}}$$

The degree of freedom is $(s_{j1}^2/n_1 + s_{j2}^2/n_2)^2 / [(s_{j1}^2/n_1)^2/(n_1 - 1) + (s_{j2}^2/n_2)^2/(n_2 - 1)]$. For example, the gene expression values are 12.79, 12.53, and 12.46 for the naive condition and 11.12, 10.77, and 11.38 for the 48-h activated condition from the T-cell immune response data. The sample sizes are $n_1 = n_2 = 3$. The sample means are 12.60 and 11.09 and the sample variances are 0.0299 and 0.0937, resulting in a pooled variance of $(0.2029)^2$. The t -statistic is $(12.60 - 11.09)/0.2029 = 7.44$ and the degree of freedom is $n_1 + n_2 - 2 = 3 + 3 - 2 = 4$. Then we can find a p -value of 0.003. If using *Welch's t -test*, the t -statistic is still 7.42 since $n_1 = n_2$, but we find the p -value of 0.0055 since the degree of freedom is 3.364 rather than 4. We claim that the probe set is differentially expressed under the two conditions because its p -value is less than a predetermined significance level (e.g., 0.05). In this manner, p -values for the other probe sets can be calculated and interpreted. In Section 4.4, the overall interpretation for p -values of all of the probe sets is described with adjustments for multiple testing. The Student's t -test and *Welch's t -test* are used for samples drawn independently from two conditions. When samples from the two conditions are paired, a different version called the *paired t -test* is more appropriate than independent t -tests:

$$t_j = \frac{\bar{X}_{jD}}{s_{jD}/\sqrt{n}}$$

where \bar{X}_{jD} and s_{jD} are the mean difference and the standard deviation of the differences and $n = n_1 = n_2$. Its corresponding p -value can be obtained from the table of t -distributions with the degree of freedom $n - 1$. For instance, three pairs of the treated and untreated samples from the platelet study data are (24.31, 25.36), (23.77, 24.46), and (24.26, 24.63) and their differences are -1.05 , -0.69 , and -0.37 . If the mean

and standard deviation are applied, its paired t -test statistic becomes -3.58 and its p -value becomes 0.07 . The peptide is not significant since its p -value of 0.07 is greater than 0.05 .

We may want to compare the means of populations from more than two conditions simultaneously, for example, naive (no exposure), 48 h activated, and CD8+ T-cell clone D4 (long-term mild exposure) in the T-cell immune response data. That is, null hypotheses are $H_j: \mu_{j1} = \mu_{j2} = \dots = \mu_{jK}$ and alternative hypotheses are $H_j^a: \text{not } H_j$ for $j = 1, 2, \dots, m$. For testing hypotheses for m candidates, an analysis of variance (ANOVA) needs to be conducted. The ANOVA F -statistic is the between-condition variation divided by the within-condition variation. Thus, we can claim that the probe set is differentially expressed under the K conditions if the F -statistic is large enough. Formally, the ANOVA F -statistic follows an F -distribution with the degrees of freedom $K - 1$ and $N - 1$. Thus, the “large enough” indicates that the calculated p -value is less than the significance level given. For example, the ANOVA F -statistic for a probe set is 118.34 with the degrees of freedom 2 and 6 . The p -value is less than 0.05 . Thus, we can claim that the probe set is differentially expressed under the three conditions. The overall interpretation for p -values of all of the probe sets is also described in Section 4.4.

4.2.3 Nonparametric Statistical Tests

We often assume that data are drawn from a normal distribution. When the normality assumption is not met, alternatively, a nonparametric test is used. Nonparametric tests do not rely on assumptions that the data are drawn from a given probability distribution. However, it is less efficient if it is used when the distribution assumption is met.

The *Wilcoxon rank-sum test* is a nonparametric test for assessing whether two samples of measurements come from the same distribution. That is, as an alternative to the two-sample t -test, this test can be used to discover differentially expressed candidates under two conditions. For example, again consider the measurements of the probe set used for the two-sample t -test. The gene expression values are 12.79 , 12.53 , and 12.46 for the naive condition and 11.12 , 10.77 , and 11.38 for the 48-h activated condition. Measurement 12.79 has rank 6 , measurement 12.53 has 5 , and measurement 12.46 has rank 4 . The rank sum of the naive condition is $6 + 5 + 4 = 15$. Then after the sum is subtracted by $n_1(n_1 + 1)/2 = 3 \times 4/2 = 6$, the Wilcoxon rank-sum test statistic becomes 9 . Considering all of the combinations of the three measurements, we can compute the probability that the rank sum happens more extremely than 9 . The probability becomes its p -value. This is the most extreme among the 20 combinations; thus the p -value is $2 \times \Pr(W \geq 9) = 2 \times \frac{1}{20} = 0.1$ for the two-sided test. It is hard to say that the probe set is differentially expressed since the p -value $0.1 > 0.05$. This test is also called the *Mann–Whitney–Wilcoxon test* because this test was proposed initially by Wilcoxon for equal sample sizes and extended to arbitrary sample sizes by Mann and Whitney. As a nonparametric alternative to the paired t -test for the two related samples, the *Wilcoxon signed-rank test* can be used. The statistic is computed by ordering absolute values of differences of paired samples. For example, consider a peptide in the platelet study data. Their differences for each

pair are 0.224, 0.109, and -0.039 . The ranks of the absolute values are 3, 2, and 1; thus, the test statistic is $3 + 2 = 5$ if summing up the ranks of positive differences. Its p -value can be obtained by comparing the test statistic with all of the possible combinations of signs and ranks. For instance, all of the combinations are 6, 5, 4, 3, 3, 2, 1, and 0. Among the eight combinations, 6 and 5 are more extreme than or equal to 5; thus, the p -value is $\frac{2}{8} = 0.25$ for the one-sided and $\frac{2}{8} \times 2 = 0.50$ for the two-sided test. Like the paired t -test, the p -value involves comparisons of differences between paired measurements. However, this test does not require assumptions about the form of the distribution of the measurements.

The *Kruskal–Wallis test* is a nonparametric alternative to the ANOVA F -test, described above, for multiple conditions. That is, it is an extension of the Wilcoxon rank-sum test to multiple conditions. Expression values are replaced with their ranks to form the test statistic without requiring an assumption of the form of the distribution. For example, the Kruskal–Wallis statistic is 7.2 and the p -value is less than 0.05 for the probe set used for illustration of the ANOVA F -test.

4.2.4 Resampling-Based Tests

In the case that the joint or marginal distribution of the test statistics is unknown, p -values can be estimated by resampling methods such as permutation and bootstrap. For example, consider a permutation algorithm to estimate p -values with large biological data in the following manner. First, permute the N sample columns of the data matrix and compute test statistics for each biomarker candidate. Let $t_{1,b}, \dots, t_{m,b}$ be test statistics for the b th permutation. When repeating this procedure many times (e.g., $B = 100$ times), the permutation p -value for hypothesis H_j is

$$p_j^* = \frac{\sum_{b=1}^B I(|t_{j,b}| \geq |t_j|)}{B}$$

where $I(\cdot)$ is unity if the condition in parentheses is true and zero otherwise. It can be one of the nonparametric tests where parametric distributions for the test statistics are not specified. However, it may be infeasible to consider all possible permutations for large N , whereas there are too small numbers of distinct permutations for small N . Bootstrap resampling differs from permutations in that it draws samples with replacement. It is more appropriate for certain situations such as when two conditions have different variances because it can preserve the covariance structure (Pollard and van der Laan, 2005).

4.2.5 Ad Hoc Tests

A number of ad hoc methods to analyze large biological data have been developed. They utilize the characteristics of the large biological data. For analyzing large data from microarray experiments, *SAM (significance analysis of microarrays)* (Tusher et al., 2001), newly developed, has been widely used. In fact, it resembles the

two-sample t -test statistic. A difference is to add a fudge factor to the variance estimate to have similar distributions of test statistics for all candidates:

$$d_j = \frac{\bar{X}_{j1} - \bar{X}_{j2}}{s_j + s_0}$$

where s_0 is a fudge factor added to compare d -statistics for all candidates. A probe set is declared as differentially expressed if its observed d -score is greater than its expected d -score plus a critical value. Unlike t -tests calculating p -values for measuring the degrees of differential expression, the SAM d -test calculates resampling-based false discovery rates. The details for false discovery rates are described in the next section.

In high-throughput experiments, replicates are often limited because of the limited supply of biological samples and the high cost of experiments. Thus, experiments are conducted with very small samples. With small samples, standard tests are often underpowered. It follows that many false positives and negatives are generated. To improve the statistical power in such cases, ad hoc methods have been developed. It is commonly observed that the variability of large biological data is heterogeneous and the magnitude of the heterogeneity depends on the intensity levels nonlinearly. Thus, the *LPE (local pooled error) test* (Jain et al., 2003) pools genes of similar expression levels so as to estimate underlying variances with small samples, resulting in an increase of statistical power. This test can be used as an alternative to the two-sample t -test when the sample size is very small, for example, duplicates or triplicates. Like SAM, the degree of the differential expression can be evaluated by a resampling-based false discovery rate. The *PLPE (paired local pooled error) test* (Cho et al., 2007) is an alternative to the paired t -test when the sample size is very small. For multiple conditions with small samples, *HEM (hierarchical error model)* (Cho and Lee 2004, 2008) can be used, particularly as enhanced with an empirical Bayes technique.

Another widely used and well-developed method is the linear modeling approach enhanced with an empirical Bayes technique (Smyth, 2004). This approach (called *limma*) was designed to analyze microarray data from various designed experiments by fitting linear models. The enhancement of an empirical Bayes technique results in more stable inference and improved power, especially for experiments with small-sample microarray data.

The above ad hoc methods are illustrated with real data in Section 4.4. In addition to the methods described above, many other methods have been developed; some of them have been compared to provide guidelines in these references (Dudoit et al., 2002; Kim et al., 2006; Kooperberg et al., 2005). Scores computed from the above test can also be used for gene set enrichment analysis (GSEA) (Efron and Tibshirani, 2007; Jiang and Gentleman, 2007; Subramanian et al., 2005; Tian et al., 2005), which analyzes the data of a predefined set of genes.

4.2.6 Applying Testing Methods in R

Statistical tests can be conducted by usual statistical software or any software developed for large biological data. In particular, many useful R packages for

bioinformatics are freely available at the Bioconductor project (<http://www.bioconductor.org>). Most of the above testing methods can be conducted by intrinsic or customized packages, sometimes with some programming, in R. For example, the above parametric and nonparametric tests can be performed without calling any other libraries in R, but some programming is needed to perform multiple testing for tens of thousands of candidates. The ad hoc tests require the specific packages such as *samr*, *limma*, *LPE*, *PLPE*, and *HEM*. These tests are illustrated with real data in Section 4.4 and programming examples are available in the appendix.

4.3 ERROR CONTROLLING

It is essential to strictly control error rates by careful significance evaluation to save the time and effort of experiments for confirming biomarker candidates selected. Error rates can be controlled more carefully and strictly by using multiple testing adjusted *p*-values rather than raw *p*-values.

4.3.1 Sensitivity, Specificity, PPV, NPV, and ROC Curve

When diagnosing a disease, there are two correct diagnoses: (1) a positive test result when a subject has a disease and (2) a negative test result when a subject does not have a disease (Table 4.1). Given that the subject has the disease, the conditional probability, $\Pr(+|+)$, that the diagnostic test is positive is called the *sensitivity*, whereas given that the subject does not have the disease, the conditional probability, $\Pr(-|-)$, that the diagnostic test is negative is called the *specificity* (Agresti, 2002). The probabilities can be expressed as the false-positive rate (FPR) and the false-negative rate (FNR), that is, $\text{FPR} = 1 - \text{specificity}$ and $\text{FNR} = 1 - \text{sensitivity}$. It is ideal to increase both probabilities; however, the increase of one probability causes the decrease of the other with the given information. That is, changing the cutoff point of the test statistic increases one probability and decreases the other. In other words, it increases (decreases) false positives and decreases (increases) false negatives. Practical performance evaluation on a test can also often be evaluated by *positive predictive value* (PPV), the proportion of true positives among all positively called (or significantly identified) candidates at a specific cutoff value, and *negative predictive value* (NPV), the proportion of true false negatives among all negatively called (or excluded) candidates at a specific cutoff value.

TABLE 4.1 Diagnosis Table

Disease	Test result	
	Negative (-)	Positive (+)
Negative (-)	Correct	False positive
Positive (+)	False negative	Correct

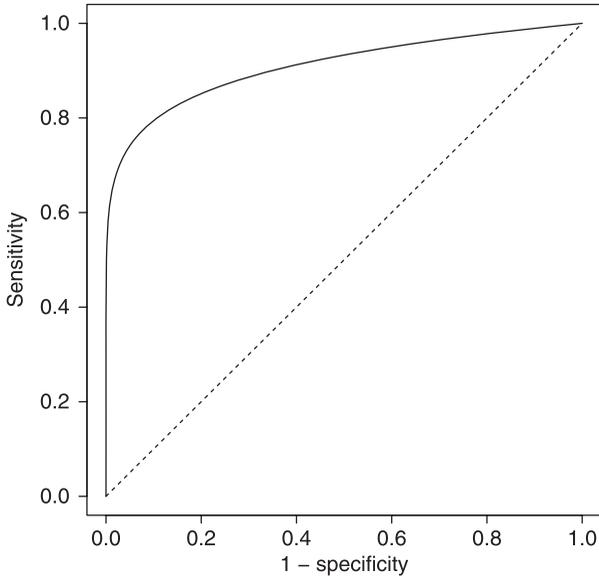


Figure 4.2 ROC curve; the solid curve is an example of a good decision and the dotted curve is an example of a random decision.

Changing the threshold of the test statistic generates various combinations of sensitivity and specificity (or FPR and FNR). The changes can be displayed as a plot of sensitivity against $(1 - \text{specificity})$ for possible cutoff points. This plot is called a *receiver operating characteristic (ROC) curve*. Random decision making shows a 45° line and a good method shows a line close to the top-left corner resulting in an *area under the curve (AUC)* of close to 1. Testing methods can be compared by drawing ROC curves on the same plot and calculating their AUCs (Fig. 4.2).

4.3.2 Multiple-Testing Problem

A type I error is the case that rejects the null hypothesis when it is true; whereas a type II error is the case that does not reject the null hypothesis when it is not true (Table 4.2). The two types of errors correspond to false positives and false negatives in Table 4.1. We typically want to maximize the power ($= 1 - \text{the type II error}$) while controlling the type I error less than or equal to a predetermined significance level α .

TABLE 4.2 Hypothesis Testing

Hypothesis	Decision about null hypothesis	
	Not rejected	Rejected
Null	Correct	Type I error
Alternative	Type II error	Correct

TABLE 4.3 Summary Table for the Multiple Testing Problem

Hypothesis	Decision about null hypothesis		Total
	Not rejected	Rejected	
Null	$m_0 - V$	V	m_0
Alternative	$m_1 - S$	S	m_1
Total	$m - R$	R	m

We consider the problem of testing simultaneously m null hypotheses H_j ; no differential expression against H_j^a : differential expression, where $j = 1, \dots, m$. A naive procedure is to compute a test statistic T_j and a p -value p_j for each candidate j . Then, select the candidates having p -values less than a significance level α and claim them as differentially expressed under the given conditions. In fact, however, this approach does not control the type I error under a significance level α . It may generate more than expected false positives. Before making a decision with the raw p -values in the naive procedure, it is essential to apply a multiple-testing procedure to control the defined type I error (Dudoit et al., 2003). Let us summarize the situation as in Table 4.3 according to Benjamini and Hochberg (1995). Suppose there exist m_0 and m_1 indifferent and differential expressed hypotheses for m candidates; however, m_0 and m_1 are unknown. By choosing the threshold of the test statistics, $m - R$ and R are determined. We are mainly interested in controlling the number (V) of false positives. Thus, the type I error rates can be defined as follows:

- *Per-family error rate (PFER)* = $E(V)$
- *Per-comparison error rate (PCER)* = $E(V)/m$
- *Familywise error rate (FWER)* = $\Pr(V \geq 1)$
- *False discovery rate (FDR)* = $E(Q)$, where $Q = V/R$ if $R > 0$ and $Q = 0$ if $R = 0$.

The PFER is the expected number of false positives and PCER is PFER divided by the number of all of the hypotheses. The FWER is the probability that the number of false positives is greater than or equal to one. The FDR is the expected proportion of false positives among the rejected hypotheses. The relationships among the error rates are $\text{PCER} \leq \text{FWER} \leq \text{PFER}$ and $\text{FDR} \leq \text{FWER}$. If all of the null hypotheses are true, that is, $m = m_0$, then $\text{FDR} = \text{FWER}$. In practice, FWER is too conservative, so FDR is often controlled, instead. Note that $\text{FDR} = 1 - \text{PPV}$.

4.3.3 Multiple-Testing Procedures

A number of procedures for controlling error rates have been developed to solve the multiple-testing problem (Dudoit et al., 2003). The *Bonferroni* procedure for controlling the FWER at level α rejects any hypothesis H_j with unadjusted p -value

less than or equal to α/m . That is, the Bonferroni adjusted p -values are given by mp_j , $j = 1, \dots, m$. However, the Bonferroni procedure is very conservative. To be less conservative, *Holm* proposed a step-down procedure (Holm, 1979), valid under an arbitrary assumption. When test statistics are independent or nonnegatively associated, *Hochberg's* and *Hommel's* procedures (Hochberg, 1988; Hommel, 1988) are also valid. Hommel's is slightly more powerful; Hochberg's is faster. *Sidak's* single-step or step-down procedures (Sidak, 1967) also control FWER strongly. However, FWER control procedures are still more conservative than FDR control procedures.

The *Benjamini–Hochberg (BH)* procedure (Benjamini and Hochberg, 1995) controls the FDR when test statistics are independent of or dependent on a certain structure. The *Benjamini–Yekutieli (BY)* procedure (Benjamini and Yekutieli, 2001) was developed for arbitrary dependence structures. The q -value, defined as the minimum FDR level, measures the proportion of false positives incurred when the particular test is called significant (Storey, 2002). Thus, H_j is rejected if q_j is smaller than or equal to a prespecified FDR level. The *SAM's* estimate of FDR (Tusher et al., 2001) is obtained by resampling expression measurements; however, it was pointed out that the SAM estimate controls $E(V | \text{all true null hypotheses})/R$ rather than $E(V | R)$ because the cutoff depends on the test statistics (Dudoit et al., 2003), that is, the control of PFER rather than FDR. The LPE and HEM proposed FDR estimates similar to the SAM approach; however, the LPE and HEM tests differ from FDR in that they utilize different permutations to retain a heterogeneous distribution with small samples. In addition, many procedures have been developed and compared. The details for them can be found in the paper of Dudoit et al. (2003).

4.3.4 Applying Multiple-Testing Procedures in R

A number of multiple-testing procedures have been implemented in R or other computer programs. In R, if we have computed raw p -values from a testing method, we can compute adjusted p -values for controlling FWER or FDR by using *mt.rawp2adjp* (in the *multtest* package) or *p.adjust*. The functions contain the above Bonferroni, Holm, Sidak, Hochberg, Hommel, Benjamini–Hochberg, and Benjamini–Yekutieli's procedures. Q -values can be calculated by the *qvalue* package and the FDR estimates of SAM, LPE, and HEM can be obtained by their own packages (*samr*, *LPE*, and *HEM*).

4.4 REAL DATA ANALYSIS

In this section, we illustrate various testing methods with multiple-testing procedures with two real data sets: T-cell immune response data and platelet study data. The former from microarray experiments is used for comparing two or multiple samples and the latter for paired samples. Programming examples for these data analyses can be found in the appendix. Parts of the results are displayed in Tables 4.4–4.6.

TABLE 4.4 Results of Various Tests for Identifying Genes Expressed Differentially Under Naive and 48 h Activated of T-Cell Immune Response Data

GeneID	LogFC	t-stat	raw.p.value	Bonf. adj.p	BH.adj.p. value	By.adj.p. value	W-stat	raw.p. value	Bonf. adj.p
100001_at	1.491126	7.419651	0.004243	1	0.032345	0.323768	9	0.1	1
100002_at	0.408612	0.332466	0.765835	1	0.866989	1	4	1	1
100003_at	-1.53996	-1.93721	0.170102	1	0.343393	1	1	0.2	1
100004_at	-1.52292	-4.13618	0.018018	1	0.07648	0.765546	0	0.1	1
100005_at	-0.5991	-3.30293	0.077565	1	0.203623	1	0	0.1	1
100006_at	-0.91671	-1.39571	0.275856	1	0.467991	1	1	0.2	1
100007_at	0.612931	5.767166	0.005448	1	0.037361	0.373971	9	0.1	1
100009_r_at	1.90881	-3.32861	0.038285	1	0.126117	1	0	0.1	1
100010_at	1.954834	20.25254	6.71E-05	0.837834	0.00449	0.044947	9	0.1	1
100011_at	3.290441	29.59833	0.000446	1	0.010508	0.105183	9	0.1	1
100012_at	1.35557	22.24261	8.78E-05	1	0.005092	0.050969	9	0.1	1
100013_at	0.820256	13.12724	0.000516	1	0.011305	0.113165	9	0.1	1
100014_at	-0.39717	-2.99802	0.080374	1	0.20844	1	0	0.1	1
100015_at	0.221187	0.102531	0.923834	1	0.960586	1	4	1	1
100016_at	0.418189	0.614328	0.593809	1	0.747001	1	5	1	1
100017_at	0.723817	1.733091	0.163281	1	0.334611	1	8	0.2	1
100018_at	-0.60184	-2.8563	0.07224	1	0.194019	1	0	0.1	1
100019_at	-0.22255	-1.51011	0.26728	1	0.458992	1	2	0.4	1
100020_at	0.691001	4.48138	0.011445	1	0.057779	0.578355	9	0.1	1
100021_at	-0.56709	-0.67183	0.568035	1	0.727998	1	4	1	1

TABLE 4.5 Results of Various Tests for Identifying Genes Expressed Differentially Under All Three Conditions of T-Cell Immune Response Data

GeneID	F-stat	raw.p. value	Bonf.adj.p	BH.adj.p. value	BY.adj.p. value	KW-stat	raw.p. value
100001_at	118.3399	1.51E-05	0.188732	0.000186	0.001863	7.2	0.027324
100002_at	0.248151	0.787871	1	0.855387	1	0.088889	0.956529
100003_at	4.265597	0.070396	1	0.141041	1	4.355556	0.113293
100004_at	18.96915	0.002546	1	0.009156	0.091651	7.2	0.027324
100005_at	20.82113	0.001997	1	0.007509	0.075161	6.0488889	0.03899
100006_at	1.017293	0.41645	1	0.549691	1	3.288889	0.19312
100007_at	31.05052	0.000684	1	0.003204	0.032066	5.422222	0.066463
100009_r_at	5.87705	0.038597	1	0.086272	0.863568	5.6	0.06081
100010_at	68.47535	7.39E-05	0.923396	0.000576	0.00577	7.2	0.027324
100011_at	207.2585	2.90E-06	0.036274	6.33E-05	0.000634	7.2	0.027324
100012_at	458.8303	2.74E-07	0.003423	1.50E-05	0.00015	7.2	0.027324
100013_at	961.159	3.01E-08	0.000376	5.24E-06	5.25E-05	7.2	0.027324
100014_at	30.49875	0.000718	1	0.003316	0.033192	7.2	0.027324
100015_at	2.103878	0.203078	1	0.324426	1	4.266667	0.118442
100016_at	2.557048	0.157337	1	0.266815	1	3.466667	0.176694
100017_at	1.854518	0.236007	1	0.364653	1	2.222222	0.329193
100018_at	9.43985	0.014025	1	0.037353	0.3739	5.422222	0.066463
100019_at	0.755124	0.509907	1	0.633793	1	1.155556	0.561144
100020_at	6.321153	0.033339	1	0.07651	0.765844	5.6	0.06081
100021_at	0.143248	0.869416	1	0.912198	1	0.266667	0.875173

BH.adj.p. value	By.adj. p.value	limma-stat	raw.p.value	BH.adj.p. value	BY.adj.p. value	SAM-stat	q.value.SA	LPE-stat	FDR.LPE
0.222841	1	7.231542	0.000468	0.004637	0.046415	0.672451	5.124669	7.493136	1.60E-05
1	1	0.384222	0.714881	0.820385	1	0.075535	47.3572	-0.03378	0.584745
0.365413	1	-2.28205	0.065401	0.155538	1	-0.68616	4.100811	-2.90726	0.006347
0.222841	1	-3.26324	0.018781	0.06103	0.610898	-0.24022	26.4977	-1.696	0.080201
0.222841	1	-3.13854	0.021844	0.068163	0.682292	-0.2798	22.25325	-2.59885	0.01357
0.365413	1	-1.63715	0.155866	0.292304	1	-0.42724	10.56808	-1.03233	0.229192
0.222841	1	4.109956	0.007176	0.030033	0.300619	0.285263	22.25325	2.946796	0.005449
0.222841	1	-3.82569	0.009791	0.03739	0.374263	-0.70206	4.100811	-1.62857	0.091043
0.222841	1	13.58108	1.60E-05	0.000876	0.008772	0.917622	1.552454	6.365339	1.60E-05
0.222841	1	21.77052	1.17E-06	0.000315	0.003153	1.535671	0	11.97212	1.60E-05
0.222841	1	10.47406	6.59E-05	0.001519	0.015205	0.647007	5.124669	8.216354	1.60E-05
0.222841	1	6.300843	0.000945	0.007221	0.072279	0.390379	15.57186	4.010384	0.000177
0.222841	1	-2.47029	0.050995	0.129358	1	-0.18782	31.91043	-1.09679	0.209597
1	1	0.11978	0.908809	0.94831	1	0.028309	57.41233	-0.37948	0.454446
1	1	0.697714	0.513077	0.662253	1	0.113803	42.54265	0.040507	0.582324
0.365413	1	1.956769	0.101206	0.214287	1	0.304348	22.25325	0.756031	0.318164
0.222841	1	-2.85023	0.031254	0.089231	0.893182	-0.27733	22.25325	-2.01718	0.043749
0.586429	1	-1.33892	0.232036	0.387491	1	-0.10908	47.3572	-0.20918	0.520154
0.22841	1	3.91936	0.008826	0.034508	0.345418	0.314679	18.59394	3.463804	0.001213
1	1	-0.77195	0.471243	0.625785	1	-0.14123	42.54265	-0.04497	0.580844

Bonf.adj. p.value	BH.adj. p.value	BY.adj. p.value	SAM-stat	q.value.SA	HEM-stat	FDR.HEM
1	0.132926	1	0.69559	0.932983	126.6247	8.01E-05
1	0.959294	1	0.130161	34.81845	0.048904	0.440157
1	0.198848	1	0.562591	2.65447	1.497035	0.021475
1	0.132926	1	0.23468	26.19212	1.849668	0.013285
1	0.132926	1	0.284405	21.90356	5.992851	0.000824
1	0.299327	1	0.273225	21.90356	0.291386	0.192165
1	0.132926	1	0.198447	30.52267	2.393575	0.007644
1	0.132926	1	0.524952	3.915465	0.447117	0.131643
1	0.132926	1	1.100358	0	22.96196	8.01E-05
1	0.132926	1	1.241934	0	40.0284	8.01E-05
1	0.132926	1	0.472697	5.801495	48.41095	8.01E-05
1	0.132926	1	0.715374	0.498186	65.15769	8.01E-05
1	0.132926	1	0.244487	21.90356	2.512736	0.006454
1	0.205345	1	0.29053	17.79125	1.310282	0.027697
1	0.284791	1	0.221216	26.19212	0.291543	0.192121
1	0.447525	1	0.186643	30.52267	0.215139	0.237615
1	0.132926	1	0.220727	26.19212	1.292783	0.02798
1	0.656446	1	0.069811	45.45318	0.008738	0.516325
1	0.132926	1	0.199039	30.52267	1.718641	0.015653
1	0.900187	1	0.089401	42.53004	0.020688	0.492047

TABLE 4.6 Results of Various Tests for Identifying Peptides Expressed Differentially Under Treated and Untreated Conditions of Mass Spectrometry Data for Platelet Study

PeptideID	LogFC	paired t-stat	raw.p.value	Bonf.adj.p	BH.adj. p.value	BY.adj. p.value	W- stat	raw.p. value	Bonf. adj.p
1	-0.6761	-3.54382	0.071224	1	0.294393	1	0	0.25	1
2	0.102027	1.289399	0.326255	1	0.527858	1	5	0.5	1
3	0.073851	0.145148	0.897901	1	0.94236	1	4	0.75	1
4	0.382872	1.185507	0.357581	1	0.527858	1	5	0.5	1
5	1.189838	2.603166	0.121297	1	0.412695	1	6	0.25	1
6	0.29284	2.191591	0.159753	1	0.412695	1	6	0.25	1
7	-0.04072	-0.3566	0.7555	1	0.851655	1	3	1	1
8	0.998802	2.841334	0.104761	1	0.386621	1	6	0.25	1
9	-1.07529	-15.8456	0.003959	0.245465	0.081822	0.385575	0	0.25	1
10	0.161881	4.120679	0.054153	1	0.294393	1	6	0.25	1
11	0.392001	1.728905	0.225967	1	0.500355	1	6	0.25	1
12	0.41543	0.64471	0.585192	1	0.740447	1	4	0.75	1
13	0.097715	0.618109	0.599512	1	0.743395	1	4	0.75	1
14	0.108058	-0.30218	0.791046	1	0.872543	1	3	1	1
15	0.174397	3.811808	0.062446	1	0.294393	1	6	0.25	1
16	-0.28801	-0.87075	0.475699	1	0.655408	1	2	0.75	1
17	1.398664	0.918388	0.455367	1	0.641653	1	4	0.75	1
18	0.810744	0.46849	0.685534	1	0.801945	1	4	0.75	1
19	0.538994	2.231861	0.155301	1	0.412695	1	6	0.25	1
20	0.322821	1.415501	0.292572	1	0.527858	1	6	0.25	1

4.4.1 Two Samples: T-Cell Immune Response Data

To illustrate the comparison of two independent samples, we first consider only the two conditions naive and 48 h activated of the T-cell immune response data. That is, we identify genes differentially expressed under the two conditions. Using the R code in the appendix, we conducted various tests described in the previous sections and obtained the results. Table 4.4 displays part of the results containing test statistics and p -values for each probe set. For example, consider the first probe set 100001_at. Its log-fold change value is 1.49, so the gene was expressed $2^{1.49} = 2.8$ times higher under the naive condition than under the 48-h activated condition. The t -statistic is 7.42 and its corresponding p -value is 0.0042, which is less than a significance level, for example, 0.05. By this, therefore, we can claim that the gene is differentially expressed under the two conditions. However, we tested 12,488 hypotheses; thus, we need to make an adjustment to control the type I error under the significance level, as described in Section 4.3. Its Bonferroni adjusted p -value is 1, indicating that it is not significant. Overall, by the raw p -values, 4130 probe sets are significant with level 0.05, but only 21 probe sets are significant after the Bonferroni adjustment. By the BH and BY procedures, 2239 and 212 probe sets are significant. The results support the Bonferroni procedure as being very conservative and the raw p -value as being very liberal.

By applying the Wilcoxon rank-sum test, we obtained a test statistic of 9 and a p -value of 0.1 for 100001_at, implying the insignificance of the gene. The same conclusion was reached by the adjustments. By applying limma, we obtained a test statistic

BH.adj. p.value	BY.adj. p.value	limma-stat	raw.p. value	BH.adj. p.value	BY.adj. p.value	SAM-stat	q.value. SA	PLPE-stat	FDR.PLPE
0.455882	1	-3.23452	0.024715	0.167248	0.788139	-2.87089	13.66224	-1.5193	0.203704
0.72093	1	0.538314	0.614521	0.762006	1	0.800056	13.66224	0.240161	1.018519
0.845455	1	0.148152	0.888282	0.933449	1	0.121464	13.66224	0.037647	1.018519
0.72093	1	1.31309	0.248783	0.416879	1	1.015993	13.66224	0.511923	1.018519
0.455882	1	3.367109	0.02146	0.167248	0.788139	2.340983	5.806452	1.42458	0.339506
0.455882	1	1.45145	0.209098	0.392851	1	1.609363	9.677419	0.531075	1.018519
1	1	-0.41181	0.69833	0.801786	1	-0.30942	13.66224	-0.52156	1.018519
0.455882	1	3.376084	0.021258	0.167248	0.788139	2.486814	5.806452	0.952668	1.018519
0.455882	1	-5.96558	0.002226	0.06902	0.325251	-9.41374	0	-1.54983	0.25463
0.455882	1	0.907911	0.407465	0.587508	1	1.880323	9.677419	0.278999	1.018519
0.455882	1	1.631242	0.166592	0.344291	1	1.414203	11.79435	0.330869	1.018519
0.845455	1	0.844105	0.438891	0.604694	1	0.582186	13.66224	0.047737	1.018519
0.845455	1	0.432077	0.684521	0.80076	1	0.461854	13.66224	0.251112	1.018519
1	1	-0.43904	0.679812	0.80076	1	-0.28458	13.66224	0.112173	1.018519
0.455882	1	0.973014	0.377278	0.556935	1	1.8826	9.677419	0.418721	1.018519
0.845455	1	-1.23074	0.275647	0.436691	1	-0.80958	13.66224	-0.68857	1.018519
0.845455	1	1.356429	0.235642	0.416879	1	0.872672	13.66224	0.097685	1.018519
0.845455	1	0.689088	0.52285	0.685993	1	0.444115	13.66224	0.927084	1.018519
0.455882	1	2.209999	0.080724	0.263415	1	1.851353	9.677419	0.587816	1.018519
0.455882	1	1.324922	0.245129	0.416879	1	1.155163	13.66224	0.143309	1.018519

of 7.23 and (un)adjusted p -values less than 0.05. The SAM procedure produced a test statistic of 0.67 and a q -value of 5.09%; the LPE procedure produced a test statistic of 7.49 and an extremely small FDR. If using a significance level of 0.05, 100001_at is significant by the raw and BH adjusted p -values of the Welch's t -test, all p -values of limma, and the FDR of LPE, while not significant by the others. In total, no gene was significant by the nonparametric procedure, whereas 1436 and 4534 genes were significant by using SAM and LPE, respectively. These data consist of three replicates for each condition, so limma and LPE may be more appropriate than the others. For comparison, log(fold changes) and various statistics are plotted in Figure 4.3.

4.4.2 Multiple Samples: T-Cell Immune Response Data

Sometimes, we may be interested in comparing more than two conditions simultaneously. The T-cell immune response data contains three conditions: naive, 48 h activated, and D8 clone. The genes identified under the three conditions were expressed differentially under at least one from the others because of rejecting their corresponding hypotheses that the mean expressions of all the conditions were the same. For instance, consider the first probe set 100001_at in Table 4.5. The F -statistic was 118.34 and its raw p -value was very small, rejecting the hypothesis that the gene was similarly expressed under the three conditions. By the Bonferroni procedure, the gene was insignificant because the p -value was adjusted as 0.189, whereas by the BH and BY procedures it was claimed as significant because of the adjusted p -values of

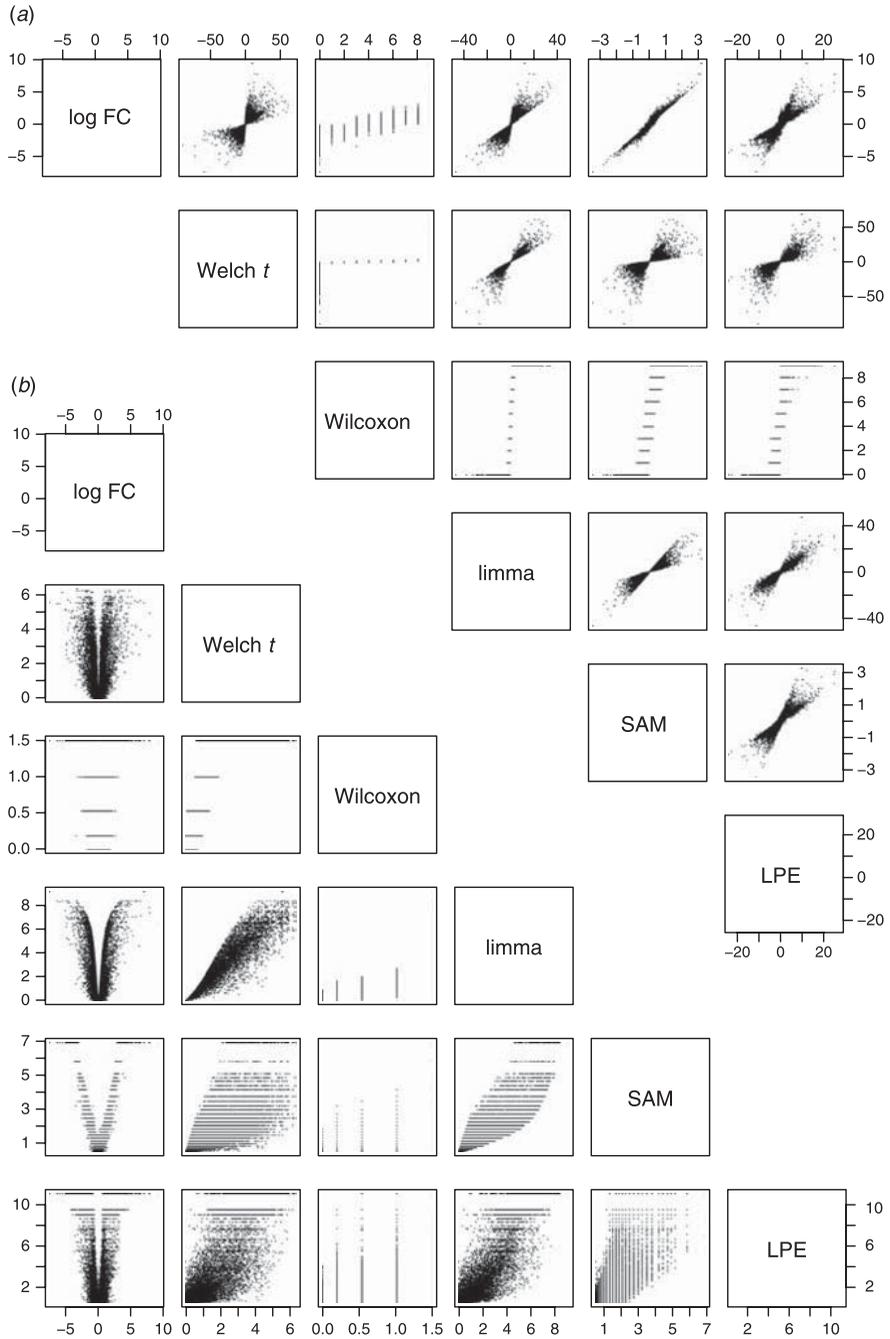


Figure 4.3 Scatter plots of test statistics and p -values from various tests; the upper panel displays log FC and test statistics and the lower panel displays log FC and $-\log(p\text{-values})$. BH-adjusted p -values were used for Welch *t*, Wilcoxon, and limma.

0.000186 and 0.001863. By the Kruskal–Wallis test, the gene was not significant with the adjustment, whereas by SAM and HEM it was very significant. In total, 5849, 654, 4969, and 3017 genes were significant with a significance level of 0.05 by the raw p -value and Bonferroni, BH, and BY adjusted p -values of F -statistics. By the Kruskal–Wallis test, 3061 genes were significant by the raw p -value, but no gene was significant by the adjusted ones. By SAM and HEM, 1727 and 5061 genes were significant, respectively.

4.4.3 Paired Samples: Platelet Study Data

To illustrate the comparison of paired samples, we consider mass spectrometry data for a platelet study. These data consisted of 62 peptides with three replicates of two paired samples: treated and untreated. Various methods were applied to the data. The results are displayed in Table 4.6. For instance, consider peptide 9. The absolute value of the log (fold change) value was about 1, that is, a twofold change between the two conditions. The raw p -values of the paired t -test and limma indicated that the peptide was statistically significant with a significance level of 0.05, that is, the difference in intensities between the two conditions. In contrast, it was not significant by the adjusted p -values of the paired t -test and limma. It was not significant by all the raw and adjusted p -values of the Wilcoxon signed-rank test. By SAM and PLPE, it was very significant since their q -value and FDR were very small. Overall, no peptide was significant by the paired t -test and the Wilcoxon signed-rank test except for the raw p -value of the paired t -test. By the raw, BH-adjusted, and BY-adjusted p -values of limma, 14, 1, and 0 peptides were significant, respectively. On the other hand, 10 and 4 peptides were significant by SAM and PLPE, respectively. By the LogFC, five peptides had two or larger changes. Among the five peptides, peptides 9 and 58 were also significant by SAM and PLPE.

4.5 CONCLUDING REMARKS

We described traditional statistical and newly developed tests, including error-controlling procedures, for biomarker discovery with large biological data. In addition to the tests and procedures, many others have been proposed. Some of them have been compared (Dudoit et al., 2002, 2003; Kim et al., 2006; Kooperberg et al., 2005). No one test is the best for all situations; however, we can do better in selecting an appropriate test after understanding the merits and limitations of these different tests.

ACKNOWLEDGEMENTS

This work was supported by the Korea Research Foundation Grant funded by the Korean Government (MOEHRD, Basic Research Promotion Fund) (KRF-2007-331-C00065).

REFERENCES

- Agresti, A. (2002). *Categorical Data Analysis*. Hoboken, NJ: Wiley.
- Benjamini, Y., and Hochberg, Y. (1995). Controlling the false discovery rate: A practical and powerful approach to multiple testing. *J. R. Stat. Soc. Ser. B (Methodol.)*, **57**(1): 289–300.
- Benjamini, Y., and Yekutieli, D. (2001). The control of the false discovery rate in multiple testing under dependency. *Ann. Stat.*, **29**(4): 1165–1188.
- Bolstad, B. M., Irizarry, R. A., Astrand, M., and Speed, T. P. (2003). A comparison of normalization methods for high density oligonucleotide array data based on variance and bias. *Bioinformatics*, **19**(2): 185–193.
- Cho, H., and Lee, J. K. (2004). Bayesian hierarchical error model for analysis of gene expression data. *Bioinformatics*, **20**(13): 2016–2025.
- Cho, H., Smalley, D. M., Theodorescu, D., Ley, K., and Lee, J. K. (2007). Statistical identification of differentially labeled peptides from liquid chromatography tandem mass spectrometry. *Proteomics*, **7**(20): 3681–3692.
- Cho, H. J., and Lee, J. K. (2008). Error-pooling empirical Bayes model for enhanced statistical discovery of differential expression in microarray data. *IEEE Trans. Syst. Man Cybernet. Part A: Syst. Hum.*, **38**(2): 425.
- Dudoit, S., Shaffer, J. P., and Boldrick, J. C. (2003). Multiple hypothesis testing in microarray experiments. *Stat. Sci.*, **18**(1): 71–103.
- Dudoit, S., Yang, Y. H., Callow, M. J., and Speed, T. P. (2002). Statistical methods for identifying differentially expressed genes in replicated cDNA microarray experiments. *Stat. Sinica*, **12**(1): 111–139.
- Efron, B., and Tibshirani, R. (2007). On testing the significance of sets of genes. *Ann. Appl. Stat.*, **1**(1): 107–129.
- Garcia, B. A., Smalley, D. M., Cho, H., Shabanowitz, J., Ley, K., and Hunt, D. F. (2005). The platelet micro-particle proteome. *J. Proteome Res.*, **4**(5): 1516–1521.
- Hochberg, Y. (1988). A sharper Bonferroni procedure for multiple tests of significance. *Biometrika*, **75**(4): 800–802.
- Holm, S. (1979). A simple sequentially rejective multiple test procedure. *Scand. J. Stat.*, **6**(65–70): 1979.
- Hommel, G. (1988). A stagewise rejective multiple test procedure based on a modified Bonferroni test. *Biometrika*, **75**(2): 383–386.
- Jain, N., Thatte, J., Braciale, T., Ley, K., O’Connell, M., and Lee, J. K. (2003). Local-pooled-error test for identifying differentially expressed genes with a small number of replicated microarrays. *Bioinformatics*, **19**(15): 1945–1951.
- Jiang, Z., and Gentleman, R. (2007). Extensions to gene set enrichment. *Bioinformatics*, **23**(3): 306–313.
- Kim, S. Y., Lee, J. W., and Sohn, I. S. (2006). Comparison of various statistical methods for identifying differential gene expression in replicated microarray data. *Stat. Methods Med. Res.*, **15**(1): 3.
- Kooperberg, C., Aragaki, A., Strand, A. D., and Olson, J. M. (2005). Significance testing for small microarray experiments. *Stat. Med.*, **24**(15): 2281–2298.
- Pollard, K. S., and van der Laan, M. J. (2005). Resampling-based multiple testing with asymptotic strong control of type I error. Preprint, Division of Biostatistics, University of California, Berkeley, CA.
- Sidak, Z. (1967). Rectangular confidence regions for the means of multivariate normal distributions. *J. Am. Stat. Assoc.*, **62**(31): 626–633.
- Smyth, G. K. (2004). Linear models and empirical bayes methods for assessing differential expression in microarray experiments. *Stat. Appl. Genet. Mol. Biol.*, **3**(1).
- Storey, J. D. (2002). A direct approach to false discovery rates. *J. R. Stat. Soc. Ser. B (Stat. Methodol.)*, **64**(3): 479–498.
- Subramanian, A., et al. (2005). Gene set enrichment analysis: A knowledge-based approach for interpreting genome-wide expression profiles. *Proc. Natl. Acad. Sci. U.S.A.*, **102**(43): 15545–15550.
- Tian, L., Greenberg, S. A., Kong, S. W., Altschuler, J., Kohane, I. S., and Park, P. J. (2005). Discovering statistically significant pathways in expression profiling studies. *Proc. Natl. Acad. Sci. U.S.A.*, **102**(38): 13544–13549.
- Tusher, V. G., Tibshirani, R., and Chu, G. (2001). Significance analysis of microarrays applied to the ionizing radiation response. *Proc. Natl. Acad. Sci. U.S.A.*, **98**(9): 5116–5121.
- Yang, Y. H., et al. (2002). Normalization for cDNA microarray data: A robust composite method addressing single and multiple slide systematic variation. *Nucleic Acids Res.*, **30**(4): e15.

CLUSTERING: UNSUPERVISED LEARNING IN LARGE BIOLOGICAL DATA

*Nabil Belacel, Christa Wang, and Miroslava Cuperlovic-Culf
National Research Council Canada, Institute for Information Technology,
Moncton, NB, Canada*

With the advancement of high-throughput biotechnologies, biological data describing DNA, RNA, protein, and metabolite biomolecules are generated faster than ever. Huge amount of information is being produced and collected. Bioinformatics uses information technology to facilitate the discovery of new knowledge from large sets of various biological data at the molecular level. Within various applications of information technology, clustering has long played an important role.

Clustering is an exploratory tool for analyzing large datasets and has been applied extensively in numerous biological, medical, and health research areas ranging from clinical information, phylogeny, genomics, proteomics, and various other omics methodologies. The classification problems can be divided into two categories. The first one is based on the notion of supervised learning and employs a set of examples belonging to well-known classes called training sets. The classification rules are defined from these training examples. The previous chapter on supervised learning methods described more this category of classification. The second category comprises the clustering problems based on the notion of unsupervised learning and consists of assembling samples in restricted classes so that samples within one class are less dispersed than between classes. Specifically, classes of samples should be homogeneous and in some methods well separated. The clustering algorithms are divided into several groups that divide simple clustering where each entity belongs to one cluster (hard or crisp clustering) from complex clustering where each entity can belong to more than one cluster with a certain degree of membership (soft or relaxed clustering). The first group includes three conventional, widely used clustering algorithms: hierarchical clustering and two nonhierarchical clustering algorithms, *k*-means and self-organizing maps (SOMs). The second group of algorithms includes fuzzy *c*-means and model-based clustering methods. In this chapter, we provide an introduction to cluster analysis in life and biomedical sciences. We will also illustrate the impact of clustering methodologies on several fascinating areas of biosciences.

This chapter begins with a high-level overview of similarity measures followed by a discussion of the commonly used clustering approaches, including few exemplary applications in biomedical sciences. The final section of this chapter is devoted to cluster validity methods developed for measuring the compactness and separation quality of the clusters produced by the analysis.

5.1 MEASURES OF SIMILARITY

Informally, the *similarity* between two objects is a numerical measure of the degree to which the two objects are alike. Because of the diverse and complex nature of data in life and biomedical science, a wide range of different objects gives rise to different analysis requirements of similarity measures and clustering solutions. According to Zhao and Karypis (2005), there are three fundamental types of data objects:

1. *Multidimensional Vectors*: Each object is characterized by a set of features (attributes), usually represented as a multidimensional vector (e.g., gene expression data).
2. *Sequences*: Each object contains a sequence of symbols with variable length, which could be nucleotides, amino acids, or secondary structure elements (e.g., DNA and protein sequence).
3. *Structures*: Each object is represented as a two- or three-dimensional (2D, 3D) structure (e.g., 3D protein structure).

Clustering plays a significant role in the analysis of all three object types with major differences in the type of similarity measures applied. A brief description of some examples is provided below.

5.1.1 Similarity Measures for Multidimensional Objects

Before delving into the specific similarity calculation, we start our discussion with the characteristics of attributes in multidimensional data objects. The attributes can be quantitative or qualitative, continuous or binary, nominal or ordinal, which determines the corresponding similarity calculation (Xu and Wunsch, 2005). Typically, distance-based similarity measures are used to measure continuous features, while matching-based similarity measures are more suitable for categorical variables.

5.1.1.1 Distance-Based Similarity Measures Similarity measures determine the proximity (or distance) between two data objects. Multidimensional objects can be formalized as numerical vectors $\mathbf{O}_i = \{o_{ij} \mid 1 \leq j \leq p\}$, where o_{ij} is the value of the j th feature for the i th data object and p is the number of dimensions for the data object o_{ij} . Figure 5.1 provides an intuitive view of multidimensional data. The similarity between two objects can be measured by a distance function of corresponding vectors \mathbf{o}_i and \mathbf{o}_j .

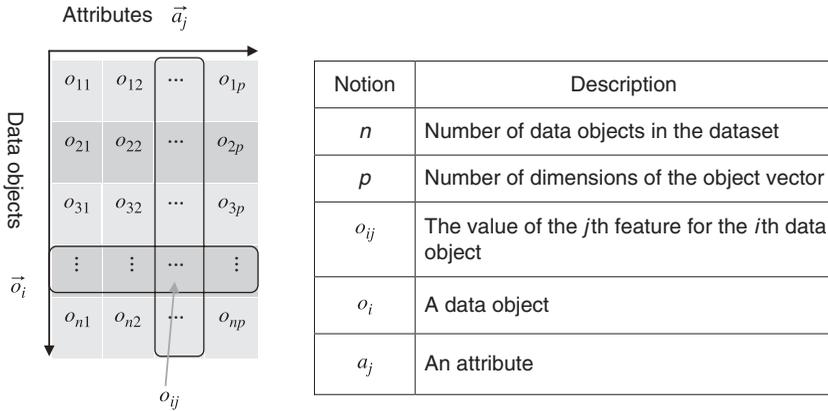


Figure 5.1 Matrix of multidimensional data.

One of the most used classes of distance functions is the *Minkowski* distance function shown in the equation

$$d(o_i, o_j) = \left(\sum_{k=1}^p |o_{ik} - o_{jk}|^r \right)^{1/r} \tag{5.1}$$

- where r = norm used
- p = number of dimensions for multidimensional objects
- o_i, o_j = data objects

In this family, three most common distances are as follows:

1. $r = 1$, the distance function is called *city block* (or *Manhattan*, or L_1 norm). The distance function is given as

$$d(o_i, o_j) = \sum_{k=1}^p |o_{ik} - o_{jk}| \tag{5.2}$$

A common application is the Hamming distance, which calculates the number of bit differences between two objects that have only binary attributes (Tan et al., 2006).

2. $r = 2$ the distance function is named *Euclidean* (or L_2 norm) distance. The distance function is

$$d(o_i, o_j) = \left(\sum_{k=1}^p |o_{ik} - o_{jk}|^2 \right)^{1/2} \tag{5.3}$$

Euclidean distance has been widely used in the partitional clustering algorithms (Xu and Wunsch, 2005).

3. $r \rightarrow \infty$, the distance function is a special case of *supremum* (L_∞ norm) distance, also known as Chebyshev distance. The *supremum norm* approach has been previously used in the fuzzy c -means clustering method (Bobrowski and Bezdek, 1991).

It should be noted that this class of measures treats all dimensions equally, regardless of their distribution (DeSmet et al., 2002; Tavazoie et al., 1999).

The application of these measures can become problematic when clustering high-dimensional data. Alternatively, researchers have resorted to another commonly used theme for determining the similarity between the shapes of two data objects, called *Pearson's correlation coefficient*.

Given two data objects o_i and o_j , Pearson's correlation coefficient is defined as

$$\text{Pearson } (o_i, o_j) = \frac{\sum_{k=1}^p (o_{ik} - \mu_{o_i})(o_{jk} - \mu_{o_j})}{\sqrt{\sum_{k=1}^p (o_{ik} - \mu_{o_i})^2} \sqrt{\sum_{k=1}^p (o_{jk} - \mu_{o_j})^2}} \quad (5.4)$$

where μ_{o_i} = mean for object \mathbf{o}_i

μ_{o_j} = mean for object \mathbf{o}_j

p = number of dimensions for object vector

Pearson's correlation coefficient has been proven effective as a similarity measure, especially for the analysis of gene expression data (Jiang et al., 2003; Tang et al., 2001, 2002). However, it still suffers from two major drawbacks (Jiang et al., 2004):

- a. It is sensitive to outliers since outliers may significantly affect the similarity measure. When measuring two data objects with a common peak or valley at a single feature, Pearson's correlation coefficient will be dominated by this feature regardless of remaining features (Jiang et al., 2004).
- b. It assumes that any two vectors \mathbf{o}_i and \mathbf{o}_j are approximately normally distributed and may be problematic for nonnormal distributions (Jiang et al., 2004).

Since there is no superior similarity measure, which can address all the issues, the selection of different measures is problem dependent. The optimal distance similarity method can be determined from the clustering results as well as the analysis of cluster quality assessment methods (see below). An example showing the K -means clustering results for a subset of gene expression data published by Bhattacharjee et al. (2001) how type of dissimilarity measure can have a great impact on the final clustering results (Fig. 5.2) and the cluster quality (Table 5.1).

5.1.1.2 Matching-Based Similarity Measures For categorical attributes, distance-based similarity measures cannot be performed directly. The most straightforward way is to compare the similarity of objects for pairs of categorical attributes (Zhao and Karypis, 2005). For two objects that contain simple binary attributes¹,

¹Binary attributes: attributes that have only two values (e.g., yes/no, true/false). Such values can be simply mapped into binary values (1/0).

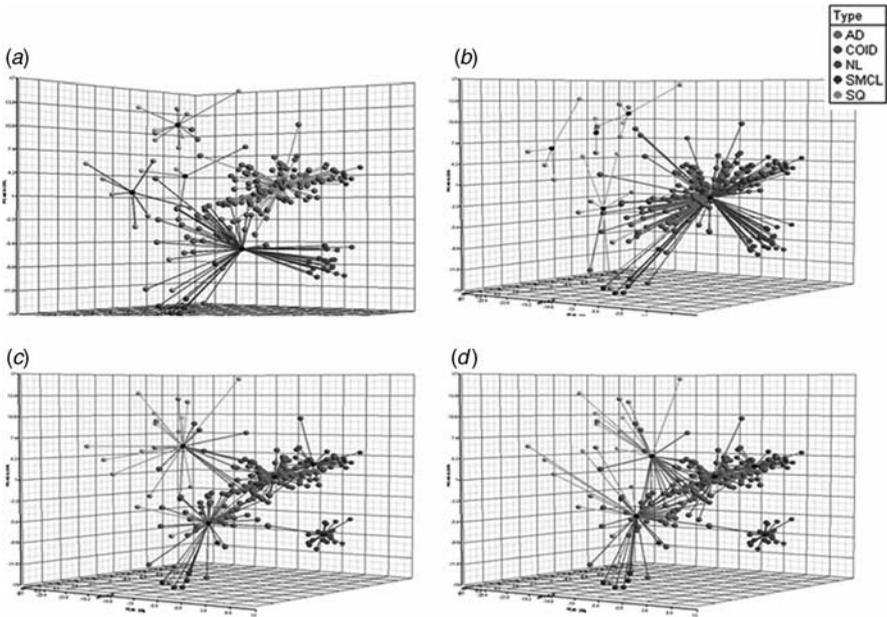


Figure 5.2 Demonstration of using different distance measures. The dataset used for this demonstration is the subset of genes from microarray study by Bhattacharjee et al. (2001). The clustering method used is the *K*-means algorithm. Different distance measures result in different clustering solutions: (a) Euclidean, (b) city block, (c) Pearson, and (d) cosine distances. (See color insert.)

similarity measures between two objects are called *similarity coefficients* (Tan et al., 2006). A value of 1 indicates that the two objects are completely identical, while a value of 0 indicates that the objects are dissimilar. Let *X* and *Y* be two objects with *n* binary attributes; the comparison of any two binary vectors will lead to the following result:

- n_{00} = number of attributes where $x_i = 0$ and $y_i = 0$
- n_{01} = number of attributes where $x_i = 0$ and $y_i = 1$
- n_{10} = number of attributes where $x_i = 1$ and $y_i = 0$
- n_{11} = number of attributes where $x_i = 1$ and $y_i = 1$

TABLE 5.1 Cluster Quality for Different Distance Measures

Cluster validation method	Euclidean	City block	Pearson’s dissimilarity	Cosine dissimilarity
Davies–Bouldin	2.72	2.00864	1.26238	1.91735
Modified Hubert	0.011	4.0664×10^{-5}	27.0519	26.3572

Note: The cluster quality measures are based on the clustering results in the Figure 5.1.

The *simple match coefficient* (SMC), one of the most used similarity coefficients, can be defined as

$$\text{SMC}(X, Y) = \frac{n_{11} + n_{00}}{n_{00} + n_{01} + n_{10} + n_{11}} \quad (5.5)$$

where $n_{11} + n_{00}$ = number of matching attributes
 $n_{00} + n_{01} + n_{10} + n_{11}$ = total number of attributes

However, the limitation of this measure is that it accounts for both presences and absences equally (Tan et al., 2006). For sparse data², the similarity result from SMC will be dominated by the number of 0–0 matches. As a result, the Jaccard coefficient [Eq. (5.6)] was proposed to calculate the similarity coefficient based on the number of shared attributes while ignoring 0–0 matches:

$$J(X, Y) = \frac{n_{11}}{n_{11} + n_{01} + n_{10}} \quad (5.6)$$

where n_{11} = number of matching presences
 $n_{11} + n_{01} + n_{10}$ = number of attributes without 0–0 matches

For nominal attributes³, an effective method is to utilize the dissimilarity criterion (Huang, 1997). Let X and Y be two objects with m nominal attributes. The dissimilarity measure between X and Y is formulated by the number of matching attributes:

$$d(X, Y) = \sum_{i=1}^m S(x_i, y_i) \quad (5.7)$$

where

$$S(x_i, y_i) = \begin{cases} 1 & \text{if } x_i \text{ and } y_i \text{ do not match} \\ 0 & \text{if } x_i \text{ and } y_i \text{ match} \end{cases}$$

When taking into account the frequency of categories in a dataset, a normalized variant of the above measure is defined as follows:

$$d(X, Y) = \sum_{i=1}^m \frac{(nx_i + ny_i)}{nx_i ny_i} S(x_i, y_i) \quad (5.8)$$

where nx_i (ny_i) is the number of objects in the dataset that have the categorical values x_i (y_i) for attribute i . According to the formula, this measure gives more weight to rare categories than the frequent ones.

²Sparse data: most attributes of an object have values of zero with fewer nonzero attributes.

³Nominal attributes: attributes have more than two states. Consider an attribute that measures patient's marital status with a set of possible values {Married, Single, Divorced, ...}.

An ordinal variable is a nominal attribute with multiple states ordered in a meaningful sequence. Consider an attribute that measures the degree of suicide risk on the scale {low, moderate, high}. Obviously, the values of the ordinal attribute can be mapped to successive integers. The dissimilarity between two objects X and Y with ordinal attributes is measured as the Manhattan distance [Eq. (5.2)] divided by the number of variables for both objects (Kaufman and Rousseeuw, 1990).

5.1.2 Similarity Measures for Sequences

Biological sequences, which consist of a sequence of nucleotides, amino acids, or secondary-structure elements (SSEs), are different from the objects we have discussed in Section 5.1.1. Hence, the above similarity measures are not applicable to biological sequences. In recent decades, many approaches have been proposed for the similarity measurement of sequences, the common ones being the alignment-based measures.

A sequence alignment can be viewed as a process of transforming a given sequence to another one with a series of editing operations such as matching, insertion, deletion, and substitution. For instance, let I denote the *insert* operation, D denote the *delete* operation, S denote the *substitution* operation, and M denote the *matching* operation. Then the string “wvinter” can be edited into “writers” as follows:

	M	S	M	D	M	M	M	I
S1	w	v	i	n	t	e	r	
S2	w	r	i		t	e	r	s

Sequence comparison, sometimes referred to as the edit distance (or Levenshtein distance) procedure (Levenshtein, 1966), is performed in order to determine the similarity by measuring the degree of agreement in the aligned positions of two sequences. The goal of sequence alignment is to complete the transformation with the minimum cost. In this context, the similarity measure between two sequences can be considered as an optimal alignment problem (Xu and Wunsch, 2005).

In general, alignment-based measures can be divided into two categories: *global sequence alignment* and *local sequence alignment*.

5.1.2.1 Global Sequence Alignment Global sequence alignment, also called Needleman–Wunsch alignment (Needleman and Wunsch, 1970), aligns the whole length of two sequences based on dynamic programming. Given two sequences S_1 of length n and S_2 of length m , the alignment score is residue based and calculated as the sum of substitution scores and gap penalties. In life science, especially for similarity measure of protein sequences, the most common substitution scoring systems are point accepted mutation (PAM) (Dayhoff et al., 1978) and the blocks substitution matrix (BLOSUM) family (Henikoff and Henikoff, 1992). Figure 5.3 is a 20×20 BLOSUM62 matrix. An alignment matrix $D(i, j)$ indexed by residues of each sequence can be formulated recursively as follows:

1. Base conditions:

$$D(0, j) = \sum_{1 \leq k \leq j} g[\rightarrow, g_2(k)]$$

	<u>A</u>	<u>R</u>	<u>N</u>	<u>D</u>	<u>C</u>	<u>Q</u>	<u>E</u>	<u>G</u>	<u>H</u>	<u>I</u>	<u>L</u>	<u>K</u>	<u>M</u>	<u>F</u>	<u>P</u>	<u>S</u>	<u>T</u>	<u>W</u>	<u>Y</u>	<u>V</u>
<u>A</u>	4	-1	-2	-2	0	-1	-1	0	-2	-1	-1	-1	-1	-2	-1	1	0	-3	-2	0
<u>R</u>	-1	5	0	-2	-3	1	0	-2	0	-3	-2	2	-1	-3	-2	-1	-1	-3	-2	-3
<u>N</u>	-2	0	6	1	-3	0	0	0	1	-3	-3	0	-2	-3	-2	1	0	-4	-2	-3
<u>D</u>	-2	-2	1	6	-3	0	2	-1	-1	-3	-4	-1	-3	-3	-1	0	-1	-4	-3	-3
<u>C</u>	0	-3	-3	-3	9	-3	-4	-3	-3	-1	-1	-3	-1	-2	-3	-1	-1	-2	-2	-1
<u>Q</u>	-1	1	0	0	-3	5	2	-2	0	-3	-2	1	0	-3	-1	0	-1	-2	-1	-2
<u>E</u>	-1	0	0	2	-4	2	5	-2	0	-3	-3	1	-2	-3	-1	0	-1	-3	-2	-2
<u>G</u>	0	-2	0	-1	-3	-2	-2	6	-2	-4	-4	-2	-3	-3	-2	0	-2	-2	-3	-3
<u>H</u>	-2	0	1	-1	-3	0	0	-2	8	-3	-3	-1	-2	-1	-2	-2	-2	-2	2	-3
<u>I</u>	-1	-3	-3	-3	-1	-3	-3	-4	-3	4	2	-3	1	0	-3	-2	-1	-3	-1	3
<u>L</u>	-1	-2	-3	-4	-1	-2	-3	-4	-3	2	4	-2	2	0	-3	-2	-1	-2	-1	1
<u>K</u>	-1	2	0	-1	-3	1	1	-2	-1	-3	-2	5	-1	-3	-1	0	-1	-3	-2	-2
<u>M</u>	-1	-1	-2	-3	-1	0	-2	-3	-2	1	2	-1	5	0	-2	-1	-1	-1	-1	1
<u>F</u>	-2	-3	-3	-2	-2	-3	-3	-3	-1	0	0	-3	0	6	-4	-2	-2	1	3	-1
<u>P</u>	-1	-2	-2	-1	-3	-1	-1	-2	-2	-3	-3	-1	-2	-4	7	-1	-1	-4	-3	-2
<u>S</u>	1	-1	1	0	-1	0	0	0	-1	-2	-2	0	-1	-2	-1	4	1	-3	-2	-2
<u>T</u>	0	-1	0	-1	-1	-1	-1	-2	-2	-1	-1	-1	-1	-2	-1	1	5	-2	-2	0
<u>W</u>	-3	-3	-4	-4	-2	-2	-3	-2	-2	-3	-2	-2	-1	1	-4	-3	-2	11	2	-3
<u>Y</u>	-2	-2	-2	-3	-2	-1	-2	-3	-1	-2	-3	2	-1	3	-3	-2	-2	2	7	-1
<u>V</u>	0	-3	-3	-3	-1	-2	-2	-3	-3	3	1	-2	1	-1	-2	-2	0	-3	-1	4

Figure 5.3 BLOSUM62 substitution matrix (20 × 20).

and

$$D(i, 0) = \sum_{1 \leq k \leq i} g[g_1(k), -] \quad (5.9)$$

2. Recurrence relations:

$$D(i, j) = \max \left\{ \begin{array}{l} D(i-1, j-1) + S[S_1(i), S_2(j)] \\ D(i-1, j) + g[S_1(i), -] \\ D(i, j-1) + g[-, S_2(j)] \end{array} \right\} \quad (5.10)$$

Where the underbar represents a space; $S[S_1(i), S_2(j)]$ is the substitution score for residues i in the sequence S_1 and j in the sequence S_2 ; $g[S_1(i), -]$, $g[-, S_2(j)]$ are the gap penalties; and $D(n, m)$ is the optimal alignment score, n is the length of the sequence S_1 , and m is the length of the sequence S_2 .

Needleman–Wunsch alignment is performed in three steps:

1. Initialization of the score matrix according to the base conditions
2. Calculation of the alignment score for each indexed matrix $D(i, j)$
3. Deduction of the best alignment from the traceback matrix

Figure 5.4 depicts a walk-through example for this alignment procedure.

Global sequence alignment is most appropriate for finding the best alignment of two sequences which are of similar length or similar across the entire length. In many applications, two sequences may not be highly similar in their entirety but may contain highly similar sections. Local sequence alignment addresses the issue, which is to find a pair of substrings with the highest alignment score among all possible pairs of substrings in the two sequences.

5.1.2.2 Local Sequence Alignment Local sequence alignment, also referred to as Smith–Waterman alignment (Smith and Waterman, 1981), is most useful for the analysis of dissimilar sequences with possible regions of similarity. Smith–Waterman alignment allows the start point and the end point of an alignment to be anywhere in the dynamic programming matrix during the recursive computation (Durbin et al., 1998; Smith and Waterman, 1981; Xu and Wuncsh, 2005). The recurrence for local sequence alignment is very similar to that for global sequence alignment with only minor changes. These changes can be summarized as follows:

1. Base conditions:

$$D(0, j) = 0 \quad \text{and} \quad D(i, 0) = 0 \quad (5.11)$$

2. Recurrence relation:

$$D(i, j) = \max \left\{ \begin{array}{l} 0 \\ D(i-1, j-1) + S[S_1(i), S_2(j)] \\ D(i-1, j) + g[S_1(i), -] \\ D(i, j-1) + g[-, S_2(j)] \end{array} \right\} \quad (5.12)$$

Consider the alignment of two sequences SAND and AND with the BLOSUM62 substitution matrix and gap penalty of 1.

(a) Initializing the score matrix according to the base conditions:

	S	A	N	D
A	0	-1	-2	-3
N	-1			
D	-2			
D	-3			

(b) Generating the score table and traceback table based on the recurrence relations:

	S	A	N	D
A	0	-1	-2	-3
N	-1	1	3	2
D	-2	0	-1	9
D	-3	-1	-1	8

	S	A	N	D
End	Up	Up	Up	Up
Left	Diag	Diag	Left	Left
Left	Up/ Diag	Up/ Diag	Diag	Left
Left	Up	Up/ Left/ Diag	Up	Diag

$$D(2,2) = \max \begin{cases} \text{Diag} = D(1,1) + S(2,2) = 0 + 1 = 1 \\ \text{Up} = D(1,2) + g(S, _) = -1 - 1 = -2 = 1 \\ \text{Left} = D(2,1) + g(_, A) = -1 - 1 = -2 \end{cases}$$

Diag contributes to the optimal alignment score for $D(2,2)$.
 $S(2,2)$ is achieved from BLOSUM62 substitution matrix.

(c) Deducing the best alignment from the traceback matrix:

SAND
 _AND
 It has the best alignment from the traceback path $[D(4,5) \rightarrow D(3,4) \rightarrow D(2,3) \rightarrow D(2,1)]$ with alignment score = 26.

Figure 5.4 Walk-through example of Needleman–Wunsch alignment.

where the underbar again represents a space; $S[S_1(i), S_2(j)]$ is the substitution score for residues i in the sequence S_1 and j in the sequence S_2 ; and $g[S_1(i), _]$, $g[_, S_2(j)]$ are the gap penalties. The base conditions ensure that the alignment can begin from any dynamic programming matrix (i, j) .

It should be noted that, for both global and local alignment, the time complexity is $O(nm)$, highly computationally expensive, in particular for clustering problems (Xu and Wunsh, 2005) or when processing a large volume of nucleic acids and amino acid sequences in a database. Alternatively, sequence comparison could be achieved via heuristic approaches, such as FASTA, BLAST, and their variants (Altschul et al., 1990, 1997; Lipman and Pearson, 1985; Pearson and Lipman, 1988). To achieve low

time complexity, such heuristic approaches identify the regions with potential high matches and then apply expensive but more accurate algorithms for further search on these regions only. The implementation of such a scheme in searching a large-scale database exhibits a significant improvement in computation time (Xu and Wuncsh, 2005). Most existing sequence clustering algorithms use the similarity measure based on local sequence alignment (Zhao and Karypis, 2005). Interested readers can follow the references for details.

5.1.3 Similarity Measures for Structural Data

Another important application of clustering in life sciences is molecular, particularly protein, structures comparison. Most of the comparative analysis involves the direct alignment and superimposition of structures in a three-dimensional space (Xiong, 2006). According to (Xiong, 2006). The structure comparison is one of the fundamental techniques in protein structure analysis for three reasons: (a) it can detect remote protein homologs, since structural comparison can reveal distant evolutionary relationships between proteins; (b) it is a prerequisite for protein structural classification; and (c) comparison of theoretically known structures with experimentally predicted ones is a critical procedure in the evaluation of protein structure prediction methods.

Several techniques are available for similarity measures on three-dimensional structures. The most common technique is *structure superposition* or *superimposition*. In this technique, one structure is rotated or reoriented until it can be superimposed onto the other structure (Wishart, 2005). The procedure starts with identifying common reference points. As soon as these points are identified, one of the structures is rotated until these common reference points are almost matching with minimum differences, that is, minimal distance between equivalent positions on the protein structure. One of the most commonly used distance measures is *root mean-square deviation* (RMSD) (Xiong, 2006), determined as

$$\text{RMSD} = \sqrt{\frac{\sum_{i=1}^N D_i^2}{N}} \quad (5.13)$$

where N = number of atoms compared
 D_i = difference of atomic distance

Several web-based tools perform structural comparisons automatically, including, DALI (Dietmann et al., 2001), CE (Shindyalov and Bourne, 2001), and VAST (Gibrat et al., 1996).

5.2 CLUSTERING

Clustering is a highly effective technique in data analysis. Prior to the introduction of clustering, it is important to understand the difference between clustering (unsupervised classification) and supervised classification. In supervised classification, we use training objects with known class labels to develop a model and then deploy

this model to assign unlabeled testing objects with a class label. Figure 5.5 displays the supervised learning process. In the case of clustering, the problem to be solved is to group a collection of objects into a set of meaningful clusters without prior knowledge of class labels. Figure 5.6 shows the general idea of clustering. The objective of clustering is to determine group objects such that the objects within a group are similar to one another and different from the objects in other groups. Before discussing specific clustering techniques, we will provide some necessary background on different types of clustering.

5.2.1 Different Types of Clustering

Clustering has been extensively studied in a wide variety of scientific domains with many different clustering algorithms available. Depending on the clustering property observed, there are several different types of algorithms.

Training Dataset

ID	Attr1	Attr2	Attr3	Class
1	10	Large	12K	Low
2	1	Small	100K	High
3	15	Small	70K	Med
4	Low
5	Low
6	High
7	Med

Testing Dataset

ID	Attr1	Attr2	Attr3	Class
1	?
2	?
3	?

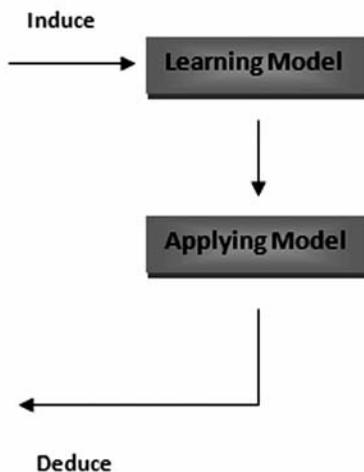


Figure 5.5 Learning procedure of supervised classification.

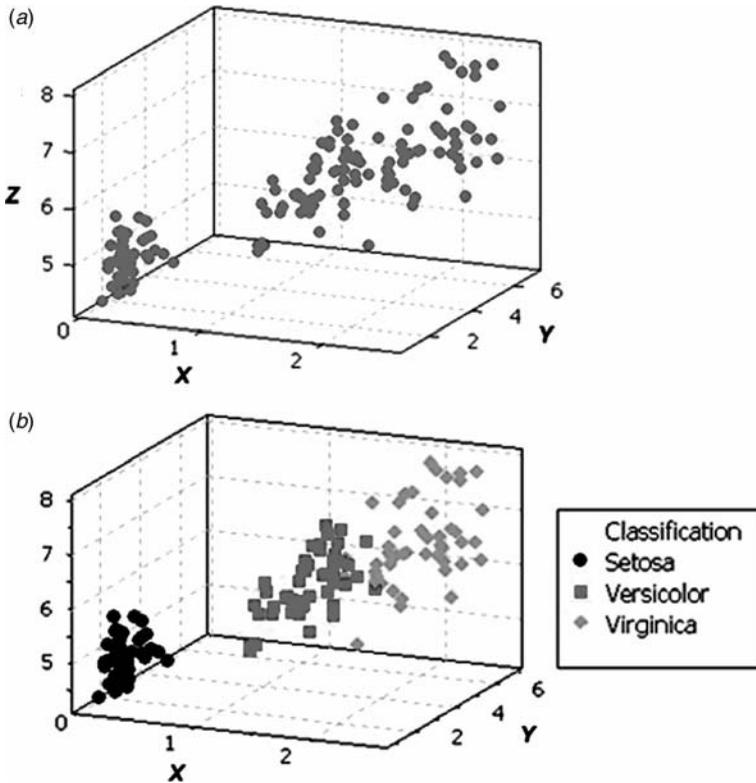


Figure 5.6 Unsupervised learning procedure of clustering: (a) before clustering (Iris dataset without class labels); (b) after clustering (three clusters: Setosa, Versicolor, and Virginica). (See color insert.)

5.2.1.1 Hierarchical versus Nonhierarchical Clustering According to the structure of final results, data can be clustered in a hierarchical or nonhierarchical manner. The taxonomy structure [the taxonomy we used here is based on the discussion in Jain et al. (1999)] is shown in Figure 5.7. Hierarchical clustering finds its final clusters by a nest of partitions, whereas nonhierarchical clustering finds its final clusters by partitioning the entire dataset into either a predefined number of clusters or an automatically detected number of clusters (Zhao and Karypis, 2005).

The *hierarchical clustering* algorithms have been used extensively for the analysis of DNA microarray data (Alizadeh et al., 2000; Eisen et al., 1998; Nielsen et al., 2002; Ramaswamy et al., 2003; Welch et al., 2002). This type of clustering groups data and provides a natural way for graphical representation of results that allows a straightforward way for the detection of the higher order relationships between clusters of profiles. Hierarchical clustering algorithms can be further divided into two categories: agglomerative and divisive clustering (discussed further in the next section).

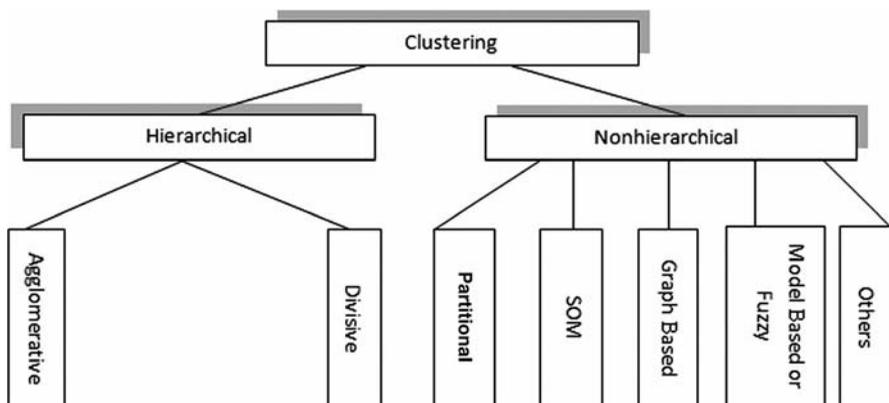


Figure 5.7 Taxonomy of clustering algorithms.

The *nonhierarchical clustering* algorithms have also yielded encouraging results in clustering biological data. Within partitional clustering algorithms, *K*-means and its variants were utilized to identify molecular subtypes of brain tumors (Shai et al., 2003), to cluster transcriptional regulatory cell cycle genes in yeast (Tavazoie et al., 1999), and to correlate changes in gene expression with major physiological events in potato biology (Ronning et al., 2003), to name only a few applications. As one of the best-known, unsupervised neural network learning algorithms, the SOM was deployed to analyze and visualize gene expression during a diauxic shift in yeast (Toronen et al., 1999), for example. In recent decades, several new clustering algorithms (e.g., graph-based clustering, model-based clustering, and fuzzy clustering) have appeared with the intention to combine and improve the features of traditional clustering algorithms.

5.2.1.2 Crisp versus Fuzzy Clustering Crisp (hard) clustering is based on the assumption that each data object can be assigned to only one cluster. Although a majority of clustering applications in biology utilized crisp methods, the restriction of one-to-one mapping might not be optimal, especially in the analysis of molecular data. A majority of genes can participate in different genetic networks and are governed by a variety of regulatory mechanisms (Futschik and Kasabov, 2002). Fuzzy clustering allows a data object to be assigned to multiple clusters with certain degrees of membership. The memberships can further be used to discover more sophisticated relations between the data object and its disclosed clusters (Xu and Wunsch, 2005). Fuzzy logic provides a so-called fuzzification process (Woolf and Wang, 2002), which is beneficial for analysis of molecular data where no prior knowledge is available for the datasets.

5.2.1.3 One-Way versus Two-Way Clustering Clustering methods that were mentioned thus far can be referred to as one-way clustering. When processing multi-dimensional data objects, one-way clustering can be applied to either the rows or the columns of the data matrix, separately. For a given dataset, one-way methods perform

global clustering, in which the feature space is globally determined and shared by all resulting clusters and the resulting clusters are exclusive and exhaustive (Jiang et al., 2004). However, in the case of, for example, gene expression data analysis, we should take into account the following situations: (1) In any cellular processes of interest, only subsets of genes are coregulated or coexpressed under certain experimental conditions (Madeira and Oliveira, 2004). (2) A gene can participate in multiple pathways that may or may not be coactive under all conditions. Hence, a gene or sample should be allowed to belong to more than one cluster or to no cluster at all (Madeira and Oliveira, 2004; Jiang et al., 2004). In this context, it is highly desirable to move beyond the traditional one-way clustering paradigm and develop new clustering schemes capable of discovering such local expression patterns (Ben-Dor et al., 2002; Madeira and Oliveira, 2004).

The term “biclustering”, also referred to as coclustering, bidimensional clustering, subspace clustering, or two-way clustering, was first introduced in the gene expression data analysis by Cheng and Church (2000). The objective of such clustering techniques is to capture coherent blocks (subspace clusters or biclusters) within the gene expression matrices by performing simultaneous clustering on both rows and columns. The resulting blocks are submatrices defined by subsets of genes and subsets of conditions. Figure 5.8 provides an intuitive view of such blocks. Several

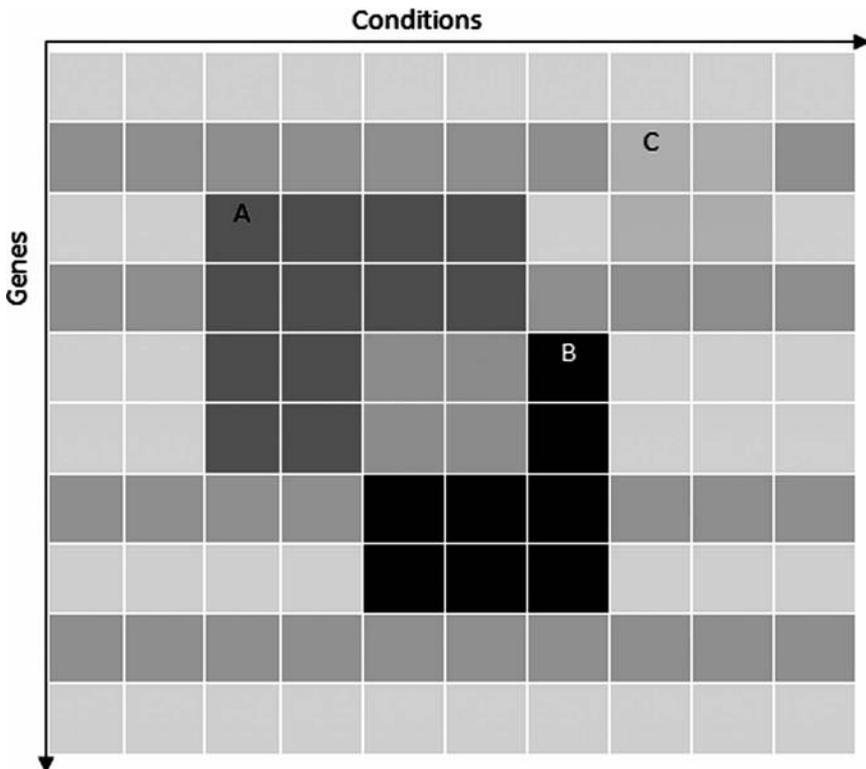


Figure 5.8 Example of biclusters in gene expression matrix. (See color insert.)

characteristics of the results of biclustering can be observed from the figure: First, the subset of genes/conditions in various biclusters could be different; second, two biclusters can share certain common features; third, some genes/conditions may not be included in any biclusters.

The majority of biological applications of biclustering are focused on microarray gene expression data and were applied to identify coregulated genes, gene functional annotations, and sample classification. Two recent surveys (Madeira and Oliveira, 2004; Tanay et al., 2006) on the topic of “biclustering” offer a comprehensive review of the different applications and algorithms. An important characteristic of gene expression data is their high level of noise. Any biclustering algorithms should be robust enough to cope with the significant level of noise and high complexity of molecular biological data.

5.2.2 Clustering Algorithms

Our discussion of clustering algorithms will start with traditional methods including hierarchical clustering, K -means, and the SOM. The PubMed publication analysis of Dopazo (2006) has shown that of 1157 papers dealing with clustering in microarrays (keywords “cluster” and “microarray”) 74% used hierarchical clustering, 15% used K -means, and 6% used SOM and only 5% of publications used alternative clustering methods. Although encouraging results have been produced by the application of traditional methods, their weaknesses often complicate clustering of biological data. For this reason, several advanced clustering algorithms have been proposed trying to overcome some of the shortcomings of the earlier methods. Those newly proposed clustering algorithms include model-based clustering, fuzzy clustering, and biclustering. In the second part of this section, we will focus on advanced clustering algorithms. Since it is impossible to give an exhaustive coverage of all clustering methods used in biological data analysis, the selection of algorithms is biased toward the most practical and most often utilized techniques.

5.2.2.1 Conventional Clustering Algorithms

Hierarchical Clustering The principle behind the hierarchical clustering is to group data objects into a tree of clusters through either agglomerative or divisive processes. Figure 5.9 depicts two approaches in hierarchical clustering.

- *Agglomerative clustering* follows a bottom-up fashion. The clustering starts with treating data objects as individual clusters. Subsequently, the closest pairs of clusters are merged so that the cluster proximity is optimized. This merging process is going on until either all objects are in a single cluster or a certain termination criterion is met.
- *Divisive clustering* represents a top-down strategy. It starts with one, all-object-inclusive cluster and splits the cluster until either each object forms its own cluster or a certain termination condition is reached.

Agglomerative hierarchical clustering is one of the most utilized clustering methods for the analysis of biological data (Quackenbush, 2001; D’Haeseleer,

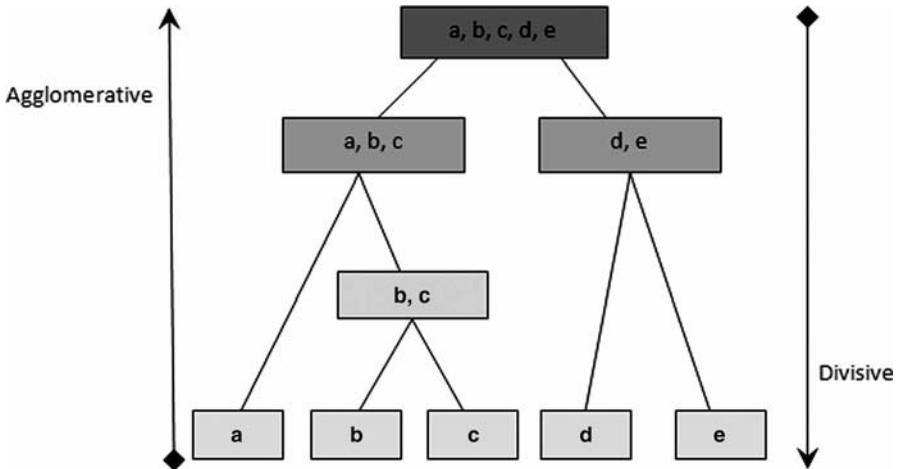


Figure 5.9 Two approaches in hierarchical clustering.

2005; Dopazo, 2006). We will confine our discussion to *agglomerative hierarchical clustering*.

As already mentioned, in every step in agglomerative clustering, the closest pairs of clusters are merged in order to optimize the overall cluster proximity. Based on the method used to determine the proximity between two clusters, agglomerative clustering algorithms can be differentiated into *single linkage*, *complete linkage*, and *average linkage* (an example of clustering results obtained using different linkage methods is presented in Fig. 5.11 below):

- In single-linkage clustering methods, the distance between two clusters is determined from the two closest objects in different clusters (shortest intercluster distance).
- In complete-linkage clustering methods, the distance between two clusters is defined as the distance of a pair of objects that are farthest apart in two clusters (farthest intercluster distance).
- In average-linkage clustering methods, the distance between two clusters is calculated as the average of the distances between all pairs of data objects in the two clusters (average intercluster distance). This type of linkage addresses the issue of sensitivity to outliers experienced by both single-linkage and complete-linkage clustering.

Based on average-linkage, agglomerative hierarchical clustering, Eisen et al. (1998) developed a clustering software package called Cluster and TreeView for gene expression data analysis. The output of the analysis is a two-dimensional dendrogram. Figure 5.10 explains how a nest of partitions leads to a graph representation—that is, dendrogram. The dendrogram provides data analysts the ability to detect the higher order relationships between clusters of profiles. The branches of a dendrogram record the formation of groups. The length of the horizontal branches indicates the similarity between the clusters.

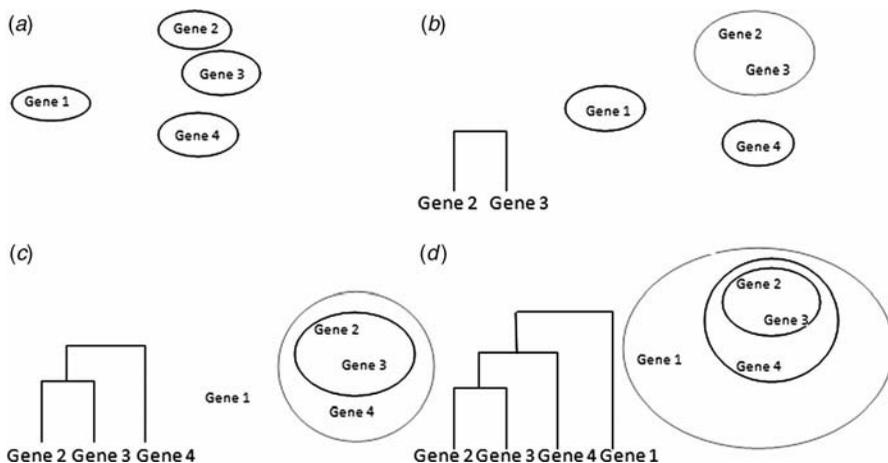


Figure 5.10 Agglomerative clustering and dendrogram generation: (a) four genes to be clustered; (b) after first merging; (c) after second merging; (d) after third merging.

The graphic representation of the results generated by hierarchical clustering allows users to visualize the global pattern in the data (Tseng, 2004), making this method a favorite among biologists. In the field of cancer research, hierarchical clustering has been used to identify cancer types (Nielsen et al., 2002; Ramaswamy et al., 2003), to discover new subtypes of cancer (Alizadeh et al., 2000), and to investigate cancer tumorigenesis mechanisms (Welch et al., 2002) from gene expression data. In addition, Mougeot et al. (2006) have applied hierarchical clustering for gene expression profiling of ovarian tissues. The results showed that hierarchical clustering can distinguish low malignant potential, early cancer, and possible precancerous stages. As shown in Figure 5.11, different clustering results generated on lung cancer data due to different linkage methods (Bhattacharjee et al. 2001).

However, several issues with hierarchical clustering still need to be addressed. The main problems can be summarized as follows: (i) lack of robustness to noise, high dimensionality, and outliers (Jiang et al., 2004); (ii) expensive in terms of computational time and space complexity (Xu and Wunsch, 2005). (iii) Once a decision is made for merging, it cannot be redone or optimized. To tackle the above problems, several alternative approaches have been proposed. To avoid the effects of noise and outliers, Ward's (1963) method changes the merging condition and merges the two clusters with the minimum change in the sum of squared error of the clusters before and after the merge. CURE (Clustering Using REpresentatives) (Guha et al., 1998) and BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies) (Zhang et al., 1996), the extension of hierarchical clustering methods, have appeared to handle large datasets and reduce the computational and space complexity.

K-Means *K*-means is a very straightforward, commonly used partitioning clustering method. In a typical implementation of the *K*-means algorithm, the first step is to randomly select *K* objects, each representing the initial cluster centroid. After the initialization step, each object in the dataset is assigned to the closest centroid, and a cluster

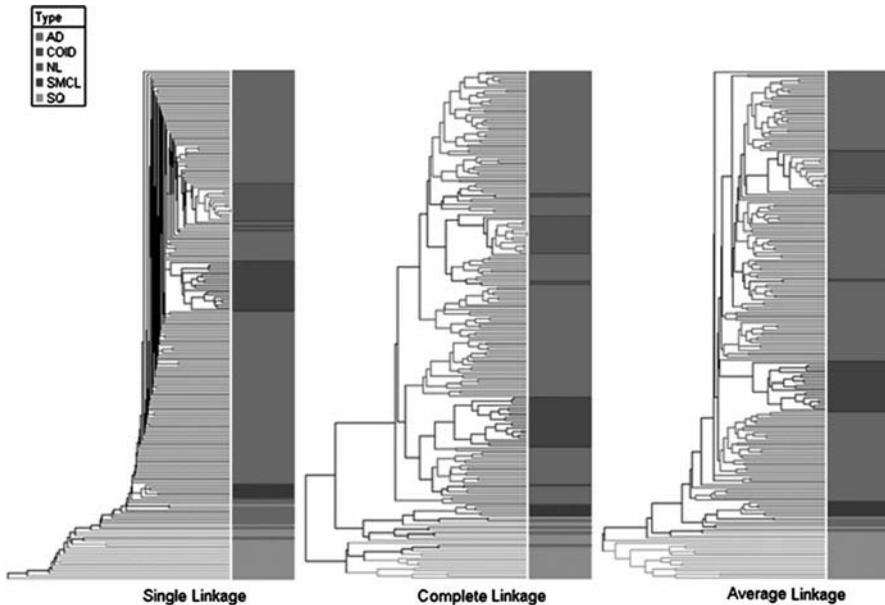


Figure 5.11 Agglomerative Hierarchical Clustering Algorithm with different linkage methods. The dataset used is the subset of the lung cancer dataset published by Bhattacharjee et al. (2001). The clustering method used is the agglomerative hierarchical clustering algorithms. The figure shows that different results generated due to different linkage methods. (See color insert.)

is then a collection of similar points. The centroid of each cluster is then recalculated based on the points assigned to the cluster. Iteratively, the assignment and update steps are repeated until the centroids remain stable. Figure 5.12 shows the general procedure of the K -means algorithm.

Tavazoie et al. (1999) used K -means to cluster the whole-genome mRNA data to identify transcriptional regulatory subnetworks in yeast. By iteratively relocating cluster members and minimizing the overall intercluster dispersion, 3000 genes were grouped into 30 clusters. With the gene coregulation information, biological significance of newly discovered genes from known genes and motifs was inferred. Shai et al. (2003) performed K -means clustering in order to identify molecular subtypes of gliomas. Three clusters corresponding to glioblastomas, lower grade astrocytomas, and oligodendrogliomas have been identified.

K -means is relatively scalable and efficient when processing large datasets as its time complexity is linear. In addition, K -means can reach a local optimum in a small number of iterations. However, K -means still suffers from several limitations, including the requirement to predefine the initial number of clusters K and the dependence of clustering result to the initial centroid selection. For biological data prior knowledge of the number of clusters is often not available. Thus, in order to detect the optimal number of clusters, an expensive fine-tuning process is necessary. Furthermore, K -means clustering is sensitive to noise and outliers. These imperfections are usually present in biological data and can substantially influence the update of cluster

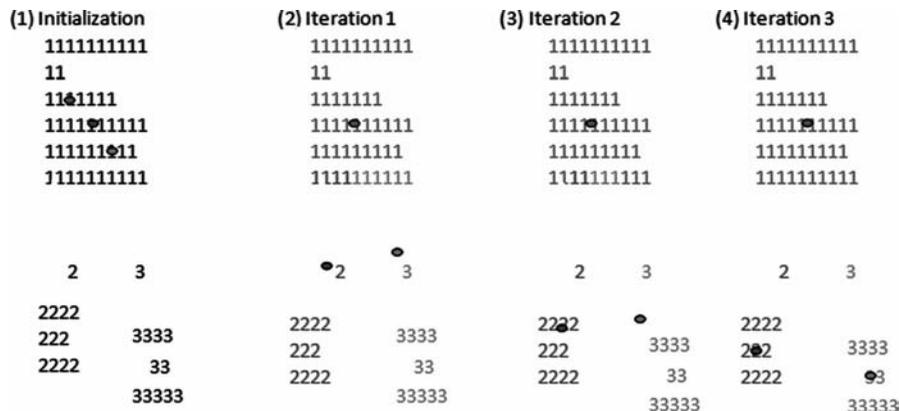


Figure 5.12 General procedure of K -means. The colored dots represent the centroids of clusters. To cluster the points into predefined three clusters: 1. The process is initialized by randomly selecting three objects as cluster centroids. 2. After the first iteration, points are assigned to the closest centroid, and cluster centroids are recalculated. 3. The iterative procedures of point reassignment and centroid update are repeated. 4. Centroids are stable; the termination condition is met. (See color insert.)

centroids and thus the final clustering results. Finally, K -means often converges at a local optimum, thus not necessarily reaching a global optimum.

To improve the robustness to noise and outliers, the K -medoid algorithm (Mercer and College, 2003) and the partitioning around medoids (PAM) algorithm (Kaufman and Rousseeuw, 1990) were introduced. A medoid is one of the data points that is selected by the algorithm as a representative point of a cluster. The application of medoid algorithm has two advantages: First, there is no limitation on attribute type; second, medoids are existing data points and thus, unlike cluster centroids, they are generated without any calculations (Berkhin, 2002).

To overcome the shortcoming of dependency on the initial number of clusters and cluster centroid selection, several researchers developed new algorithms to automatically detect the optimal number of clusters. Yeung and co-workers (2001a) applied probabilistic models to automatically detect the optimal number of clusters. Hastie et al. (2000) proposed the “gene shaving” method, a statistical method utilized to identify distinct clusters of genes.

One of the major criticisms of K -means is that it only converges to a local optimum. Several metaheuristic-based optimization techniques such as tabu search, simulated annealing, and genetic algorithm have been utilized to achieve near-global optimization. A series of hybridized algorithms combines the best features of both genetic algorithm and K -means to achieve near globally optimal solution with limited computational costs. These algorithms include genetic K -means (GKA) (Krishna and Narasimha Murty, 1999), fast genetic K -means (FGKA) (Lu et al., 2004b), and incremental genetic K -means (Lu et al., 2004a).

Self-Organizing Maps Self-organizing maps, first introduced by Kohonen (1984), represent an unsupervised neural network (NN)-based algorithm utilizing a

single-layered artificial neural network. As shown in Figure 5.13, the data objects are located at the input side of the network, and the output neurons are organized as two-dimensional $n \times m$ grids. Each neuron is associated with a weight known as a reference vector. The weight vectors of neurons are randomly initialized. Each data object acts as a learning example which directs the movement of the weight vectors toward the denser areas of the input vector space. Clustering is performed by having neurons compete for data objects. The neuron whose weight vector is closest to the current object becomes the winning unit. Then the weight vector of the best-matching neuron and its set of neighbors move toward the current object. While the training process proceeds, the adjustment of weight vectors is diminished. When the training is complete, the weight vectors trained to fit the distributions of the input dataset and clusters are identified by mapping all data objects to the output neurons.

SOM is a quantization method and its neural network structure gives it the ability to simplify and reduce the high dimensionality of the dataset (Wang et al., 2002). In addition, one of the appealing features of SOM is that it provides an intuitive view for mapping a high-dimensional dataset. The neuron learning process makes SOM more robust to noisy data (Jiang et al., 2004). Furthermore, SOM is efficient in handling a large-scale dataset since its time complexity is linear (Herrero and Dopazo, 2002). Toronen et al. (1999) developed and employed the software called GenePoint, a tree-based SOM and Sammon's mapping to analyze and visualize gene expression during a diauxic shift in yeast. Their work demonstrated that SOM is a reliable and

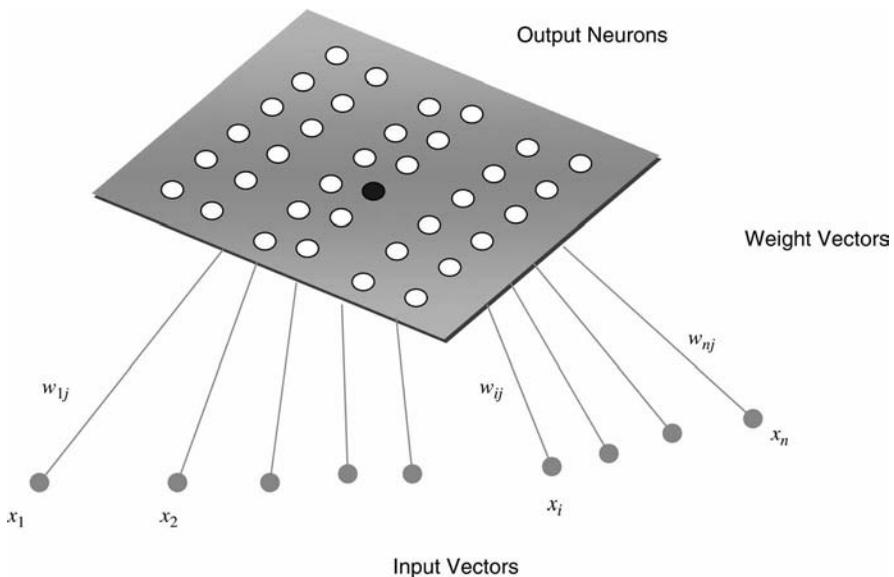


Figure 5.13 Topology of a SOM with 6×6 output neurons. The input vectors are connected to the output neurons (the connections to neuron j is shown in the figure). The output layer in the SOM structure displays the similarity of pattern so that the similar patterns are adjacent, and different ones are well separated.

fast analysis tool for analyzing and visualizing gene expression profiles. Tamayo et al. (1999) implemented the software named Genecluster based on SOM (currently available as part of the GenePattern software at <http://www.broad.mit.edu/cancer/software/genepattern/>). Another application of SOM was developed by Vesanto and co-workers as a SOM Toolbox for Matlab (<http://www.cis.hut.fi/projects/somtoolbox/>). The example presented in Figure 5.14 shows results of sample SOM clustering based on the data for expression of several genes using Matlab SOM Toolbox.

Like other conventional clustering algorithms, SOM has its own pitfalls. In order to perform clustering, users have to predefine the initial number of the grids and the topology of the output neurons. The clustering results are closely related to the parameters such as the learning rate and the topology of the neurons. SOM can only

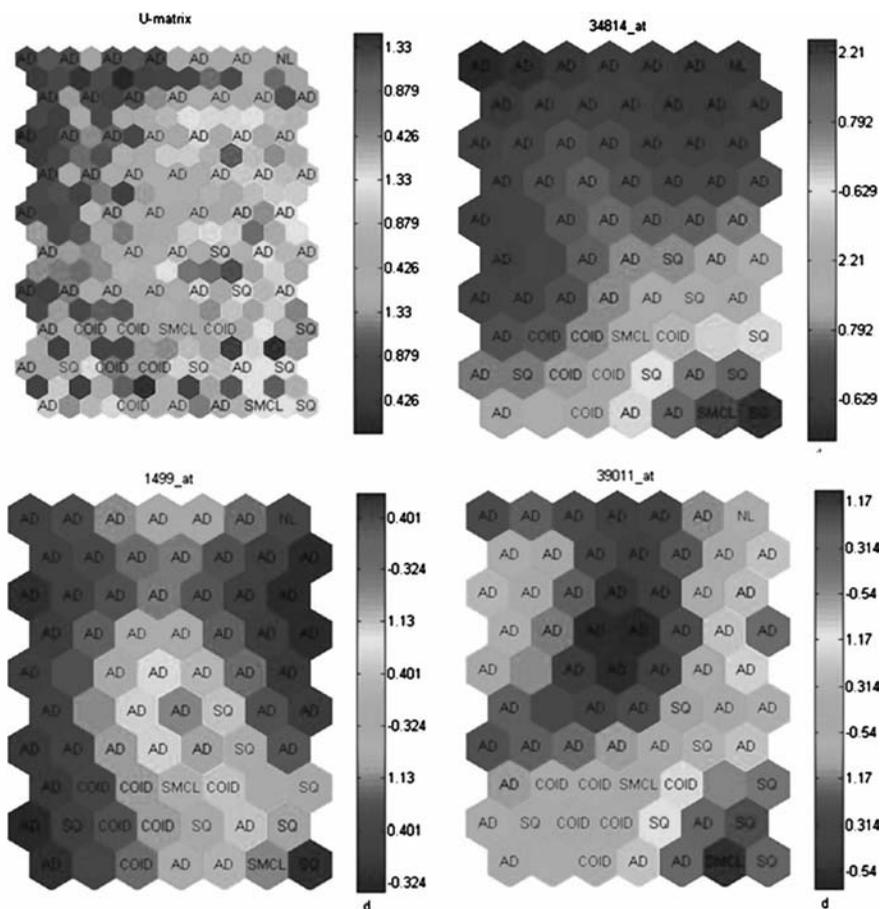


Figure 5.14 SOM Toolbox clustering of lung cancer samples using gene expression data for three representative genes measured by Bhattacharjee et al. (2001). (See color insert.)

reach a local optimum if the initial weights are not chosen properly (Jain et al., 1999). In the case of processing large-scale biological data, SOM is prone to problems when handling the data abundant in irrelevant and invariant data (Jiang et al., 2004). Those data can dominate the majority of clusters. As a result, most of the meaningful patterns might be lost.

To improve the performance of SOM, a number of variants have been proposed. Within them, Su and Chang (2001) introduced a novel model called double SOM (DSOM). In DSOM, each neuron is related not only to an n -dimensional weight vector but also to a two-dimensional position vector. During the learning process, both weight and position vectors are updated. By plotting the number of groups of two-dimensional position vectors, the number of clusters can be determined. Inspired by the idea of integrating merits of both Hierarchical clustering and SOM, the self-organizing tree algorithm (SOTA) (Dopazo and Carazo, 1997; Herrero et al., 2001) has been proposed and implemented. SOTA combines the best features of both SOM and the *divisive-based hierarchical clustering*. SOTA maps the output neurons (nodes) in the topology of a binary tree instead of a two-dimensional grid. The number of nodes in SOTA is not fixed and the tree structure of the nodes grows during the clustering procedure. As indicated in Dopazo (2006), the main difference between SOTA and SOM is that the neurons are directly updated via the changes of the tree structure. After the convergence of the network, the neuron with the most population of data objects is split into two sister nodes, causing the binary tree to grow. SOTA stops when a threshold of variability is reached for each neuron. In this way, the number of clusters is not required to be predefined. The threshold variability can be determined by means of permutation tests on the dataset. Xiao et al. (2003) recommended a hybrid clustering approach which applied both SOM and a simple evolutionary method, particle swarm optimization (PSO) (Kennedy and Eberhart, 1999). The hybrid clustering approach deploys PSO to evolve the weights for SOM. The rate of convergence is improved by introducing a conscience factor to the SOM.

5.2.2.2 Advanced Clustering Algorithms The limitations of the conventional clustering algorithms can be summarized as follows:

1. *Requirement to Predefine Number of Clusters:* For both K -means and SOM, the decision on the number of clusters has to be made before the clustering starts. For hierarchical clustering, it is necessary to determine where to cut the tree. However, in order to find the optimal number of clusters, an expensive fine-tuning process is required.
2. *Vulnerability to Noisy Data and Outliers:* In the conventional clustering algorithms, cluster proximity is measured by distance metrics. Outliers and high level of noise, often present in the biological data, can substantially influence the calculation of cluster centroids.
3. *Lack of Robustness to High Dimensionality of Biological Data:* High-dimensional data require clustering algorithms to take more time and computational space to process.

4. *Crisp Clustering Concept*: The conventional clustering algorithms are based on the assumption that each data object belongs to only one cluster. However, the restriction of one-to-one mapping might not be suitable to biological data analysis. Current thinking in molecular biology holds that, for example, genes interact with each other in different pathways (Sheng et al., 2005). Hence, when clustering gene expression data, it would be desirable to take into account that a gene might be directly or indirectly involved in several pathways.
5. *One-Way Clustering*: As mentioned before, conventional clustering algorithms utilize one-way clustering, where the clustering process is applied separately to either the rows or the columns of the data matrix. However, we should keep in mind that discovering local gene expression patterns would be more appropriate under the following scenarios: (a) In any cellular processes of interest, only subsets of genes are coregulated or coexpressed under certain experimental conditions. (b) A gene can participate in multiple pathways that may or may not be coactive under all conditions.

Although the limitations we have listed here are not exhaustive, they give rise to the clear directions for the development of clustering algorithms tailored to biological data analysis. To address the issues in conventional clustering algorithms, a large number of clustering methods have been proposed in the literature. Within them, model-based clustering, fuzzy clustering, and biclustering are the most practical techniques in biological data analysis. Since the detailed description of these algorithms is beyond the scope of this chapter, we will focus our discussion on the merits of such advanced clustering algorithms in biological data analysis compared to conventional methods.

Fuzzy Clustering Fuzzy clustering provides one-to-many mapping where a single feature belongs to multiple clusters with a certain degree of membership. The memberships can further be used to discover more sophisticated relations between the data and its disclosed clusters (Xu and Wunsch, 2005). A fuzzy approach is more desirable in situations where a majority of features, such as genes, participate in different genetic networks and are governed by a variety of regulatory mechanisms. Furthermore, fuzzy clustering is robust to the high level of noise often present in omics data.

Fuzzy C-means (FCM) is the most popular fuzzy clustering algorithm in biological data analysis. It considers each gene as a member of all clusters with different degrees of membership. Membership of a gene is closely related to the similarity or distance between the gene and a given centroid. High similarity between a gene and the closest centroid indicates a strong association to the cluster, and its degree of membership is close to 1; otherwise, its membership value is close to 0. Fuzzy logic provides a systematic and unbiased way to transform precise, numerical values into qualitative descriptors via a fuzzification process (Woolf and Wang, 2002). The fuzzification process provides more flexibility for data analysis when no prior knowledge is available. In the literature, FCM has been extensively applied to select genes that are correlated to multiple clusters and to unravel complex regulatory pathways that control the expression pattern of genes. Belacel et al. (2004) embedded

FCM into a variable neighborhood search (VNS) metaheuristic to improve the performance of conventional FCM. This method has been tested on four microarray datasets and provides superior accuracy in clustering the data. Wang et al. (2003) used FCM to perform tumor classification and marker gene prediction. Dembélé and Kastner (2003) applied FCM for clustering microarray data by assigning membership values to genes. Their work has elucidated that FCM clustering provides a convenient way to define subsets of genes that exhibit tight association with given clusters. An example of the result on clustering samples from gene expression values using FCM is presented in Figure 5.15. From the membership values it is possible to get more detailed information about sample subtypes and similarities.

Model-Based Clustering Model-based clustering is based on statistical mixture models. It assumes that the data are generated by a finite number of underlying probability distributions (e.g., Gaussian distribution) with each component corresponding to a distinct cluster (Yeung et al., 2001a). The problem associated with model-based Clustering, when processing biological data, is to find each feature with the best underlying distribution in the mixture and at the same time to estimate the parameters for each of these distributions (Sheng et al., 2005). Like fuzzy clustering, model-based clustering allows each data object to belong to multiple clusters with certain probability. This approach facilitates the identification of overlapped groups of genes under conditional coregulations.

The *EM-based clustering* is one of the well-known statistical mixture models, first proposed by Dempster et al. (1977). The results of *EM clustering* are different

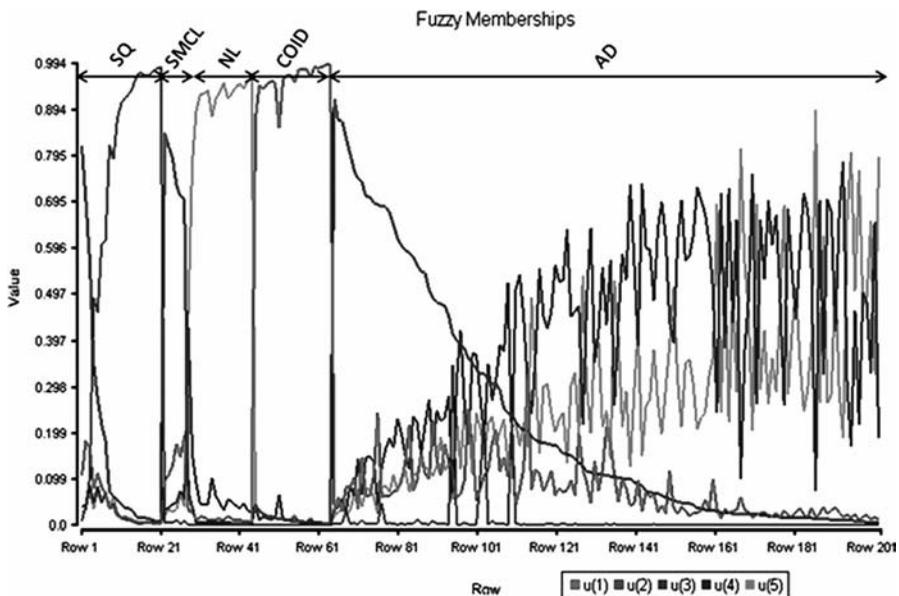


Figure 5.15 Membership values obtained from FCM clustering on lung cancer samples based on gene expression information. Samples were clustered in five groups. (See color insert.)

from those generated by the *K-means clustering*. While the *K-means* algorithm assigns data objects to clusters by trying to maximize the distance between clusters, EM clustering computes the classification probabilities, that is, the probabilities of assigning each data object to each cluster. When the EM clustering algorithm converges, each data object is assigned to the cluster with the maximum classification probability.

The EM-based clustering has been widely used in gene expression data analysis. McLachlan et al. (2002) applied its principle in a software EMMIX-GENE. Mar and McLachlan (2003) successfully deployed EMMIX-GENE to cluster breast cancer samples on the basis of gene expressions. Experimental results showed that model-based clustering can efficiently reduce a large number of genes to a more manageable size, making it more appealing for classification of cancer tissue samples. Yeung et al. (2001a) applied model-based clustering to three gene expression datasets. Their work showed that model-based clustering is an alternative option for the heuristic algorithms in terms of determining the optimal number of clusters.

Biclustering *Biclustering* addresses the challenge of exploring coherent subspace clusters within the gene expression matrices by performing simultaneous clustering on both rows and columns. Unlike the *one-way clustering* algorithms, biclustering approaches identify a subset of genes that share similar activity patterns under a subset of the experimental conditions. In addition, biclustering overcomes several restrictions of the conventional clustering algorithms by allowing the gene and/or sample to belong to more than one cluster or not to be clustered at all and to be grouped using a subset of genes and/or samples (Madeira and Oliveira, 2004). Biclustering techniques become most desirable to use when the following situations apply (Madeira and Oliveira, 2004):

- Only a subset of genes participates in a cellular process of interest.
- A cellular process of interest is only active in a subset of conditions.
- A gene may participate in multiple pathways that may be coregulated under a subset of conditions.

Cheng and Church (2000) were the first to introduce biclustering to gene expression data analysis. In Cheng and Church's algorithm, the biclustering problem is treated as an optimization problem, defining a subset of genes and conditions with a high similarity score. This similarity score is called *mean-squared residue*. The algorithm identifies one bicluster at a time, defines a similarity score for the candidate bicluster, and develops heuristics to solve the constrained optimization problem and find other biclusters. To address the complexity issue, Getz et al. (2000) proposed the coupled two-way clustering (CTWC) algorithm aimed at identifying a set of biclusters at one time. The algorithm repeatedly performs one-way hierarchical clustering in order to discover significant stable clusters by introducing a generic scheme for transforming one-way clustering into two-way clustering. In this process two-way clustering will recursively apply one-way clustering in order to find subsets of genes from stable condition clusters and subsets of conditions using stable gene clusters. The final biclusters are those stable submatrices. To enhance biclustering performance, several combinatory biclustering algorithms such as the statistical algorithm method

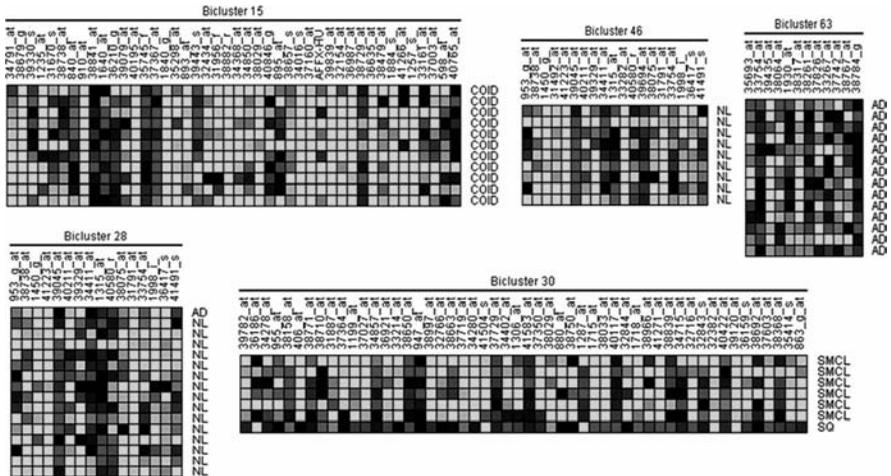


Figure 5.16 Result of SAMBA biclustering on lung cancer data. The different clusters show the sample clustering as well as the subsets of genes that are most relevant for the distinction of any given sample subgroup. (See color insert.)

for bicluster analysis (SAMBA) (Tanay et al., 2002), a combination model of biclustering and probabilistic graph theory, the Plaid model (Lazzeroni and Owen, 2002), a mixtures of normal distribution, and the iterative signature algorithm (ISA) (Ihmels et al., 2002; Bergman et al., 2003) have been proposed. An example of the application of the SAMBA method for the clustering of a lung cancer dataset is shown in Figure 5.16.

Biclustering approaches have been extensively applied in biological data analysis with particularly useful results in (i) Identification of coregulated genes; (ii) automatic gene functional annotation; and (iii) sample/tissue classification for disease diagnosis (Madeira and Oliveira, 2004).

5.3 ASSESSMENT OF CLUSTER QUALITY

Given the same dataset, different choices of preprocessing, clustering algorithms, and distance measures could lead to varied clustering results. Therefore, the assessment of cluster validity is of utmost important. However, in practice, the cluster quality is hard to evaluate, particularly in the analysis of biological data.

Cluster validity is a measure of correspondence between a cluster structure and the data within the structure (Mirkin, 2005). The adequacy of a clustering structure refers to the sense in which the clustering structure provides true information about the data (Jain and Dubes, 1988). The validity of a clustering structure can be expressed based on three different criteria (Jiang et al., 2004):

- *Internal Measures:* Internal criteria assess the fit between the clustering structure and the data without prior knowledge. Internal measures are the most appropriate cluster validity criteria for unsupervised clustering. Many internal measures of

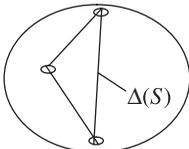
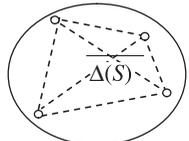
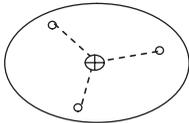
cluster validity are based on the notions of cohesion or separation: the measures of cluster compactness, which determine how cohesive the objects within a cluster are, and measures of cluster separation, which decide how well separated a cluster is from other clusters (Tan et al., 2006).

- *External Measures:* External criteria measure clustering performance by matching clustering structure to prior knowledge. In the case of biological data analysis, prior knowledge could be expert knowledge, such as known functional families of genes or known clinical diagnosis of normal or cancer tissues (Jiang et al., 2004). This type of cluster validity is based on the agreement between clustering results and prior knowledge (Jiang et al., 2004).
- *Measures for Predictive Strength and Cluster Stability:* This type of cluster validity focuses on the reliability of clusters, that is, whether cluster structure was predicted by chance (Jiang et al., 2004). In addition, the analysis of predictive power, particularly in biological data analysis, is performed by investigating whether clustering results obtained from one set of experimental condition can be related to the ones from another set (Zhao and Karypis, 2005).

5.3.1 Internal Measures

Internal measures are unsupervised measures of cluster validity, performed for the analysis of cluster quality in data without prior knowledge. This type of measure can be further divided into two classes: measures of cluster cohesion (compactness) and measures of cluster separation.

TABLE 5.2 Intracluster Measures

Measure	Description	Formula	Graph-based view
Complete diameter	Distance between most remote pairwise objects within a cluster	$\Delta(S) = \max_{x,y \in S} \{d(x, y)\}$ where $x, y =$ pairwise objects within cluster S	
Average diameter	Average distance between all objects within a cluster	$\overline{\Delta(S)} = \frac{1}{ S \cdot (S - 1)} \sum_{\substack{x,y \in S \\ x \neq y}} d(x, y)$ where $x, y =$ pairwise objects within cluster S $ S =$ number of objects within cluster S	
Centroid diameter	Average distance between all objects within a cluster and the cluster centroid	$\Delta(S) = 2 \left(\frac{\sum_{x \in S} d(x, \bar{c})}{ S } \right)$ where $x =$ object within cluster S $\bar{c} =$ centroid of cluster S $ S =$ number of objects within cluster S	

There are several forms of validity measures assessing cluster cohesion or compactness via intracluster variance. The most popular representative of this approach is the variance of the sum-of-squared errors (Handl et al., 2005). A number of alternatives include measuring intracluster homogeneity via either the similarities between pairwise objects within a cluster or the centroid-based similarities (Handl et al., 2005). Table 5.2 provides detailed description for some well-known intracluster measures. Cluster separation can be quantified by the dissimilarity measure between two clusters. Similarly, the dissimilarity can be evaluated by the intercluster distance. The intercluster distance can be computed as the distance between cluster centroids, or as the average weighted intercluster distances, or as the minimum/maximum distances between objects in different clusters (similar to distance calculations previously described for cluster analysis). Table 5.3 lists the most commonly used intercluster measures.

However, there is an opposite trend between intracluster homogeneity and cluster separation: “while intra-cluster homogeneity improves with an increasing number of clusters, the distance between clusters tends to deteriorate” (Handl et al., 2005, p. 3204). Therefore, several techniques have been proposed to combine the above two classes of measures into one. The most well-known measures in biological data analysis include the Dunn index (Dunn, 1974), the Davies–Bouldin index (Davies and Bouldin, 1979), and the silhouette width (Rousseeuw, 1987), summarized in Table 5.4.

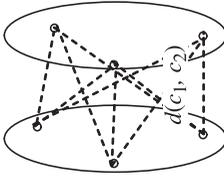
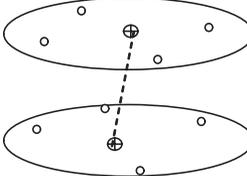
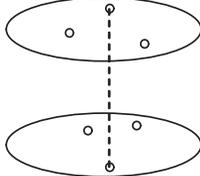
5.3.2 External Measures

Standard external measures take both clustering result and known-class labels (prior knowledge or ground truth) as inputs and then compare these inputs to assess the degree of consensus.

Given a clustering result $U: c_1, c_2, \dots, c_n$ and the ground truth $P: L_1, L_2, \dots, L_m$, one can build the binary matrices for the clustering result and the ground truth, respectively. For instance, we have a dataset with six data objects, $p_1, p_2, p_3, p_4, p_5, p_6$. The current clustering result shows that these six data objects have been grouped into two clusters: $c_1 = \{p_1, p_2, p_3\}$ and $c_2 = \{p_4, p_5, p_6\}$. The ground truth indicates there are two classes and the distribution of data objects are as follows: $L_1 = \{p_1, p_2, p_3, p_4\}$ and $L_2 = \{p_5, p_6\}$. We can construct binary matrices for both the clustering result and the ground truth (Tables 5.5 and 5.6) following the rules: the entry is assigned 1 if two objects i, j are in the same cluster and 0 otherwise. Subsequently, the agreement between U and P can be determined with the following conditions:

- n_{11} is the number of object pairs having the same cluster and the same class, that is, $U_{ij} = 1, P_{ij} = 1$ in the binary matrices;
- n_{10} is the number of object pairs having the same cluster but a different class, that is, $U_{ij} = 1, P_{ij} = 0$ in the binary matrices;
- n_{01} is the number of object pairs having a different cluster but the same class, that is, $U_{ij} = 0, P_{ij} = 1$ in the binary matrices; and
- n_{00} is the number of object pairs having a different cluster and a different class, that is, $U_{ij} = 0, P_{ij} = 0$ in the binary matrices.

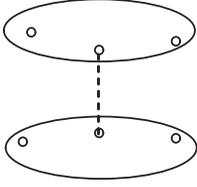
TABLE 5.3 Intercluster Measures

Measure	Description	Formula	Graph-based view
Average linkage	Average distance between all of belonging to two different clusters	$d(c_1, c_2) = \frac{1}{n_1 n_2} \sum_{x \in c_1, y \in c_2} d(x, y)$ <p>where $x, y =$ objects in clusters c_1 and c_2, respectively $n_1, n_2 =$ numbers of objects included in clusters c_1, c_2, respectively</p>	
Centroid linkage	Distance between centroids of two clusters	$d(c_1, c_2) = d(\bar{c}_1, \bar{c}_2)$ <p>where $\bar{c}_1 = (1/n_1) \sum_{x \in c_1} x$, the centroid of cluster c_1 $x =$ object in cluster c_1 $n_1 =$ number of objects in cluster c_1 $\bar{c}_2 = (1/n_2) \sum_{y \in c_2} y$, the centroid of cluster c_2 $y =$ object in cluster c_2 $n_2 =$ number of objects in cluster c_2</p>	
Complete linkage	Distance between the most remote objects belonging to two different clusters	$d(c_1, c_2) = \max_{x \in c_1, y \in c_2} \{d(x, y)\}$ <p>where $x, y =$ objects in cluster c_1 and c_2</p>	

Single linkage

Distance between closest objects belonging to two different clusters

$d(c_1, c_2) = \min_{x \in c_1, y \in c_2} \{d(x, y)\}$
 where $x, y =$ objects in clusters c_1, c_2



Average to centroid linkage

Distance between cluster centroid and all objects in different cluster

$$d(c_1, c_2) = \frac{1}{n_1 + n_2} \left(\sum_{x \in c_1} d(x, \bar{c}_2) + \sum_{y \in c_2} d(y, \bar{c}_1) \right)$$

where $x =$ object in cluster c_1
 $y =$ object in cluster c_2

$n_1, n_2 =$ number of objects in clusters c_1, c_2
 $\bar{c}_1, \bar{c}_2 =$ centroids for clusters c_1, c_2

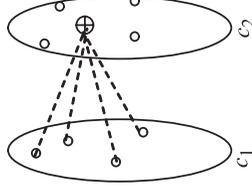


TABLE 5.4 Combinatory Measures for Cluster Validity

Cluster validity	Formula	Description
Dunn's index	$D = \min_{1 \leq i \leq n} \left\{ \min_{1 \leq j \leq n, i \neq j} \left(\frac{d(c_i, c_j)}{\max_{1 \leq k \leq n} [d'(c_k)]} \right) \right\}$ <p>where $d(c_i, c_j)$ = intercluster distance between cluster c_i and c_j</p> <p>$d'(c_k)$ = intracluster distance</p> <p>n = number of clusters</p>	Dunn's index is designed to identify how compact and well separated the resulting clusters are. The number of clusters with maximum Dunn's index value is taken as the optimal number of clusters.
Davies–Bouldin index	$DB = \frac{1}{n} \sum_{i=1}^n R_i$ <p>where n = number of clusters</p> <p>$R_i = \max_{1 \leq j \leq n} (R_{ij})$, maximum value of similarity between cluster i and any cluster j in current partition</p> <p>$R_{ij} = (S_i + S_j)/d_{ij}$</p> <p>$S_i$ = average distance between all objects in cluster i to its cluster centroid</p> <p>S_j = average distance between all objects in cluster j to its cluster centroid</p> <p>d_{ij} = distance between centroids of clusters i and j</p>	The Davies–Bouldin index is based on the dispersion measure of a cluster and the cluster dissimilarity measure. The Davies–Bouldin index has a small value for a good clustering.
Silhouette width	<p>For a given cluster $c_j (j = 1, 2, \dots, n)$:</p> <p>1. the silhouette width for the ith sample in cluster c_j is</p> $s_i = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}$ <p>where $a(i)$ = average distance between ith sample and all samples in c_j</p> <p>$b(i)$ = minimum average distance between ith sample and samples in c_j</p> <p>$i \in \{1, 2, \dots, n\}$, n = number of clusters;</p> <p>m = number of samples (or objects) in a cluster</p>	The silhouette width calculates the Silhouette width for each sample, the average silhouette width for each cluster, and the global average silhouette width for a partition. The largest global silhouette width indicates the best partition.

(Continued)

TABLE 5.4 *Continued*

Cluster validity	Formula	Description
	2. the silhouette width for a cluster is	
	$S_j = \frac{1}{m} \sum_{i=1}^m s(i)$	
	where m = number of samples (or objects) in a cluster	
	$s(i)$ = silhouette width for a sample	
	3. For any partition $U: c_1, c_2, \dots, c_n$, the global silhouette width is	
	$GS_U = \frac{1}{n} \sum_{j=1}^n S_j$	
	where n = number of clusters in a partition	
	S_j = silhouette width for cluster	

TABLE 5.5 Similarity Matrix for Clustering Result

Points	p_1	p_2	p_3	p_4	p_5	p_6
p_1	1	1	1	0	0	0
p_2	1	1	1	0	0	0
p_3	1	1	1	0	0	0
p_4	0	0	0	1	1	1
p_5	0	0	0	1	1	1
p_6	0	0	0	1	1	1

Note: The clustering result: two clusters: $c_1 = \{p_1, p_2, p_3\}$ and $c_2 = \{p_4, p_5, p_6\}$.

TABLE 5.6 Similarity Matrix for Ground Truth

Points	p_1	p_2	p_3	p_4	p_5	p_6
p_1	1	1	1	1	0	0
p_2	1	1	1	1	0	0
p_3	1	1	1	1	0	0
p_4	0	0	0	0	1	1
p_5	0	0	0	0	1	1
p_6	0	0	0	0	1	1

Note: The ground truth is: $L_1 = \{p_1, p_2, p_3, p_4\}$ and $L_2 = \{p_5, p_6\}$.

A number of indices have been introduced to determine the degree of agreement between U and P . The most commonly used ones are

$$\text{Rand} = \frac{n_{11} + n_{00}}{n_{11} + n_{10} + n_{01} + n_{00}} \quad (5.14)$$

$$\text{Jaccard} = \frac{n_{11}}{n_{11} + n_{10} + n_{01}} \quad (5.15)$$

$$\text{Minkowski} = \sqrt{\frac{n_{10} + n_{01}}{n_{11} + n_{01}}} \quad (5.16)$$

As observed, the *Rand index* assesses the degree of similarity between U and P via a function of positive and negative agreements in the binary matrices while Jaccard and Minkowski indices ignore the term n_{00} . The common understanding in gene-based clustering suggests that a majority of genes would distribute in separate clusters and the term n_{00} would dominate over the other three terms in the final results (Jiang et al., 2004). Obviously, Jaccard and Minkowski indices are the most appropriate to address this issue.

5.3.3 Measures for Predictive Strength and Cluster Stability

The measure presented in this section is performed by repeatedly resampling or perturbing the original dataset and then reclustering the data into new clustering results. The assessment of cluster quality (cluster stability) is then determined from the consistency of the consecutive results. In the case of gene expression data analysis, Tavazoie et al. (1999) used the hypergeometric distribution to obtain the chance probability of observing the number of genes from a particular functional category within each cluster. More specifically, the authors calculated the p -value for each cluster showing the probability of observing a predefined number of genes from a functional category within a cluster of certain size. The p -value is used to measure the statistical significance in terms of functional category enrichment (Tavazoie et al., 1999). Later, Jakt and co-workers (2001) integrated the measure of this type to assess the functional significance between gene clusters and the corresponding postulated regulatory motifs.

The *figure of merit* (FOM) (Yeung et al., 2001b) compares outputs of different clustering algorithms in terms of their predictive power and homogeneity. FOM is a combination of the jackknife approach and the leave-one-out cross-validation. The key idea of this method is that if a cluster of genes formed with respect to a set of samples has possible biological significance, then the expression levels of the genes within this cluster should also be similar to each other in “test” samples that were not used to form the cluster. Thus in the test clustering is applied to all experimental conditions except one, the left-out condition. If the algorithm performs well, the clusters generated from the remaining conditions should be highly consistent with the results for the left-out condition.

Measures of this type are still an active research area. We will not provide a detailed description here. The interested reader can follow the references for details.

5.4 CONCLUSION

In this chapter, we have introduced different steps of the clustering process: (1) similarity measures, (2) grouping approaches, and (3) cluster validity analysis. Also we have discussed the applications of well-known clustering methods in life and biomedical sciences. The goal of clustering is to provide an initial identification of groups of highly correlated data that can be further investigated for functions and properties. Recently several new clustering algorithms (e.g., graph-theoretical clustering, model-based clustering) have been developed with the intention of combining and improving the feature of traditional clustering algorithms. However, clustering algorithms are based on different assumptions, and the performance of each algorithm depends on properties of the input dataset. Therefore, the winning clustering algorithm does not exist for all data sets, and optimization of existing clustering algorithms is still a vibrant research area. It is essential to keep in mind that the optimal method has to be determined for a problem at hand and that it is crucial to test the quality of clusters obtained using different clustering tools in combination with different data preprocessing routines. Ultimately, clustering of biological data is performed in order to obtain certain biological knowledge either about samples and systems or biological molecules, and the most optimal method is the one that provides accurate results and allows extraction of this useful information. Based on clustering algorithms, many commercial and open-source software tools have been developed for clustering of high-throughput biological data, designed either to perform specific analysis or to provide the whole set of data analysis steps, including data preprocessing, dimensionality reduction, normalization, clustering, classification, and visualization (Belacel et al., 2006). The well-known open source used for biological data analysis is the Bioconductor toolkit, which is based on the statistical language R. R programming with the Bioconductor packages offers the best solution for data analysis. R is an open-source re-creation of the language S-Plus (www.r-project.org). The Bioconductor is a collection of R-packages for the analysis of high-throughput biological data. Extensive documentation on how to perform some of clustering methods described in this book are available from the Bioconductor website (www.bioconductor.org).

REFERENCES

- Alizadeh, A., et al. (2000). Distinct types of diffuse large B-cell lymphoma identified by gene expression profiling. *Nature*, **403**: 503–511.
- Altschul, S. F., et al. (1990). Basic local alignment search tool. *J. Mol. Biol.*, **215**(3): 403–410.
- Altschul, S. F., et al. (1997). Gapped BLAST and PSI-BLAST: A new generation of protein database search programs. *Nucleic Acids Res.*, **25**(17): 3389–3402.
- Belacel, N., Cuperlovic-Culf, M., Laflamme, M., and Ouellette, R. (2004). Fuzzy J-means and VNS methods for clustering genes from microarray data. *Bioinformatics*, **20**(11): 1690–1701.
- Belacel, N., Wang, Q., and Cuperlovic-Culf, M. (2006). Clustering methods for microarray gene expression data. *OMICS*, **10**(4): 507–531.
- Ben-Dor, A., Chor, B., Karp, R., and Yakhini, Z. (2002). Discovering local structure in gene expression data: The order-preserving submatrix problem. In *Proc. 6th Int. Conf. Computational Biology (RECOMB 02)*, April 18–21, 2002, Washington, DC, USA. ACM, ISBN 1-58113-498-3-02104, pp. 49–57.

- Bergman, S., Ihmels, J., and Barkai, N. (2003). Iterative signature algorithm for the analysis of large-scale gene expression data. *Phys. Rev. E Stat. Nonlin. Soft Matter Phys.*, **67**(3 Pt. 1): 03190201–18.
- Berkhin, P. (2002). Survey of clustering data mining techniques. Available: <http://www.stats.ox.ac.uk/~mercer/documents/Transfer.pdf>, accessed March 6, 2008.
- Bhattacharjee, A., et al. (2001). Classification of human lung carcinomas by mRNA expression profiling reveals distinct adenocarcinoma subclasses. *PNAS*, **98**: 13790–13795.
- Bobrowski, L., and Bezdek, J. (1991). c-Means clustering with the l_1 and l_∞ norms. *IEEE Trans. Syst. Man Cybernet.*, **21**(3): 545–554.
- Cheng, Y., and Church, G. M. (2000). Biclustering of expression data. In *Proc. 8th Int. Conf. Intelligent Systems for Molecular Biology (ISMB '00)*, Lajolla/San Diego, CA, USA, AAAI2000, ISBN 1-57735-115-0, pp. 93–103.
- Davies, D. L. and Bouldin, D. W. (1979). Cluster separation measure. *IEEE Trans. Pattern Anal. Machine Intell.*, **1**(2): 95–104.
- Dayhoff, M. O., Schwartz, R. M., and Rubin, D. B. (1978). A model of evolutionary change in proteins. *Atlas Prot. Seq. Struct.*, **5**: 345–352.
- Dembélé, D., and Kastner, P. (2003). Fuzzy C-means for clustering microarray data. *Bioinformatics*, **19**: 973–980.
- Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *J. R. Stat. Soc. B*, **39**: 1–38.
- DeSmet, F., et al. (2002). Adaptive quality-based clustering of gene expression profiles. *Bioinformatics*, **18**: 735–746.
- D’Haeseleer, P. (2005). How does gene expression clustering work? *Nat. Biotechnol.*, **23**: 1499–1501.
- Dietmann, S., et al. (2001). A fully automatic evolutionary classification of protein folds: Dali Domain Dictionary version 3. *Nucleic Acids Res.*, **29**: 55–57.
- Dopazo, J. (2006). Clustering—Class discovery in the post-genomic era. In *Fundamentals of Data Mining in Geonomics and Proteomics*, W. Dubitzky, M. Granzow, and D. P. Berrar, (Eds.), New York: Springer Science + Business Media, pp. 123–148.
- Dopazo, J., and Carazo, J. M. (1997). Phylogenetic reconstruction using an unsupervised growing neural network that adopts the topology of a phylogenetic tree. *J. Mol. Evol.*, **44**: 226–233.
- Dunn, J. C. (1974). Well separated clusters and optimal fuzzy partitions. *J. Cybernet.*, **4**: 95–104.
- Durbin, R., Eddy, S., Krogh, A., and Mitchison, G. (1998). *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge: Cambridge University Press.
- Eisen, M. B., Spellman, P. T., Brown, P. O., and Botstein, D. (1998). Cluster analysis and display of genome wide expression patterns. *PNAS*, **95**: 14863–14868.
- Futschik, M. E., and Kasabov, N. K. (2002). Fuzzy clustering of gene expression data. In *Proceedings of the World Congress of Computational Intelligence (WCCI)*, Hawaii 2002, IEEE Press, Vol. 1, pp. 414–419.
- Getz, G., Levine, E., and Domany, E. (2000). Coupled two-way clustering analysis of gene microarray data. *PNAS*, **97**: 12079–12084.
- Gibrat, J. F., Madej, T., and Bryant, S. H. (1996). Surprising similarities in structure comparisons. *Curr. Opin. Struct. Biol.*, **6**: 377–385.
- Guha, S., Rastogi, R., and Shim, K. (1998). CURE: An efficient clustering algorithm for large database. In *Proceedings of the 1998 ACM SIGMOD International Conference of Management of Data*, Seattle, Washington, USA, ACM-SIGMOD Press, pp. 73–84.
- Handl, J., Knowles, J., and Kell, D. B. (2005). Computational cluster validation in post-genomic data analysis. *Bioinformatics*, **21**(15): 3201–3212.
- Hastie, T., et al. (2000). “Gene shaving” as a method for identifying distinct sets of genes with similar expression patterns. *Genomic Biol.* **1**(2): 1–21.
- Henikoff, S., and Henikoff, J. G. (1992). Amino acid substitution matrices from protein blocks. *PNAS*, **89**: 10915–10919.
- Herrero, J., and Dopazo, J. (2002). Combining hierarchical clustering and self-organizing maps for exploratory analysis of gene expression patterns. *J. Proteome Res.*, **1**: 467–470.
- Herrero, J., Valencia, A. and Dopazo, J. (2001). A hierarchical unsupervised growing neural network for clustering gene expression patterns. *Bioinformatics*, **17**(2): 126–136.

- Huang, Z. (1997). A fast clustering algorithm to cluster very large categorical data sets in data mining. In *Proceedings of ACM SIGMOD Workshop on Data Mining and Knowledge Discovery*, May 11, 1997, pp. 146–151.
- Ihmels, J., et al. (2002). Revealing modular organization in the yeast transcriptional network. *Nat. Genet.*, **31**(4): 370–377.
- Jain, A. K., and Dubes, R. C. (1988). *Algorithms for Clustering Data*. Englewood Cliffs, NJ: Prentice-Hall.
- Jain, A. K., Murty, M. N., and Flynn, P. J. (1999). Data clustering: A review. *ACM Comput. Surv.*, **31**(3): 264–323.
- Jakt, L. M., Cao, L., Cheah, K. S. E., and Smith, D. K. (2001). Assessing clusters and motifs from gene expression data. *Genome Res.*, **11**: 112–123.
- Jiang, D., Pei, J., and Zhang, A. (2003). DHC: A density-based hierarchical clustering method for time-series gene expression data. In *Proceedings of BIBE2003: 3rd IEEE International Symposium on Bioinformatics and Bioengineering*, March 10–12, 2003, Bethesda, MD.
- Jiang, D., Tang, C., and Zhang, A. (2004). Cluster analysis for gene expression data: A survey. *IEEE Trans. Knowledge Data Eng.*, **16**(11): 1370–1386.
- Kaufman, L., and Rousseeuw, P. J. (1990). *Finding Groups in Data: An Introduction to Cluster Analysis*. New York: Wiley.
- Kennedy, J., and Eberhart, R. C. (1999). Particle swarm optimization. In *Proceedings of IEEE International Conference on Neural Networks*, Vol. 4, Piscataway, NJ: pp. 1942–1948.
- Kohonen, T. (1984). *Self-Organization and Associative Memory*. Springer-Verlag, Berlin.
- Krishna, K., and Narasimha Murty, M. (1999). Genetic K-means algorithm. *IEEE Trans. Syst. Man Cybern.*, **29**(Part B): 433–439.
- Lazzeroni, L., and Owen, A. (2002). Plaid models for gene expression data. *Stat. Sinica*, **12**(61): 61–86.
- Levenshtein, V. I. (1966). Binary codes capable of correcting insertion and reversals. *Sov. Phys. Dokl.*, **10**: 707–710.
- Lipman, D. J., and Pearson, W. R. (1985). Rapid and sensitive protein similarity searches. *Science*, **227**: 1435–1441.
- Lu, Y., Lu, S., Deng, Y., and Brown, S. J. (2004a). Incremental genetic K-means algorithm and its application in gene expression data analysis. *BMC Bioinformatics*, **5**: 172–182.
- Lu, Y., et al. (2004b). FGKA: A fast genetic K-means clustering algorithm. In *Proceedings of the 2004 ACM Symposium on Applied Computing (SAC)*, Nicosia, Cyprus, March 2004, pp. 622–623.
- Madeira, S. C., and Oliveira, A. L. (2004). Biclustering algorithms for biological data analysis: A survey. *IEEE Trans. Computat. Biol. Bioinformatics*, **1**(1): 24–45.
- Mar, J.C., and McLachlan, G. J. (2003). Model-based clustering in gene expression microarrays: An application to breast cancer data. *Int. J. Software Eng. Knowledge Eng.*, **13**(6): 579–592.
- McLachlan, G. J., Bean, R. W. and Peel, D. (2002). A mixture model-based approach to the clustering of microarray expression data. *Bioinformatics*, **18**: 413–422.
- Mercer, D. P., and College, L. (2003). Clustering large dataset. Available: <http://www.stats.ox.ac.uk/~merc/~/merc/~/documents/Transfer.pdf>, accessed March 6, 2008.
- Mirkin, B. (2005). *Clustering for Data Mining: A Data Recovery Approach*. Boca Raton, FL: CRC Press.
- Mougeot, J. L., et al. (2006). Gene expression profiling of ovarian tissues for determination of molecular pathways reflective of tumorigenesis. *J. Mol. Biol.*, **358**: 310–329.
- Needleman, S., and Wunsch, C. (1970). A general method applicable to search for similarities in amino acid sequence of two proteins. *J. Mol. Biol.*, **48**: 443–453.
- Nielsen, T. O., et al. (2002). Molecular characterization of soft tissue tumors: A gene expression study. *Lancet*, **359**: 1301–1307.
- Pearson, W. R., and Lipman, D. J. (1988). Improved tools for biological sequence comparison. *PNAS*, **85**: 2444–2448.
- Quackenbush, J. (2001). Computational analysis of microarray data. *Nat. Rev. Genet.*, **2**: 418–427.
- Ramaswamy, S., Ross, K. N., Lander, E. S., and Golub, T. R. (2003). A molecular signature of metastasis in primary solid tumors. *Nat. Genet.*, **33**: 49–54.
- Ronning, C. M., et al. (2003). Comparative analysis of potato expressed sequence tag libraries. *Plant Physiol.*, **131**: 419–429.
- Rousseeuw, P. J. (1987). Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *J. Computat. Appl. Math.*, **20**: 53–65.

- Shai, R. et al. (2003). Gene expression profiling identifies molecular subtypes of gliomas. *Oncogene*, **22**: 4918–4923.
- Sheng, Q., et al. (2005). Advances in cluster analysis of microarray data. In *Data Analysis and Visualization in Genomics and Proteomics*, F. Azuaje and J. Dopazo, (Eds.). West Sussex, England: Wiley.
- Shindyalov, I. N., and Bourne, P. E. (2001). A database and tools for 3-D protein structure comparison and alignment using the Combinatorial Extension (CE) algorithm. *Nucleic Acids Res.*, **29**(1): 228–229.
- Smith, T. F., and Waterman, M. S. (1981). Identification of common molecular subsequences. *J. Mol. Biol.*, **147**: 195–197.
- Su, M., and Chang, H. (2001). A new model of self-organizing neural networks and its application in data projection. *IEEE Trans. Neural Network*, **12**: 153–158.
- Tamayo, P., et al. (1999). Interpreting patterns of gene expression with self-organizing maps: Methods and application to hemotopoietic differentiation. *PNAS*, **96**: 2907–2912.
- Tan, P., Steinbach, M., and Kumar, V. (2006). *Introduction to Data Mining*. New York: Addison Wesley.
- Tanay, A., Sharan, R., and Shamir, R. (2002). Discovering statistically significant biclusters in gene expression data. *Bioinformatics*, **18**: 136–144.
- Tanay, A., Sharan, R., and Shamir, R. (2006). Biclustering algorithms: A survey. In *Handbook of Computational Molecular Biology*, S. Aluru (Ed.). Boca Raton, FL: CRC Press, pp. 26-1–26-17.
- Tang, C., Zhang, L., and Zhang, A. (2002). An iterative strategy for pattern discovery in high-dimensional data sets. In *Proceedings of 11th International Conference on Information and Knowledge Management (CIKM02)*, ACM Press, McLean, VA, USA, November 4–9, 2002, pp. 10–17.
- Tang, C., Zhang, L., Zhang, A., and Ramanathan, M. (2001). Interrelated two-way clustering: An unsupervised approach for gene expression data analysis. In *Proceedings of BIBE2001: 2nd IEEE International Symposium on Bioinformatics and Bioengineering*, November 4–5, 2001, Bethesda, MD, pp. 41–48.
- Tavazoie, S., et al. (1999). Systematic determination of genetic network architecture. *Nat. Genet.*, **22**(3): 281–285.
- Toronen, P., Kolehmainen, M., Wong, G., and Castren, E. (1999). Analysis of gene expression data using self-organizing maps. *FEBS Lett.* **451**: 142–146.
- Tseng, G. (2004). A comparative review of gene clustering in expression profile. *Paper presented at the 8th International Conference on Control, Automation, Robotics and Vision (ICARCV)*, pp. 1320–1324.
- Wang, D., Ressom, H., Musavi, M., and Domnisoru, C. (2002). Double self-organizing maps to cluster gene expression data. In *ESANN's 2002 Proceedings—European Symposium on Artificial Neural Networks*, Bruges, Belgium, April 24–26, 2002, ISBN 2-9307-0201, pp. 45–50.
- Wang, J., Bo, T. H., Jonassen, I. and Hovig, E. (2003). Tumor classification and marker gene prediction by feature selection and fuzzy c-means clustering using microarray data. *BMC Bioinformatics*, **4**: 60.
- Ward, J. H. (1963). Hierarchical grouping to optimize an objective function. *J. Am. Stat. Assoc.*, **58**: 235–244.
- Welsh, P. L., et al. (2002). BRCA1 transcriptionally regulates genes involved in breast tumorigenesis. *PNAS*, **99**(11): 7560–7565.
- Wishart, D. S. (2005). Protein structure prediction and analysis. In *Bioinformatics: A Practical Guide to the Analysis of Genes and Proteins*, A. D. Baxeavanis, and B. F. Ouellette (Eds.). Hoboken, NJ: Wiley, pp. 223–252.
- Woolf, P. J., and Wang, Y. (2002). A fuzzy logic approach to analyzing gene expression data. *Genome Biol.*, **3**: 9–15.
- Xiao, X., et al. (2003). Gene clustering using self-organizing maps and particle swarm optimization. In *Proceedings of the 17th International Symposium on Parallel and Distributed Processing*, Nice, France, ISBN 0-7695-1926-1, pp. 154b.
- Xiong, J. (2006). *Essential Bioinformatics*. New York: Cambridge University Press.
- Xu, R., and Wunsch, D. (2005). Survey of clustering algorithms. *IEEE Trans. Neural Networks*, **16**(3): 645–678.
- Yeung, K. Y., et al. (2001a). Model-based clustering and data transformations for gene expression data. *Bioinformatics*, **17**: 977–987.

- Yeung, K. Y., Haynor, D. R., Raftery, A. E. and Ruzzo, W. L. (2001b). Validating clustering for gene expression data. *Bioinformatics*, **17**(4): 309–318.
- Zhang, T., Ramakrishnan, R., and Livny, M. (1996). BIRCH: An efficient data clustering method for very large database. Paper presented at the ACM SIGMOD Conference, pp. 103–114.
- Zhao, Y., and Karypis, G. (2005). Data clustering in life sciences. *Mol. Biotechnol.*, **31**(1): 55–80.

**CLASSIFICATION: SUPERVISED
LEARNING WITH
HIGH-DIMENSIONAL
BIOLOGICAL DATA***Hongshik Ahn**Department of Applied Mathematics and Statistics, SUNY at Stony Brook,
Stony Brook, New York, USA**Hojin Moon**Department of Mathematics and Statistics, California State University,
Long Beach, California, USA***6.1 INTRODUCTION**

In this chapter we introduce and compare various algorithms which have been widely used for classification and feature selection. Class prediction is a supervised learning method where the algorithm learns from a training set and establishes a prediction rule to classify new samples. This method can be used, for instance, to classify cancer types using gene expression profiling and to predict, based on protein expression profiles, which breast cancer patients will relapse within two years of diagnosis versus which will remain disease free. It can also be applied to predict, based on either genomics or proteomics data, which patients are likely to experience severe toxicity from a new drug versus which will tolerate it well. Golub et al. (1999) classified acute myeloid leukemia (AML) and acute lymphoblastic leukemia (ALL) subtypes using a variant of linear discriminant analysis based on gene expression profiling. On the other hand, Petricoin et al. (2002a, b) used support vector machines for early detection of ovarian and prostate cancers based on proteomics profiles.

There are a number of classification methods for analyzing data, including artificial neural (ANNs; see Beale and Jackson, 1990) networks, k -nearest-neighbor (k -NN) methods, decision trees, support vector machines (SVMs), and Fisher's linear discriminant analysis (LDA). Among these methods, a decision tree is a flow-chart-like tree structure. An intermediate node denotes a test on a predictive attribute, and a branch represents an outcome of the test. A terminal node denotes class distribution.

Tree-structured approaches are developed for classification (Breiman et al., 1984; Loh and Vanichsetakul, 1988). The first tree-structured approach was the automatic interaction detection (AID) program introduced by Morgan and Sonquist (1963). In the AID program, recursive partitioning was used as an alternative to the least-squares regression for model fitting. Breiman et al. (1984) developed the classification and regression trees (CART) method of selecting a tree of appropriate size for classification and piecewise constant regression. Loh and Vanichsetakul (1988) proposed a fast algorithm for classification trees (FACT) by recursive application of linear discriminant analysis. Their splitting rule is based on a separation of multivariate normal distributions. They used the F -ratio to determine when to split and when to stop splitting. Among the classification tree algorithms, CART and QUEST (Loh and Shih, 1997) yield binary trees and CRUISE (Kim and Loh, 2001) yields multiway splits. In contrast to the exhaustive search method of CART, QUEST and CRUISE use a discriminant-based procedure for splitting. QUEST and CRUISE use linear combination splits in order to achieve gains in classification accuracy over univariate splits. Construction of a decision tree consists of two phases: growing a huge tree and pruning. A decision tree is constructed by splitting nodes according to values of predictive attributes selected on the basis of maximum reduction in node impurity (Breiman et al., 1984).

As an alternative to the classification methods listed above, logistic regression can be used as a parametric approach. A logistic regression model cannot be used if the number of predictors exceeds the number of observations. In this case, a variable selection needs to be conducted in order to utilize logistic regression.

It is a well-understood phenomenon that a prediction model built from a high-dimensional dataset consisting of thousands of available predictor variables and a relatively small sample size can be quite unstable (Miller, 2002). Models that are developed using an intense selection process are highly prone to change with a new training sample. Furthermore, there is a multiplicity of good models when $n \ll m$, as observed by examination of model fit, where n is the sample size and m is the number of predictor variables.

In the gene expression literature, much has been written on the instability of gene expression signatures. Numerous studies have illustrated the variability of the components of the signature and have cautioned that the final models are simply representative of many other potential profiles (Sparano et al., 2005; Michiels et al., 2005). Moreover, the model selected may generalize poorly to a new dataset. Simple model averaging (Burnham and Anderson, 2002) has been utilized to address this issue by using a weighted average of many competing models, where the weight is based on an information-theoretic statistic such as AIC (Akaike, 1974). This allows the final prediction to be a function of many different models, a result that is inherently more stable than a single model.

Any method that incorporates information from training samples in the design of a classifier employs learning. Creating classifiers involves some general form of model, or form of the classifier, and using training patterns to learn or estimate the unknown parameters of the model. Learning refers to some form of algorithm for reducing the error on a set of training data. Learning comes in several general forms.

Development of a class prediction algorithm generally consists of three steps: first, selection of predictors; second, fitting the prediction model to develop the

classification rule; and third, performance assessment. The first two steps build a prediction model, and the third step assesses the performance of the model. Some classification algorithms, such as the classification tree or stepwise logistic regression, perform the first two steps simultaneously.

With genomic and proteomic data, the set of predictors is often large while the sample size is relatively small. In such a case, a model may capture the patterns in the training data extremely well. Consequently, it may lead to a problem of overfitting the training data. Overly fitted trees have too many branches, and some may reflect anomalies due to noise or outliers. Overfitted trees have poor accuracy for unseen samples. To avoid overfitting in classification, pruning the tree is required. Hawkins (2004) illustrates the problem of overfitting that violates model parsimony.

Ensemble methodology is a natural next step to simple model averaging for class prediction. Ensemble of classifiers are gaining acceptance in the data-mining community due to the significant improvement in accuracy (Breiman, 1996; Freund and Schapire, 1996; Bauer and Kohavi, 1999). The motivation for ensembles is to combine the outputs of many weaker classifiers to produce a strong classifier (Hastie et al., 2001). Significant improvement in classification accuracy can be resulted from growing an ensemble of trees and classifying by majority voting from each classifier. In Section 6.5, we demonstrate mathematically why an ensemble of independently voting classifiers can be expected to enhance class prediction, which gives insight into how ensembles should be constructed.

Recently three ensemble voting approaches, boosting, bagging (Breiman, 1996), and random subspace (Ho, 1998), have received attention. These methods are generated by resampling training sets from the original dataset. The learning algorithm builds up a base classifier for each training set. Boosting was developed as a method for improving the performance of any weak learning algorithm, which requires only being slightly better than a random guess. Boosting changes adaptively the distribution of the training set based on the performance of previously created classifiers. For combining the weak classifiers, it takes a weighted majority vote of their predictions. The bagging algorithm uses bootstrap samples to build the base classifiers. Each bootstrap sample is formed by randomly sampling, with replacement, the same number of observations as the training set. The final classification produced by the ensemble of these base classifiers is obtained using equal-weight voting. The random subspace method combines multiple classification trees constructed in randomly selected subspaces. The final classification is obtained by an equal-weight voting of the base trees.

Breiman (2001) developed the random forest (RF) method by combining tree predictors such that each tree depends on the values of a random vector sampled independently. RF is based on bagging. Moon et al. (2006) and Ahn et al. (2007) introduced the CERP (classification by ensembles from random partitions) methodology. CERP generates multiple ensembles by randomly repartitioning the feature space and building base classifiers in order to achieve a further improvement. Majority voting is performed among these ensembles. CERP uses classification trees (C-T CERP) or logistic regression trees (LR-T CERP) as base classifiers. There are major differences among boosting, bagging, random subspace, and CERP. The same features are used by each classifier in the other methods, while different features are used by each classifier in CERP. Boosting uses the same training samples used by each classifier, with

hard-to-classify samples getting more weight by design, while bagging yields incomplete overlap of training samples among classifiers, with some samples getting more weight randomly. Boosting, bagging, and random subspace cause dependence among classifiers, while the classifiers are supposed to be less correlated in an ensemble of CERP. Examples illustrating how to use the software for the classification methods discussed above are given in Section 6.7.

6.2 CLASSIFICATION AND PREDICTION METHODS

In this section widely used classification methods are discussed. For description of each method, we use the following notation. Let the total sample size of each dataset be n , with m total predictors under consideration. The predictors are represented by an $n \times m$ matrix $\mathbf{X} = (x_{ij})$, where x_{ij} is the j th measurement for an independent observation i . The class label for the observation i is y_i , where $\mathbf{y} = (y_1, y_2, \dots, y_n)'$ and y_i is defined to be 0 or 1 depending on class status.

6.2.1 Naive Bayes Classifier

Bayesian decision theory is a fundamental statistical approach to the problem of classification. This approach is based on quantifying the trade-offs between various classification decisions using probability and the costs that accompany such decisions. It makes the assumption that the decision problem is posed in probabilistic terms and that all of the relevant probability values are known.

A naive Bayes classifier is a simple probabilistic classifier based on the so-called Bayes theorem with strong independence assumptions and is particularly suited when the dimensionality of the inputs is high. The naive Bayes model assumes that, given a class $\Gamma = j$, the features X_i are independent. Despite its simplicity, the naive Bayes classifier is known to be a robust method even if the independence assumption does not hold (Michalski and Kaufman, 2001).

The probabilistic model for a naive Bayes classifier is a conditional model $P(\Gamma | X_1, X_2, \dots, X_m)$ over a dependent class variable Γ , conditional on features X_1, X_2, \dots, X_m . Using Bayes's theorem, $P(\Gamma | X_1, \dots, X_m) \propto P(\Gamma)p(X_1, \dots, X_m | \Gamma)$. The prior probability $P(\Gamma = j)$ can be calculated based on the ratio of the class j samples such as $P(\Gamma = j) = (\text{number of class } j \text{ samples})/(\text{total number of samples})$. Having formulated the prior probabilities, the likelihood function $p(X_1, X_2, \dots, X_m | \Gamma)$ can be written as $\prod_{i=1}^m p(X_i | \Gamma)$ under the naive conditional independence assumptions of the feature X_i with the feature X_j for $j \neq i$. A new sample is classified to a class with maximum posterior probability, which is $\arg \max_{\Gamma_j \in \Gamma} P(\Gamma_j) \prod_{\Gamma} P(X_i | \Gamma_j)$. If the independence assumption is correct, it is the Bayes optimal classifier for a problem. Extensions of the naive Bayes classifier can be found in Demichelis et al. (2006).

6.2.2 Logistic Regression

Logistic regression is a model used for prediction of the probability of occurrence of an event. It makes use of several predictor variables that may be either numerical or

categorical. Logistic regression analyzes binomially distributed data of the form

$$Y_i \sim \text{Bin}(n_i, p_i) \quad i = 1, 2, \dots, m,$$

where the numbers of Bernoulli trials n_i are known and the probabilities of success p_i are unknown. For each trial there is a set of explanatory variables that might inform the final probability. The functional form of the logistic regression is

$$P(y = 1 | \mathbf{x}) = \frac{\exp\left(\sum_j \beta_j x_j\right)}{1 + \exp\left(\sum_j \beta_j x_j\right)}$$

where $y = 1$ if the sample is positive, $y = 0$ if the sample is negative, and the x_j are predictor variables. The logits of the unknown binomial probabilities are modeled as a linear function of the \mathbf{x} ,

$$\text{logit}(p_i) = \ln\left(\frac{p_i}{1 - p_i}\right) = \mathbf{x}\boldsymbol{\beta}$$

The logistic regression model can be used in a classification problem by assigning the class into one of the two classes according to the predicted value. The default decision threshold uses 0.5 to predict class membership, but it can be adjusted to reflect prior probabilities, for example, $n_1/(n_0 + n_1)$, where n_i is a sample size in class i . The decision rule assigns a subject to class 1 if $\hat{y} \geq 0.5$ and to class 0 if $\hat{y} < 0.5$, where \hat{y} is the fitted value.

6.2.3 Linear Discriminant Analysis

The LDA is used to separate samples of distinct groups by maximizing their between-class separability while minimizing their within-class variability. There are various forms of LDA. In contrast to the more common Fisher's LDA (FLDA) technique, diagonal linear discriminant analysis (DLDA) is a simple classification rule based on a linear discriminant function. The class densities are therefore assumed to have the same diagonal covariance matrix, estimated by the pooled covariance matrix. Each new observation is classified based on the distance of each group mean vector using the Mahalanobis distance. The Mahalanobis distance with mean vector $\boldsymbol{\mu}$ and covariance matrix \mathbf{S} for a set of features \mathbf{X} is defined as $\sqrt{(\mathbf{X} - \boldsymbol{\mu})' \mathbf{V}^{-1} (\mathbf{X} - \boldsymbol{\mu})}$. DLDA ignores correlation between predictors and is a simple weighted sum across features. It has been shown by Dudoit et al. (2002) to be a consistently good classifier, even when compared with RF and other complex classifiers. Despite its simplicity, DLDA has been shown in Dudoit et al. to give better generalization performance than other forms of LDA, such as DQDA (diagonal quadratic discriminant analysis: nonlinear discriminant function) and FLDA (which requires estimation of the full covariance matrix). It is often beneficial in DLDA to initially reduce the dimension of the feature set by a variable selection. In our comparison, the R package (stat.diag.da) is used, and the BW ratio (see Section 6.3.1) is used in the variable selection.

LDA assumes normal distribution of the explanatory variables, while logistic regression does not. If the Gaussian assumptions are met, then LDA is a more powerful and efficient model than logistic regression.

6.2.4 *k*-Nearest-Neighbor Classifiers

The *k*-NN procedure classifies a data point by considering the closest *k* neighbors to it based on the distance between \mathbf{x}_i and \mathbf{x}_j , where $\mathbf{x}_i = (x_{i1}, x_{i2}, \dots, x_{im})'$ and $\mathbf{x}_j = (x_{j1}, x_{j2}, \dots, x_{jm})'$. In our comparison, we use Euclidean distance on standardized data. The *k*-NN rule classifies a new observation \mathbf{x}^* based on the *k*-NN in the training set. The class designation for a new observation is based on the majority vote of the *k* neighbors. As *k* increases, the variance of the classifiers will decrease, but the bias will be high. The *k*-NN classifier has been shown (Friedman, 1997; Dudoit et al., 2002) to be a highly consistent classifier. The *k*-NN procedure in R is used in our comparison. A drawback of *k*-NN is that a variable preselection is required when $n < m$. Following the method of Dudoit et al. (2002), we used the ratio of between-group to within-group sums of squares (BW ratio: see Section 6.3.1) for each feature and retain those with the highest ratio. The number of features that should be retained is data specific; therefore, we present the best result obtained after a reasonable search across *n*.

6.2.5 Shrunken Centroid

The shrunken centroid (SC) algorithm was developed by Tibshirani et al. (2002). It is similar to nearest centroid classification, but one important modification to standard nearest centroid classification is to shrink each class centroid toward the overall centroid for all classes by threshold. The shrinkage consists of moving the centroid toward zero by threshold.

Let *n* be the total sample size and x_{ij} be the expression for genes $i = 1, 2, \dots, p$ and samples $j = 1, 2, \dots, n$. Assume that there are *K* classes. We let I_k be indices of the n_k samples in class *k*. Then, the centroid for gene *i* and class *k* is $\bar{x}_{ik} = \sum_{j \in I_k} x_{ij} / n_k$ and the overall centroid for gene *i* is $\bar{x}_i = \sum_{j=1}^n x_{ij} / n$. Let

$$d_{ik} = \frac{\bar{x}_{ik} - \bar{x}_i}{m_k(s_i + s_0)}$$

where

$$s_i = \sqrt{\left(\sum_k \sum_{j \in I_k} (x_{ij} - \bar{x}_{ik})^2 \right) / (n - K)}$$

is the pooled within-class standard deviation for gene *i*, $m_k = \sqrt{1/n_k + 1/n}$ and s_0 equals the median value of the s_i over the set of genes. The SC shrinks each d_{ik} toward zero such that $\bar{x}'_{ik} = \bar{x}_i + m_k(s_i + s_0)d'_{ik}$ using soft thresholding defined by $d'_{ik} = \text{sign}(d_{ik})(|d_{ik}| - \Delta)_+$, where $t_+ = t$ if $t > 0$ and $t_+ = 0$ otherwise. Typically, Δ is chosen to be the value yielding the minimum generalized misclassification

error rate by K -fold cross-validation. Whichever class centroid the new sample is closest to becomes its predicted class.

Advantages of the SC algorithm would be the following: It is further improved from the standard nearest centroid classification by reducing the effect of noisy genes by thresholding, and thereby it does automatic gene selection.

6.2.6 Artificial Neural Networks

Artificial neural networks are machine-based learning models and were first developed by Bernard Widrow of Stanford University in the 1950s. They have the ability to learn and process information in a wider context than simple nonalgorithmic or rule-based systems, thus allowing them to successfully simulate human reasoning (Bishop, 1995).

A neural network is a system for processing of information, the structure and function of which are motivated by analogy with a biological nervous system. A neural network consists of a set of neurons, and each neuron is connected to several other neurons with an associated weight (Fausett, 1994). A neural network is a data-driven and nonparametric model. Figure 6.1 shows a simple network scheme of ANNs.

Neural network learning procedures include both unsupervised learning and supervised learning. The learning is embodied in modifications of synaptic weights, which can be determined by learning algorithms. In a correlational learning rule, weights are adjusted according to Hebb's rule: $\Delta(W_{ij}) = O_i O_j$, where $\Delta(W_{ij})$ is a change in weights in units connecting between units i and j and O_i is an output of unit i . This rule is used in many models. In a competitive learning rule, output neurons

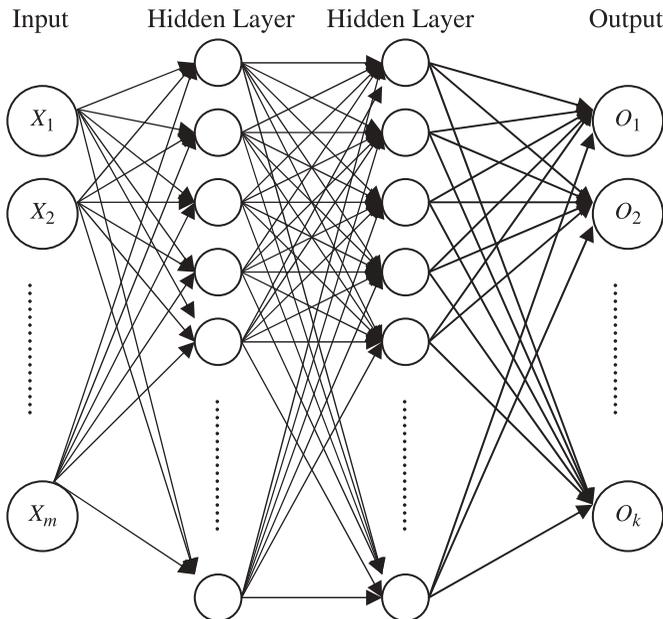


Figure 6.1 Schematic diagram for ANN.

compete until one dominates and weights are adjusted by $\Delta(W_{ij}) = O_i(O_j - W_{ij})$, and it is used in adaptive resonance theory models. In a learning rule via error correction, weights are adjusted by minimizing output errors with respect to weights: $\Delta(W_{ij}) = E(O_i O_j)$, and it is used in Perceptron, MADALINE, and back-propagation models. In a stochastic learning rule, weights are adjusted by seeking a globally optimal solution in order to discover relationships between arbitrary sets of patterns. It is used in Boltzmann and Cauchy machines.

The ANNs are capable of solving simultaneous systems of nonlinear equations and generally perform well. However, they are complex and computationally expensive.

The Statistica (version 7.1, Statsoft, Inc., Tulsa, OK) Artificial Neural Network Package [*STATISTICA* Automated Neural Networks (SNN)] is used in this comparison. Multilayer perceptrons with back propagation were performed. The error function used was the cross-entropy function. A linear synaptic function was used and a combination of the following four activation functions was used: linear, hyperbolic, softmax, and logistic. The number of epochs to train the model was set to 100 although the network always converged in less number of epochs. The learning rate was set to 0.01 and there was one hidden layer in the network.

6.2.7 Classification Trees

Recently tree-based methods have been developed by many researchers. Among the classification tree algorithms, CART and QUEST (Loh and Shih, 1997) yield binary trees and CRUISE (Kim and Loh, 2001) yields multiway splits. The optimal trees obtained from these algorithms are included in the comparison. In contrast to the exhaustive search method of CART, QUEST and CRUISE use a discriminant-based procedure for splitting. QUEST and CRUISE use linear combination splits in order to achieve gains in classification accuracy over univariate splits. The binary executables for QUEST and CRUISE are obtained from <http://www.stat.wisc.edu/~loh/quest.html> and <http://www.stat.wisc.edu/~loh/cruise.html>, respectively. Shell programs are developed to conduct 20 cross-validations for these executables. The linear combination split option is available in both QUEST and CRUISE. We used the defaults for the other options.

In CART, a node in a tree splits recursively with the goal of making data more homogeneous according to a splitting criterion until the tree is fully grown. A measure of node impurity given node t can be defined by the Gini diversity index (Breiman

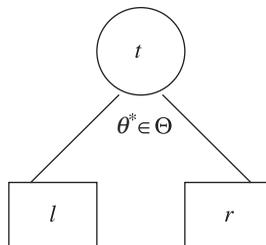


Figure 6.2 A branch in a tree T .

et al., 1984), $i(t) = 1 - \sum_k p^2(k | t)$, where $p(k | t)$ is a probability that a sample is in class k given that it falls into a node t . We consider a branch in a tree T (see Fig. 6.2). We let l and r be the left and right subnodes, respectively, of a parent node t , in which a split $\theta^* \in \Theta$ maximizes a goodness-of-split function. A goodness-of-split criterion in this chapter will be chosen such that a split at node t maximizes the reduction of the impurity, $\Delta i(\theta, t) = i(t) - [p_l i(l) + p_r i(r)]$, where a split θ of node t sends a proportion p_l of data cases in node t to l and a proportion p_r to r .

We define $N(t)$ as the total number of cases in node t . For growing a large initial tree, the nodes will continue splitting until a terminal node is either pure (i.e., the cases in the node are all in one class) or $N(t) \leq 5$ as in CART.

Two approaches are considered to avoid overfitting data. One is the cross-validation approach with minimal cost-complexity pruning method of CART (Breiman et al., 1984). In CART, after the initial large tree T_{\max} is constructed, a nested sequence of subtrees is obtained by progressively deleting branches according to the pruning method. The other method to avoid overfitting data is the method developed by Ahn and Loh (1994), which uses a cross-validators multistep look-ahead stopping rule to determine whether the node should be split or not and bootstrap resampling to determine the proper depth of a tree. The splitting point of a node is based on an analysis of the pattern of residuals.

6.2.8 Random Forest

The bagging algorithm uses bootstrap samples to build base classifiers. Each bootstrap sample is formed by randomly sampling, with replacement, the same number of observations as the training set. The final classification produced by the ensemble of these base classifiers is obtained using equal-weight voting.

RF was developed by Breiman (2001) and is available as a package (RandomForest) in R. It is an ensemble-based approach building on single CART overfitted trees that has shown to be a consistently good classifier, comparable to results obtained by SVM and boosting methods. RF is based on bagging, which is different from boosting. Boosting uses the same training samples used by each classifier, with hard-to-classify samples getting more weight by design, while bagging yields incomplete overlap of training samples among classifiers, with some samples getting more weight randomly.

RF creates diversity in two ways: It perturbs the training set used by taking bootstrap samples for each tree generated (bagging) and it selects a random subspace of the predictors at each node. Since RF changes the variable space at each node, the resulting tree is not optimal with respect to any fixed subspace of the data. The number of trees generated may vary using the `ntree` option in R, but it has been shown to work well at the default of `ntree = 500`. More precisely, let $D(\mathbf{x}, \mathbf{y} | b)$ be the classifier built with bootstrap sample b , $b = 1, 2, \dots, 500$. The predicted class of observation i ($i = 1, 2, \dots, n$) is based on the majority vote for class c : $\max_c \left\{ \sum_{b=1}^{500} I[D(\mathbf{x}, \mathbf{y} | b) = c] \right\}$.

The number of features selected randomly at each node may also be varied; however, the default value of \sqrt{m} (or $\lfloor \sqrt{m} \rfloor$ for a noninteger value) seems to have consistently good results across many examples. The RF program in the R package with the default number of trees is used in our comparison.

6.2.9 Support Vector Machines

SVM was introduced by Vapnik (1995). This method finds optimal linear hyperplanes in input space and kernel space in the sense of avoiding overfitting or may be expanded to allow the boundary to be found in a higher dimensional space by projecting the input space into a large, sometimes infinitely large space. Explicit characterization of this expanded space is not necessary since we may relate to this space by using only the input vectors and the so-called kernel trick that computes inner products in this enlarged space.

Let the training data consist of n pairs $(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_n, y_n)$, with $\mathbf{x}_i \in \mathcal{R}^p$ and $y_i \in \{-1, 1\}$. If the data are linearly separable, the SVM classifier finds the closest points in convex hulls and finds a hyperplane (P_0) bisecting the closest points, where P_0 is defined by $\{\mathbf{x}: f(\mathbf{x}) = \mathbf{x}'\boldsymbol{\beta} + \beta_0 = 0\}$, and $\|\boldsymbol{\beta}\| = 1$. Then, the classifier creates a parallel hyperplane (P_1), $\{\mathbf{x}: f(\mathbf{x}) = \mathbf{x}'\boldsymbol{\beta} + \beta_0 = -1\}$, on a point in class -1 closest to P_0 and a second parallel hyperplane (P_2), $\{\mathbf{x}: f(\mathbf{x}) = \mathbf{x}'\boldsymbol{\beta} + \beta_0 = 1\}$, on a point in class 1 closest to P_0 . The optimal hyperplane that separates the data can be found by maximizing the margin (M) that is a perpendicular distance between two parallel supporting planes P_1 and P_2 . A resulting classifier would be $\hat{y} = \text{sgn}(\mathbf{x}'\boldsymbol{\beta} + \beta_0)$. Since the classes are separable, we know that $M = 2/\|\boldsymbol{\beta}\|$. Since maximizing the margin M is the same as minimizing $\|\boldsymbol{\beta}\|/2$, this problem can be reformulated as finding $\boldsymbol{\beta}$ and β_0 such that $\Phi(\boldsymbol{\beta}) = \|\boldsymbol{\beta}\|/2$ is minimized subject to $y_i(\mathbf{x}'_i \boldsymbol{\beta} + \beta_0) \geq 1$ for all $\{(\mathbf{x}_i, y_i), i = 1, 2, \dots, n\}$.

When the training set is noisy or the classes overlap in feature space, slack variables ξ_i , a distance from a point on the wrong side of the margin to a hyperplane, can be added to allow misclassification of difficult or noisy samples. Then, the constrained optimization problem becomes to find $\boldsymbol{\beta}$ and β_0 such that $\Phi(\boldsymbol{\beta}) = \|\boldsymbol{\beta}\|/2 + C \sum \xi_k$ is minimized subject to $y_i(\mathbf{x}'_i \boldsymbol{\beta} + \beta_0) \geq 1 - \xi_i$ for all $\{(\mathbf{x}_i, y_i), i = 1, 2, \dots, n\}$ and $\xi_i \geq 0$ for all i . The parameter C can be viewed as a control parameter to avoid overfitting.

For datasets that are not linearly separable, support vector machines map the data into higher dimensional space where the training set is separable via some transformation $K: \mathbf{x} \rightarrow \phi(\mathbf{x})$. A kernel function $K(\mathbf{x}_i, \mathbf{x}_j) = \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle$ computes inner products in some expanded feature space. Some kernel functions such as linear $K(\mathbf{x}_i, \mathbf{x}_j) = \langle \mathbf{x}_i, \mathbf{x}_j \rangle$ and Gaussian (radial-basis function) $K(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2 / 2\sigma^2)$ are widely used.

The SVM program in R using the e1071 package is used in the comparison. Some care needs to be taken with respect to the choice of kernel (we examined linear and radial basis using default parameters) as well as the parameters for these transformations.

6.2.10 Boosting Methods

Boosting, a metaclassifier, changes adaptively the distribution of the training set based on the performance of previously created classifiers. As described in the introduction, boosting combines weak classifiers and takes a weighted majority vote of their predictions.

AdaBoost (Freund and Schapire, 1997) is an algorithm that creates many base classifiers (decision “stumps”: trees with a single split) from the same dataset and combines the trees using majority voting. The base trees are formed using weighted versions of the same dataset, where the weights are based on the previous classifier’s fit. For difficult-to-classify observations, the weight is increased in the subsequent classifiers. The decision stumps are highly biased and lower variance classifiers, and boosting has been shown to work as a bias reduction tool. Schapire et al. (1998) showed that boosting is “especially effective” on increasing the margins of the training examples, where the margin for a two-class system is defined as the difference between the weight assigned to the correct label and the weight assigned to an incorrect label. LogitBoost is similar to AdaBoost; however, it uses the logit loss function and fits an additive logistic regression model (Friedman et al., 2000). We examined variable preselection and found little impact on the accuracy.

The R boosting package contains four different approaches: AdaBoost, LogitBoost, L2Boost, and BagBoost. L2Boost is an implementation of the LogitBoost algorithm for binary classification (Dettling and Bühlmann, 2003). Since BagBoost conducts repeated baggings in addition to boosting, it takes much more computing time than the other three approaches. The performance of L2Boost and BagBoost are somewhat similar (see Ahn et al., 2007). These R boosting programs use the classification tree (rpart) in R with a single split as the base classifier. We tried 30–100 iterations of weighted voting. Typically 20 iterations of weighted voting are sufficient to converge to the final prediction.

6.2.11 Classification by Ensembles from Random Partitions

CERP (Moon et al., 2006; Ahn et al., 2007) is an ensemble-based approach for classification. This approach is designed specifically for high-dimensional datasets for which a classifier is sought. Variable preselection is not required. CERP is able to bypass the constraint of a large number of predictor variables m and a small sample size n by randomly partitioning the variables into k mutually exclusive subspaces. The optimal number of subspaces is sought by an adaptive binary search algorithm (Ahn et al., 2007). A benefit in partitioning the input space is that each subspace may be treated separately until aggregation. This leads to a computational advantage that will be important as the dimension of datasets grows beyond the size that can be easily handled in a whole.

CERP uses two major approaches for generating base classifiers: C-T CERP creates optimal classification trees (Breiman et al., 1984) and LR-T CERP creates logistic regression trees (Ahn and Chen, 1997) within each subspace using only the m/k -dimensional space of the variables in the partition. CERP combines the results of these multiple trees to achieve an improved accuracy of class prediction by a majority voting or by taking the average of the predicted values within an ensemble. A major advantage of LR-T CERP is that the logistic regression is available without losing the ensemble accuracy for data with $n \ll m$ by a random partition without a variable selection. A variable selection needs to be conducted in order to utilize logistic regression if the number of predictors exceeds the sample size. In LR-T CERP, however, variable selection is not required because each tree will be

constructed from a small subspace of the predictor variables. Therefore, CERP can be easily implemented for applications to high-dimensional data.

Multiple ensembles are generated by randomly repartitioning the feature space and building optimal trees. While CERP captures most of the features contained in the data, only a few selected variables are used by each tree classifier in C-T CERP or LR-T CERP. When we have multiple ensembles, fresh new information can be obtained by a different partition of the variables in each additional ensemble. The multiple ensembles contribute to a further gain of the overall accuracy.

6.3 FEATURE SELECTION AND RANKING

While DNA microarray technology provides tools to simultaneously study the expression profiles of thousands of distinct genes in a single experiment, one major challenge in the classification of patients is the large number of genes over a relatively small number of patients in the dataset. Since many of those genes are not relevant, feature selection is a commonly addressed problem in classification (Blum and Langley, 1997; Kohavi and John, 1997). The goal of gene selection is to identify a set of genes which plays an important role in the classification. A common approach is to select a fixed number of the highest ranked genes based on statistics similar to the t -test or some discrimination measures (Dudoit et al., 2002; Liu et al., 2002), SVM (Guyon et al., 2002; Tsai et al., 2004), or RF (Breiman, 2001; Díaz-Uriarte and de Andrés, 2006).

6.3.1 Various Algorithms for Variable Importance Ranking

A crude criterion for identifying the genes that discriminate between all the classes is the t -test. The t -statistic selects features under the following ranking criterion:

$$t_i = \frac{[|\bar{x}_i^+ - \bar{x}_i^-|]}{\sqrt{\left[\frac{(s_i^+)^2}{n^+}\right] + \left[\frac{(s_i^-)^2}{n^-}\right]}}$$

where \bar{x}_i^+ and \bar{x}_i^- are the mean values of feature i over positive and negative samples, respectively; s_i^+ and s_i^- are the corresponding sample standard deviations; and n^+ and n^- are the corresponding sample sizes.

A feature selection method that is used in a tree-structured classification algorithm is the mean decrease in node impurity (MDI; Breiman et al., 1984). In the MDI method a variable that has much reduction of node impurity is ranked as more important. The measure of node impurity is based on the Gini diversity index (Breiman et al., 1984) and it can be used in the tree-structured algorithms. Figure 6.3 illustrates an example of the MDI variable selection method. The reduction of node impurity Δ is measured in each node. Evaluating the sum of Δ 's of each corresponding X_i in the tree T , the MDIs in nodes X_1 , X_2 , X_3 , and X_4 would be 0.4, 0.34, 0.05, and 0.01, respectively, in Figure 6.3. In the figure, terminal nodes are represented by squares and intermediate nodes are represented by circles.

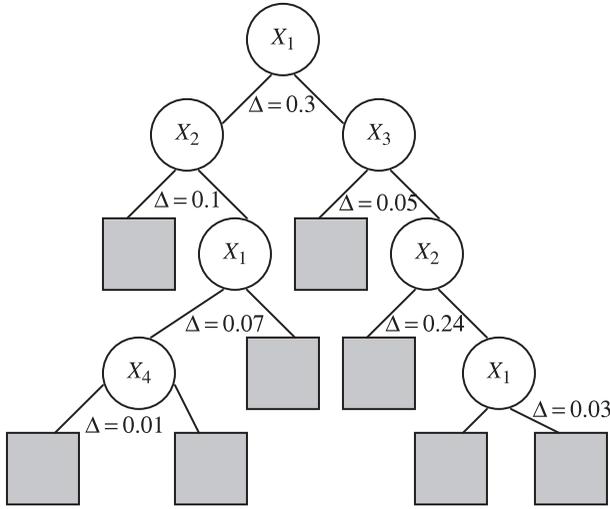


Figure 6.3 Example of mean decrease in node impurity.

The measure of mean decrease in accuracy (MDA) is used to rank the importance of features for RF (Breiman, 2001). This method determines the variable importance by measuring mean decrease in out-of-bag accuracy for each tree. Bootstrapping is a resampling method with replacement from the training data, and about one-third of the training data is left out as training samples (Breiman, 2001). Out-of-bag accuracy is calculated by using the left-out samples from each bootstrap sample as a test set (Diaz-Uriarte and de Andrés, 2006). The mean decrease is measured as the difference between the prediction accuracies from the original samples and the one with the out-of-bag portion of the data by randomly permuting the values of a predictor variable in the out-of-bag set of each tree. This method is used with the randomForest package implemented in R in this chapter.

Guyon et al. (2002) introduced the support vector machines recursive feature elimination (SVM-RFE) method for selecting important genes in classification. The SVM-RFE method is known as the wrapper approach that feature selection takes into account the contribution to the performance of SVM. After training SVM, the predictor variables are ranked by the absolute value of variable weights. Starting with all the features, at each RFE step a certain percentage of genes with the smallest ranking score (the least important feature) according to a threshold is subsequently removed from the variables of SVM with a linear kernel function in each iteration. At each step, the associated magnitudes of the weight vector of the SVM, related to the support of features for the discrimination function, are used as the feature-ranking criterion. The SVM-RFE procedure continues with the remaining variables recursively.

Dudoit et al. (2002) introduced the ratio of between-group to within-group sums of squares (BW ratio). The BW ratio for a gene j , $BW(j)$, is defined as

$$BW(j) = \frac{\sum_i \sum_k I(y_i = k)(\bar{x}_{kj} - \bar{x}_j)^2}{\sum_i \sum_k I(y_i = k)(x_{ij} - \bar{x}_{kj})^2} \quad \text{where } \bar{x}_j = \frac{\sum_i x_{ij}}{N}; \bar{x}_{kj} = \frac{\sum_{I(y_i=k)} x_{ij}}{N_k}$$

and i indicates a sample. Here, $I(\cdot)$ is the indicator function, $\bar{x}_{\cdot j}$ denotes the average value of a gene j across all the training samples, and \bar{x}_{kj} denotes the average value of a gene j for a class k . This criterion has been shown to be reliable for variable selection from high-dimensional datasets (Dudoit et al., 2002). The variable importance can be obtained according to the BW ratio of each variable obtained in the corresponding learning set in the CV. The genes with largest BW ratios are ranked high as significant genes.

Recently, Baek et al. (2008) introduced a weighted frequency measure (WFM) for ranking variable importance in CERP. Each split variable used in trees in an ensemble is measured by a weight based on the sample proportions. We suppose there are n training samples in the root node. We suppose the root node is split into n_l samples in the left-child node and n_r samples in the right-child node. Then, the rates n_l/n and n_r/n are weights for ranking variables in the left- and right-child nodes, respectively, where $n_l + n_r = n$. As the tree splits further, the weights for the second left- and right-child nodes split from the first left-child node would be n_{ll}/n and n_{lr}/n , respectively, where $n_{ll} + n_{lr} = n_l$. Similarly, the weights for the second left- and right-child nodes split from the first right-child node would be n_{rl}/n and n_{rr}/n , respectively, where $n_{rl} + n_{rr} = n_r$. This weight is multiplied by the frequency measure on each variable in trees in an ensemble to compute the mean weighted frequency in the cross-validation. Each predictor variable is ranked with this weighted frequency measure.

These methods of feature selection and ranking can be categorized into three approaches: *filter*, *wrapper*, and *embedded*. The filter approaches to feature selection do not acquire a learning algorithm in the feature selection process. In other words, the feature selection criteria do not depend on a classifier to discern between good and poor features. Hence, gene selection can be done with certain stand-alone methods such as the t -test and the BW ratio. In the wrapper approach, the feature selection depends on a classifier to determine if a feature is good or poor. The wrapper approaches are known as recursive procedures. In contrast to the filter approaches, the contribution of genes in the classifier is accounted for gene selection according to classification. The advantage of the wrapper methods is twofold: (1) the feature selection criterion is consistent with the classification criterion and (2) the gene correlation structure is considered in the feature selection. The embedded approaches to feature selection search the feature selection space and the parameter space of a

TABLE 6.1 Algorithms for Feature Selection and Ranking by Category

Algorithms	Approaches of feature selection	Software
T-test	Filter	R ^a : t.test ^b , stats ^c
BW ratio	Filter	R ^a : stat.diag.da ^b , sma ^c
SVM-RFE	Wrapper (sequential)	MATLAB ^a : Spider package
MDA	Embedded (Tree-based; Bagging)	R ^a : randomForest ^b , randomForest ^c
MDI	Wrapper (Tree-based)	C/C++
WFM	Embedded (Tree-based; Cross-Validation)	C/C++

^aSoftware.

^bFunction.

^cLibrary.

classifier simultaneously (Blum and Langley, 1997; Kohavi and John, 1997). A major difference between the filter approach and the wrapper and embedded approaches is that the difference of means is considered in the filter approach, but the contribution of the features in classification is considered in the wrapper and embedded approaches. Table 6.1 summarizes various algorithms for feature selection and ranking by category.

6.3.2 Comparison of Feature Selection Methods

Current chemotherapy enables a high percentage of patients with AML to enter complete remission, but a large number of them experience relapse with resistant disease (Yagi et al., 2003). Due to the wide heterogeneity of AML, predicting a patient's risk for treatment failure or relapse at the time of diagnosis is critical for an optimal treatment. Baek et al. (2008) investigated this gene expression data consisting of 53 AML pediatric patients (less than 15 years old) with an oligonucleotide microarray containing 12,566 probe sets in order to identify genes that are associated with prognoses of pediatric AML patients. The dataset is available at <ftp://ftp.ncbi.nih.gov/pub/geo/DATA/SOFT/GDS/GDS1059.soft.gz>.

Patients with complete remission (CR) for more than three years are classified as good prognosis, while patients experiencing induction failure or relapse within one year of the first CR are considered as poor prognosis (R: relapsed). The dataset consists of 28 CRs and 25 Rs. Using classification methods, we can find potential prognostic biomarkers associated with pediatric AML prognosis. Hence, prognostic biomarkers can be used to predict outcomes in pediatric AML and to formulate individual risk-adjusted treatment strategies for pediatric AML patients.

Figure 6.4 shows the selected 20 genes in classification ranked with the t -test, weighted frequency measure from CERP, and the BW ratio. The performance of the feature selection methods applied with the classification algorithms for the pediatric AML data showed about 70% generalized accuracy (Baek et al., 2008).

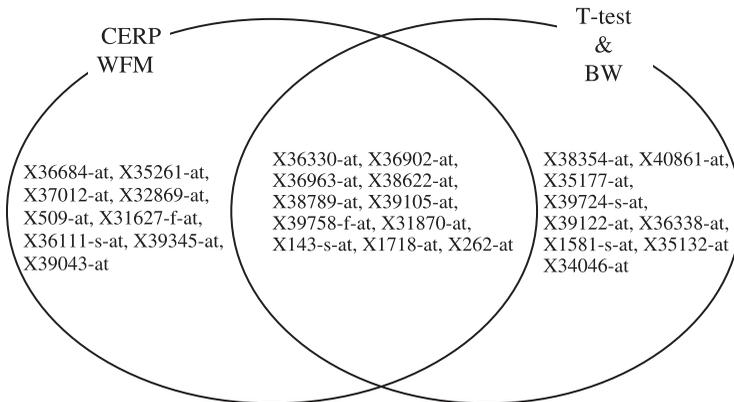


Figure 6.4 Venn diagram for selected 20 genes for pediatric AML data. The probe set IDs are listed in the diagram. Since the t -test and the BW ratio share common genes, the t -test and BW are combined in the diagram.

Among 20 genes selected, gene numbers 2230 (X36330-at), 2416 (X36902-at), 3231 (X38789-at), and 3382 (X39105-at) are commonly ranked highly in classification of pediatric AML prognosis by using feature selection methods of all three selection criteria. The t -test and the BW ratio show an agreement in gene selection. The top 20 genes selected by these two methods were the same with different ranks, but CERP-WFM selected approximately half of these genes in the top 20.

6.4 CROSS-VALIDATION

In order to appropriately evaluate the performance of a gene set from each method, cross-validation (CV) is widely used. CV utilizes resampling without replacement of the entire data to repeatedly develop classifiers on a training set and evaluate classifiers on a separate test set and then averages the procedure over the resampling.

If data are large, a subset of the data can be set aside for validation of the model to assess the performance of the classification algorithms. However, in general, a dataset is often insufficient to satisfy the need, and it is usually not possible to obtain such a dataset. A popular method to remedy this difficulty is CV.

In V -fold CV, the data are randomly divided into V roughly equal subsets (see Table 6.2). In each CV run, a set of genes is selected by each method from a training set of data ($V - 1$ subsets together). Using a set of selected genes, each classifier is built and a separate test set of data (the remaining subset) is applied to the model. This process is repeated V times and the V estimates are combined to evaluate the performance of the model. This completes one trial of the V -fold CV.

With $V = N$ (the total number of samples), commonly referred to as leave-one-out cross-validation (LOOCV), each CV run uses $N - 1$ samples for training and the remaining one sample for testing. Since the N training samples are so similar in each run, CV is approximately unbiased for the true prediction error. However, the variance of the true prediction error would be large and it is also computationally expensive. In practice, the preference of the number of folds depends on the size of the dataset. Tenfold CV has become generally accepted.

Another validation approach is the holdout validation. Holdout validation is not considered a CV because the data are not crossed over. The data are randomly divided into a training set and a test set. Typically less than one-third of the data is chosen as the test set.

TABLE 6.2 V -Fold CV Procedure

CV run	Data subset				
	1	2	...	$V - 1$	V
1	Train	Train	...	Train	Test
2	Train	Train	...	Test	Train
\vdots	\vdots	\vdots	\vdots	\vdots	\vdots
$V - 1$	Train	Test	...	Train	Train
V	Test	Train	...	Train	Train

Ambroise and McLachlan (2002) address that CV may cause selection bias in gene selection and leads too optimistic test error rate with few genes. If the CV error is calculated within the feature-selection process, there may be a selection bias in it when it is used as an estimate of the prediction error. To correct for this selection bias, it is necessary that CV be used external to the gene selection process or gene selection should be performed in the learning phase of CV.

For this purpose a validation set can be set aside from the training and testing sets. In this case the data are split into three parts: training set, test set, and validation set. The validation set is not a part of CV. The validation set is required in order to avoid overfitting or choose the best input parameters for a classification model. In this scheme, first the training data in CV is used to estimate the accuracy or other measures. Second, the validation set is used to choose the best parameters or adjust the structure of the model. This is repeated for many parameter choices and the best choice is selected on the validation set. This best choice of parameters is used to model it for estimation on the test data.

6.5 ENHANCEMENT OF CLASS PREDICTION BY ENSEMBLE VOTING METHODS

6.5.1 Mathematical Justification

We introduced several classification methods based on ensemble voting in Section 6.2. In Section 6.6, we will see that these methods tend to perform well compared to methods based on a single classifier. In this section we demonstrate why and how the ensemble approach enhances the class prediction.

A motivation for ensembles is that a combination of the outputs of many weak classifiers produces a powerful committee (Hastie et al., 2001). We assume independence among the n classifiers, where n is odd. We note that making n odd prevents ties.

Let X_i denote a random variable indicating a correct classification by the i th classifier. If the prediction accuracy of each classifier is p , then $X_i \sim \text{Bernoulli}(p)$. The number of accurate classifications by ensemble majority voting is $Y = \sum_{i=1}^n X_i \sim \text{binomial}(n, p)$. We let $n = 2k + 1$, where k is a nonnegative integer and define $A_n = P(Y \geq k + 1)$. Then the prediction accuracy of the ensemble classification by a majority voting is

$$A_n = \sum_{i=k+1}^n \binom{n}{i} p^i (1-p)^{n-i} \quad (6.1)$$

Lam and Suen (1997) showed that $A_{2k+1} = 0.5$ for $k = 0, 1, \dots$ when $p = 0.5$; the sequence $\{A_{2k+1}\}$ is strictly increasing when $p > 0.5$ and strictly decreasing when $p < 0.5$. If n is large, then $Y \xrightarrow{d} N[np, np(1-p)]$ by the central limit theorem. Ahn et al. (2007) showed that $\lim_{k \rightarrow \infty} A_{2k+1} = 1$, and the prediction accuracy of the ensemble voting method converges to 1 when $p > 0.5$.

If the classifiers in the ensemble are correlated, then we can use the beta-binomial model (Williams, 1975). This model allows only positive correlation ρ in order to satisfy $\text{Var}(p) > 0$. Prentice (1988) showed that the beta-binomial model

may be extended to cases where $\rho < 0$ for certain values. His extended beta-binomial model is valid when $\rho \geq \max\{-p(n-p-1)^{-1}, -(1-p)[n-(1-p)-1]^{-1}\}$.

The improvement of the ensemble accuracy discussed above is valid under the assumption of equal accuracy of the base classifiers and equal correlation among the classifiers. Without these constraints, Breiman (2001) obtained the upper bound for the generalization error when the accuracy of each classifier is at least a half. According to this theorem, the ensemble accuracy converges to 1 when the classifiers are independent. When the classifiers are correlated, the accuracy will converge to a point within the range between $1 - \bar{\rho}(1 - s^2)/s^2$ and 1, where s is the strength of the set of classifiers and $\bar{\rho}$ is the average correlation of the tree classifiers.

6.5.2 Example

Table 6.3 illustrates the prediction accuracy obtained by ensemble majority voting. When $\rho = 0$, the standard binomial probability in Eq. (6.1) is used for $n \leq 25$ and the normal approximation is used for a larger n . The beta-binomial model is used when the correlation is positive, and the extended beta-binomial model is used when the correlation is negative. The table illustrates that negatively correlated classifiers improve the prediction accuracy more rapidly than the independent classifiers,

TABLE 6.3 Enhancement of Prediction Accuracy by Ensemble Majority Voting

n	p	p (Prediction accuracy of each base classifier)						
		0.50	0.55	0.60	0.70	0.80	0.90	0.95
3	-0.05	0.5	0.58	0.66	0.80	0.91	0.98	NA ^a
	0	0.5	0.57	0.67	0.78	0.90	0.97	0.99
	0.1	0.5	0.57	0.64	0.76	0.87	0.95	0.98
	0.3	0.5	0.56	0.62	0.73	0.84	0.93	0.97
7	-0.025	0.5	0.62	0.73	0.90	0.98	NA	NA
	0	0.5	0.61	0.71	0.87	0.97	1.00	1.00 ^b
	0.1	0.5	0.59	0.67	0.81	0.92	0.98	0.99
	0.3	0.5	0.57	0.63	0.75	0.86	0.94	0.97
15	-0.01	0.5	0.67	0.81	0.96	1.00	NA	NA
	0	0.5	0.65	0.79	0.95	1.00	1.00	1.00
	0.1	0.5	0.60	0.70	0.85	0.95	0.99	1.00
	0.3	0.5	0.57	0.64	0.76	0.87	0.95	0.98
25	-0.01	0.5	0.72	0.88	0.99	NA	NA	NA
	0	0.5	0.69	0.85	0.99	1.00	1.00	1.00
	0.1	0.5	0.61	0.71	0.87	0.96	0.99	1.00
	0.3	0.5	0.57	0.64	0.77	0.87	0.95	0.98
101	0	0.5	0.84	0.98	1.00	1.00	1.00	1.00
	0.1	0.5	0.62	0.73	0.89	0.97	1.00	1.00
	0.3	0.5	0.57	0.64	0.77	0.88	0.95	0.98

^aNot available using the extended beta-binomial model by Prentice (1988).

^bGreater than or equal to 0.995.

while the improvement slows down when the correlation increases. For example, when the prediction accuracy of each classifier is 0.95, the class prediction accuracy by the ensemble voting method reaches 1.00 with $n = 7$ when the classifiers are independent. In contrast, the accuracy of the ensemble voting reaches only 0.98 with $n = 101$ when the correlation is 0.3. This result implies that the ensemble voting can be improved in terms of class prediction accuracy by avoiding high correlation caused by the overlap of the predictor variables. Note that the correlation cannot be highly negative under the extended negative binomial model due to the lower bound given in Section 6.5.1. The lower bound depends on n , and it approaches zero as n increases.

6.6 COMPARISON OF CLASSIFICATION METHODS USING HIGH-DIMENSIONAL DATA

6.6.1 Breast Cancer Data

Gene expression data might be used to improve disease prognosis in order to prevent some patients from having to undergo painful unsuccessful therapies and unnecessary toxicity. For example, adjuvant chemotherapy for breast cancer after surgery could reduce the risk of distant metastases; however, 70–80% of patients receiving this treatment would be expected to survive metastasis free without it (van't Veer et al., 2002). The strongest predictors for metastases, such as lymph node status and histological grade, fail to classify accurately breast tumors according to their clinical behavior (McGuire, 1991; van't Veer et al., 2002).

Predicting patient response to therapy or the toxic potential of drugs based on high-dimensional data is a common goal of biomedical studies. Classification algorithms can be used to process high-dimensional genomic data for better prognostication of disease progression and better prediction of response to therapy to help individualize clinical assignment of treatment. The predictive model built is required to be highly accurate, since the consequence of misclassification may result in suboptimal treatment or incorrect profile. Commonly, there are numerous genomic and clinical predictor variables over a relatively small number of patients for biomedical applications, which presents challenges for most traditional classification algorithms to avoid overfitting the data.

The objective of van't Veer et al. (2002) was to use gene expression data to identify patients who might benefit from adjuvant chemotherapy according to prognostication of distant metastases for breast cancer. The van't Veer et al. data contain 78 primary breast cancers (34 from patients who developed distant metastases within five years (poor prognosis) and 44 from patients who continue to be disease free (good prognosis) after a period of at least five years). These samples have been selected from patients who were lymph node negative and under 55 years of age at diagnosis. Out of approximately 25,000 gene expression levels, approximately 5000 significantly regulated genes (at least a twofold difference and a p -value of less than 0.01) in more than three tumors out of 78 were selected (van't Veer et al., 2002).

Sensitivity (SN) and specificity (SP) as well as positive predictive value (PPV) and negative predictive value (NPV) are primary criteria used in the evaluation of the

TABLE 6.4 Performance of Classification Algorithms for the Breast Cancer Genomic Data Based on 20 Repetitions of 10-Fold CV

Algorithms	Accuracy	Sensitivity	Specificity	PPV	NPV
ANN	0.659 (0.022)	0.629 (0.022)	0.693 (0.021)	0.624 (0.022)	0.704 (0.021)
C-T CERP ^a	0.653 (0.021)	0.543 (0.039)	0.738 (0.036)	0.616 (0.031)	0.676 (0.018)
RF	0.625 (0.019)	0.468 (0.032)	0.747 (0.032)	0.589 (0.029)	0.645 (0.014)
AdaBoost	0.588 (0.041)	0.321 (0.089)	0.794 (0.069)	0.550 (0.094)	0.603 (0.028)
LogitBoost	0.652 (0.049)	0.556 (0.084)	0.726 (0.061)	0.611 (0.067)	0.680 (0.043)
SVM	0.565 (0.029)	0.396 (0.053)	0.697 (0.027)	0.501 (0.042)	0.599 (0.025)
DLDA	0.625 (0.019)	0.524 (0.023)	0.703 (0.026)	0.578 (0.026)	0.656 (0.015)
SC	0.609 (0.019)	0.506 (0.026)	0.689 (0.023)	0.557 (0.024)	0.643 (0.016)
<i>k</i> -NN ^b	0.581 (0.029)	0.360 (0.056)	0.751 (0.045)	0.528 (0.055)	0.603 (0.021)
NB	0.617 (0.018)	0.456 (0.032)	0.742 (0.018)	0.577 (0.026)	0.639 (0.015)
CART	0.546 (0.028)	0.175 (0.058)	0.832 (0.047)	0.446 (0.084)	0.566 (0.018)
CRUISE	0.551 (0.048)	0.215 (0.100)	0.810 (0.059)	0.456 (0.112)	0.573 (0.034)
QUEST	0.565 (0.044)	0.228 (0.080)	0.826 (0.077)	0.510 (0.117)	0.581 (0.027)

Note: Numbers in parentheses are the standard deviation (SD).

^aThe average number of partitions: 72.1.

^bMode of *k* used in the learning phase of *k*-NN.

performance of a classification algorithm. SN is the proportion of correct positive classifications out of the number of true positives. SP is the proportion of correct negative classifications out of the number of true negatives. The accuracy is the total number of correct classifications out of the total number of samples. PPV is the probability that a patient is positive given a positive prediction, while NPV is the probability that a patient is negative given a negative prediction. Algorithms with high SN and high SP as well as high PPV and high NPV, which will have high accuracy, are obviously desirable.

Table 6.4 shows performance of classification algorithms for the van't Veer et al. (2002) breast cancer genomic data based on 20 repetitions of 10-fold cross-validation. The balance between SN and SP is reasonably good for ANN, CERP, DLDA, LogitBoost, and SC. Sensitivities of ANN, CERP, DLDA, LogitBoost, and SC are higher (over 50%) than the rest (under 50%). The positive predictive values from ANN, CERP, and LogitBoost are higher (over 60%) than the others. ANN shows an excellent balance between sensitivity and specificity. Accuracies of the single optimal trees are lower than those of other classification algorithms, and the balance between SN and SP in these single trees is unsatisfactory. Logistic regression is not included in this comparison because the model has a restriction that the sample size needs to be larger than the number of predictors.

6.6.2 Gastrointestinal Bleeding Data

Acute gastrointestinal bleeding (GIB) is an increasing health care problem due to rising nonsteroidal anti-inflammatory drug (NSAID) use and an aging population (Rockall

et al., 1995). Further reductions in mortality will most likely require introduction of novel strategies to aid identification of the cohort requiring aggressive resuscitation and endoscopic intervention to prevent complications and death from ongoing bleeding (Baradarian et al., 2004; Elta, 2004). Delays in intervention usually result from failure to adequately recognize the source and severity of the bleed. Although several models and scores have been developed to risk stratify patients, no single model aids in the identification of the subgroup of patients presenting with acute upper or lower GIB most likely to benefit from urgent intervention (Das et al., 2003; Das and Wong, 2004). Chu et al. (2008) conducted a study to utilize mathematical models to formulate a decision support system utilizing clinical and laboratory information available within a few hours of patient presentation to predict the source and need for intervention and disposition in patients with acute upper, mid, and lower GIB.

Patients with acute GIB were identified from the hospital medical records database using International Classification of Diseases ICD-9 codes for GIB. A total of 189 patients meeting inclusion and exclusion criteria were identified retrospectively in hospital electronic medical records. The definitive source of bleeding was the irrefutable identification of a bleeding source at upper endoscopy, colonoscopy, small bowel enteroscopy, or capsule endoscopy. Variables utilized to predict the source of GIB included past medical history and presenting symptoms (prior history of GIB, hematochezia, hematemesis, melena, syncope/presyncope), risk factors [risk for stress ulceration, cirrhosis, acetylsalicylic acid (ASA)/NSAID use], physical exam and laboratory tests [blood pressure (SBP/DBP), heart rate, orthostasis, nasogastric lavage, rectal exam, platelet count, creatinine, blood urea nitrogen (BUN), and international normalized ratio (INR)]. Six classification methods—RF, SVM, SC, LDA, *k*-NN, and ANN—are trained and tested. For the source of bleeding response, there were 30 input neurons, 11 hidden neurons (neurons in the hidden layer), and 3 output neurons in ANN.

Table 6.5 summarizes the results for each outcome prediction variable for the evaluation step. Accuracies obtained using the ANN model was inferior to those using SVM, RF, and LDA. Based on McNemar's test, *k*-NN performed the worst for predicting source of bleeding. LDA appeared to demonstrate a good overall performance with regards to sensitivity, specificity, PPV, NPV, and accuracy (Table 6.5). ROC (receiver operating characteristic) curves are calculated and areas under the curve (AUC) are compared for each of the models. An ROC curve is a

TABLE 6.5 Performance of Models for Output Source of Bleeding

	ACC	SN	SP	PPV	NPV	AUC
RF	0.943 (0.007)	0.980 (0.004)	0.932 (0.007)	0.967 (0.005)	0.959 (0.006)	0.998
SVM	0.930 (0.007)	0.965 (0.005)	0.945 (0.007)	0.973 (0.005)	0.932 (0.007)	0.979
SC	0.914 (0.008)	0.965 (0.005)	0.890 (0.009)	0.946 (0.007)	0.927 (0.008)	0.978
LDA	0.931 (0.007)	0.965 (0.005)	1.000 (0.000)	1.000 (0.000)	0.935 (0.007)	0.987
<i>k</i> -NN	0.697 (0.013)	0.901 (0.009)	0.287 (0.013)	0.717 (0.013)	0.591 (0.014)	0.658
ANN	0.917 (0.008)	0.972 (0.005)	0.936 (0.007)	0.968 (0.005)	0.944 (0.007)	0.999

Note: Numbers in parentheses are the standard errors.

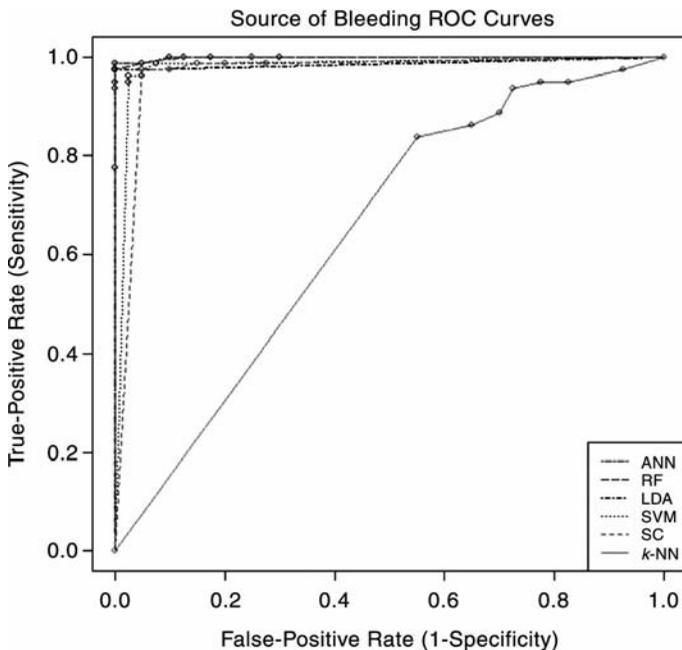


Figure 6.5 ROC curves for predicting source of bleeding (evaluation step).

graphical plot of the sensitivity versus false-positive rate (1 minus specificity) as its discrimination threshold is varied. AUC is obtained using the Mann–Whitney statistic, which is equivalent to the area under the ROC curve (Bamber, 1975). In addition to accuracies, a balance between sensitivity and specificity is considered by evaluating ROC curves. ROC curves were constructed (Fig. 6.5), and overall RF and ANN have the highest AUC, followed by SVM and LDA. Overall, data from evaluation and validation steps suggest that the RF model consistently performs the best.

6.7 SOFTWARE EXAMPLES FOR CLASSIFICATION METHODS

Table 6.6 shows classification models by category. The R package can be freely downloaded from <http://cran.cnr.berkeley.edu>. Commonly used basic functions are included in the package downloaded from the website. After installing R, special software packages need to be downloaded additionally as needed. To download a package, click “package” on the top-down menu and choose “Install package.” Select one of the sites listed, and then choose the package that you want to download. Once a software package is installed, it is listed when the user clicks “package” on the top-down menu and selects “load.” Choose the package you need from the list before you use it. A description for using selected R software packages is given below.

TABLE 6.6 Classification Models by Category

Category	Model	Software	Computational cost
Parametric	Logistic regression	R library (statmod)	Low
	LDA	R library (mass)	Low
	DLDA	R library (sma)	
	Naive Bayes	R library (e1071)	Low
Nonparametric	k -NN	R library (class)	Low
	Shrunken centroid	R library (pamr)	Low
Tree based	CART	R library (rpart)	High
	CRUISE	Fortran 90 program ^a	High
	QUEST	Fortran 90 program ^a	High
Aggregation (kernel based)	SVM	R library (e1071)	High
	ANN	STATISTICA Neural Networks	High
Ensemble	RF	R library (randomForest)	High
	Boosting	R library (boost)	High
	CERP	C program (C-T CERP) ^b R program (LR-T CERP) ^c	High

^aExecutable file of CRUISE can be obtained from <http://www.stat.wisc.edu/~loh/cruise.html>; executable file of QUEST can be obtained from <http://www.stat.wisc.edu/~loh/quest.html>.

^bExecutable file of C-T CERP can be obtained from <http://www.ams.sunysb.edu/~hahn/research/CERP.html>.

^cR code of LR-T CERP can be provided upon request.

1. Logistic Regression The R logistic regression package `statmod` includes the following `glm` function:

```
output<-glm(y~x, family=binomial)
```

where y is a response vector, x is a matrix of the predictor variables, and “family” can be chosen as “binomial” for logistic regression. The estimate of the regression coefficient can be obtained by

```
Beta<-output$coef
```

2. DLDA The R LDA packages are “mass” for LDA and “sma” for DLDA. The `sma` function `stat.diag.da` implements a simple Gaussian maximum-likelihood discriminant rule for diagonal class covariance matrices. The usage is

```
stat.diag.da(ls, cll, ts, pool=1)
```

where ls is a learning set data matrix, with rows corresponding to cases and columns to predictor variables, cll is class labels for learning set with consecutive integers, ts is a test set data matrix, with rows corresponding to cases and columns to predictor variables, and “pool” is a logical flag. If `pool = 1`, the covariance matrices are assumed to be constant across classes and the

discriminant rule is linear in the data. If $\text{pool} = 0$, the covariance matrices may vary across classes and the discriminant rule is quadratic in the data.

3. ***k*-NN** The name of the R *k*-NN package is “class.” An example of the command is

```
knn(train, test, cl, k=a, prob=FALSE, use.all=TRUE)
```

where “train” is a matrix or a data frame of training set cases and “test” is a matrix or a data frame of test set cases. A vector will be interpreted as a row vector for a single case. Here *cl* is a factor of true classifications of training set, *k* is the number of neighbors considered; the optimal value of *k* can be found by cross-validation; *prob* is a logical variable. If this is true, the proportion of the votes for the winning class is returned as attribute *prob*. The indicator *use.all* controls handling of ties. If true, all distances equal to the *k*th largest are included. If false, a random selection of distances equal to the *k*th is chosen to use exactly *k* neighbors.

4. **SVM** The R SVM function is included in the package *e1071*. The object function is created by

```
fit<-svm(learnx ~., learny=learndata, kernel=choice of
kernel)
```

where *learnx* is the response from the learning set, *learny* is a matrix containing the predictor variables from the learning set, and “kernel” is a choice of a kernel. Linear kernel is “linear” and the radial basis kernel is “radial.” The output from *svm* can be used to predict the data as

```
yhat<-predict(fit, testx, decision.values, probability, na.
action)
```

where “fit” is an object of class *svm* and *testx* is a matrix containing the new input data. A vector will be transformed to an $n \times 1$ matrix. The logical indicator *decision.values* controls whether class probabilities should be computed and returned, and “probability” is also a logical variable indicating whether class probabilities should be computed and returned. This option is possible only if the model was fitted with the probability option enabled. Finally, *na.action* is a function to specify the action to be taken if NAs are found. The default action is *na.omit*, which leads to rejection of cases with missing values on any required variable.

5. **Boosting** The name of the R boosting package is “boost.” It contains four boosting functions: *adaboost*, *logitboost*, *l2boost*, and *bagboost*. To run *adaboost*, for example, the command is given as

```
fit<-adaboost(xlearn, ylearn, xtest, presel=200,
mfinal=100)
```

The result is assigned to “fit.” Here, *xlearn* is a matrix of the predictors in the learning set, *ylearn* is the response vector in the learning set, *xtest* is a matrix of the predictors in the test set, *presel* is the number of variables to be

preselected. Default of `presel` is all the variables without a preselection. The `mfinal` is the desired maximum number of iteration. Predicted values for the test set are classified into either 0 or 1 if the value of “fit” is less than 0 or not, respectively. The other three boosting functions can be run similarly, but `bagboost` requires an additional input parameter, “bag,” which is the desired number of bagging.

- 6. Random Forest** The `randomForest` function is included in the `randomForest` package in R. In a learning phase, it can be trained as

```
fit <- randomForest(y~., learn.data, ntree, mtry, importance)
```

where `y` is a response vector, `learn.data` is a learning data set, `ntree` is the number of trees to grow (default is 500), `mtry` is the number of variables randomly sampled as candidates at each split (default is \sqrt{m} , where m is the number of predictor variables), and “importance” is the logical variable if importance of predictors is assessed. The test data set can be predicted as

```
predict(fit, test.data, type="vote", norm.votes=FALSE)
```

where “fit” is an object of class `randomForest`, `test.data` is the test data set, “type” indicates a type of output, one of “response,” “prob,” or “vote,” and `norm.votes` is a logical variable if vote counts need to be normalized.

- 7. Naïve Bayes** To use the naive Bayes function in R, the package `e1071` should be included using `library(e1071)`. With a learning dataset, this function can be used as

```
fit <- naiveBayes(y~., learn.data)
```

where `y` is a class vector and `learn.data` is a learning dataset. A class variable can be predicted using

```
predict(fit, newdata, type)
```

where “fit” is an object of class `naiveBayes`, `newdata` is the test data set, and “type” is either “class” or “raw.”

- 8. CART** CART can be implemented by using the library `rpart` in R and by fitting

```
fit <- rpart(y~., data, method, parms, control=temp.rpart)
```

where `y` is a response vector, “data” is a learning data set, “method” can be one of “anova,” “poisson,” “class,” or “exp,” and `parms` indicates optional parameters for the splitting function. For classification, `list(split='gini')` can be used. The “control” option controls details of the `rpart` algorithm by using the `rpart.control` function such as

```
rpart.control(xval=10, minbucket=3)
```

where `xval` indicates the number of cross-validations and `minbucket` is the minimum number of observations in any terminal node.

REFERENCES

- Ahn, H., and Chen, J. J. (1997). Tree-structured logistic model for over-dispersed binomial data with application to modeling developmental effects. *Biometrics*, **53**: 435–455.
- Ahn, H., and Loh, W.-Y. (1994). Tree-structured proportional hazards regression modeling. *Biometrics*, **50**: 471–485.
- Ahn, H., Moon, H., Fazzari, M. J., Lim, N., Chen, J. J., and Kodell, R. L. (2007). Classification by ensembles from random partitions of high-dimensional data. *Computat. Stat. Data Anal.*, **51**: 6166–6179.
- Akaike, H. (1974). A new look at the statistical identification model. *IEEE Trans. Automatic Control*, **19**: 716–723.
- Ambrose, C., and McLachlan, G. J. (2002). Selection bias in gene extraction on the basis of microarray gene-expression data. *Proc. Nat. Acad. Sci.*, **99**: 6562–6566.
- Baek, S., Moon, H., Ahn, H., Kodell, R. L., Lin, C. J., and Chen, J. J. (2008). Identifying high-dimensional biomarkers for personalized medicine via variable importance ranking. *J. Biopharm. Stat.*, **18**: 901–914.
- Bamber, D. (1975). The area above the ordinal dominance graph and the area below the receive operating graph. *J. Math. Psychol.*, **12**: 387–415.
- Baradaran, R., et al. (2004). Early intensive resuscitation of patients with upper gastrointestinal bleeding decreases mortality. *Am. J. Gastroenterol.*, **99**(4): 619–622.
- Bauer, E., and Kohavi, R. (1999). An empirical comparison of voting classification algorithms: Bagging, boosting and variants. *Machine Learning*, **36**: 105–139.
- Beale, R., and Jackson, T. (1990). *Neural Computing: An Introduction*. Boca Raton, FL: CRC Press.
- Bishop, C. M. (1995). *Neural Networks for Pattern Recognition*. Oxford: Oxford University Press.
- Blum, A., and Langley, P. (1997). Selection of relevant features and examples in machine learning. *Artificial Intell.*, **97**: 245–271.
- Breiman, L. (1996). Bagging predictors. *Machine Learning*, **24**: 123–140.
- Breiman, L. (2001). Random forest. *Machine Learning*, **45**: 5–32.
- Breiman, L., Friedman, J. H., Olshen, R. A., and Stone, C. J. (1984). *Classification and Regression Trees*. Belmont, CA: Wadsworth.
- Burnham, K. P., and Anderson, D. R. (2002). *Model Selection and Multinomial Inference: A Practical Information-Theoretic Approach*, 2nd ed. New York: Springer Verlag.
- Chu, A., Ahn, H., Halwan, B., Kalmin, B., Artifon, E. L., Barkun, A., Lagoudakis, M. G., and Kumar, A. (2008). A decision support system to facilitate management of patients with acute gastrointestinal bleeding. *Artif. Intell. Med.*, **42**: 247–259.
- Das, A., et al. (2003). Prediction of outcome in acute lower-gastrointestinal haemorrhage based on an artificial neural network: Internal and external validation of a predictive model. *Lancet*, **362**: 1261–1266.
- Das, A., and Wong, R. C. (2004). Prediction of outcome of acute GI hemorrhage: A review of risk scores and predictive models. *Gastrointest. Endosc.*, **60**(1): 85–93.
- Demicheli, F., Magni, P., Piergiorgi, P., Rubin M. A., and Bellazzi, R. (2006). A hierarchical naive Bayes model for handling sample heterogeneity in classification problems: An application to tissue microarrays. *BMC Bioinformatics*, **7**: 514.
- Dettling, M., and Bühlmann, P. (2003). Boosting for tumor classification with gene expression data. *Bioinformatics*, **19**: 1061–1069.
- Diaz-Uriarte, R., and Álvarez de Andrés, S. (2006). Gene selection and classification of microarray data using random forest. *BMC Bioinformatics*, **7**(3): doi:10.1186/1471-2105-7-3.
- Dudoit, S., Fridlyand, J., and Speed, T. P. (2002). Comparison of discrimination methods for the classification of tumors using gene expression data. *J. Am. Statist. Assoc.*, **97**: 77–87.
- Elta, G. H. (2004). Urgent colonoscopy for acute lower-GI bleeding. *Gastrointest. Endosc.*, **59**(3): 402–408.
- Faussett, L. (1994). *Fundamentals of Neural Networks: Architectures, Algorithms, and Applications*. Upper Saddle River, NJ: Prentice Hall.
- Freund, Y., and Schapire, R. (1996). Experiments with a new boosting algorithm. In *Machine Learning: Proceedings of the Thirteenth International Conference*, M. Kaufman (Ed.). San Francisco, CA: Kluwer, pp. 148–156.
- Freund, Y., and Schapire, R. (1997). A decision-theoretic generalization of online learning and an application to boosting. *J. Comput. Syst. Sci.*, **55**: 119–139.

- Friedman, J. (1997). On bias, variance, 0/1-loss, and the curse-of-dimensionality. *Data Mining Knowledge Discov.*, **1**: 55–77.
- Friedman, J., Hastie, T., and Tibshirani, R. (2000). Additive logistic regression: A statistical review of boosting. *Ann. Statist.*, **28**: 337–407 (with discussion).
- Golub, T. R., et al. (1999). Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. *Science*, **286**(5439): 531–537.
- Guyon, L., Weston, J., Barnhill, S., and Vapnik, V. (2002). Gene selection for cancer classification using support vector machines. *Machine Learning*, **46**: 389–422.
- Hastie, T., Tibshirani, R., and Friedman, J. (2001). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. New York: Springer.
- Hawkins, D. M. (2004). The problem of overfitting. *J. Chem. Inform. Comput. Sci.*, **44**: 1–12.
- Ho, T. K. (1998). The random subspace method for constructing decision forests. *IEEE Trans. Pattern Anal. Machine Intell.*, **20**: 832–844.
- Kim, H., and Loh, W.-Y. (2001). Classification trees with unbiased multiway splits. *J. Am. Statist. Assoc.*, **96**: 589–604.
- Kohavi, R., and John, G. H. (1997). Wrappers for feature subset selection. *Artificial Intell.*, **97**: 273–324.
- Lam, L., and Suen, C. Y. (1997). Application of majority voting to pattern recognition: An analysis of its behavior and performance. *IEEE Trans. Syst. Man Cybernet.*, **27**: 553–568.
- Liu, H., Li, J., and Wong, L. (2002). A comparative study on feature selection and classification methods using gene expression profiles and proteomic patterns. *Genome Informat.*, **13**: 51–60.
- Loh, W.-Y., and Shih, Y. S. (1997). Split selection methods for classification trees. *Statist. Sinica*, **7**: 815–840.
- Loh, W.-Y., and Vanichsetakul, N. (1988). Tree-structured classification via generalized discriminant analysis (with discussion). *J. Am. Statist. Assoc.*, **83**: 715–728.
- McGuire, W. L. (1991). Breast cancer prognostic factors: Evaluation guidelines. *J. Nat. Cancer Inst.*, **83**: 154–155.
- Michalski, R. S., and Kaufman, K. (2001). Learning patterns in noisy data: The AQ approach. In *Machine Learning and its Applications*, G. Paliouras, V. Karkaletsis, and C. Spyropoulos (Eds.). Berlin: Springer-Verlag, pp. 22–38.
- Michiels, S., Koscielny, S., and Hill, C. (2005). Prediction of cancer outcome with microarrays: A multiple random validation strategy. *Lancet*, **365**(9458): 488–492.
- Miller, A. (2002). *Subset Selection in Regression*, 2nd ed. Los Angeles, CA: Chapman and Hall/CRC.
- Moon, H., Ahn, H., Kodell, R. L., Lin, C. J., Baek, S., and Chen, J. J. (2006). Classification methods for the development of genomic signatures from high-dimensional genomic data. *Genome Biol.*, **7**: R121.1–R121.7.
- Morgan, J. N., and Sonquist, J. A. (1963). Problems in the analysis of survey data, and a proposal. *J. Am. Statist. Assoc.*, **58**: 415–434.
- Petricoin, E. F., et al. (2002a). Use of proteomic patterns in serum to identify ovarian cancer. *Lancet*, **395**(9306): 572–577.
- Petricoin, E. F., et al. (2002b). Serum proteomic patterns for detection of prostate cancer. *J. Nat. Cancer Inst.*, **94**: 1576–1578.
- Prentice, R. L. (1988). Correlated binary regression with covariates specific to each binary observation. *Biometrics*, **44**: 1033–1048.
- Rockall, T. A., Logan, R. F., Devlin, H. B., and Northfield, T. C. (1995). Incidence of and mortality from acute upper gastrointestinal haemorrhage in the United Kingdom. Steering Committee and members of the National Audit of Acute Upper Gastrointestinal Haemorrhage. *Br. Med. J.*, **311**: 222–226.
- Schapire, R. E., Freund, Y., Bartlett, P. A., and Lee, W. S. (1998). Boosting the margin: A new explanation for the effectiveness of voting methods. *Ann. Statist.*, **26**: 1651–1686.
- Sparano, J. A., Fazzari, M. J., and Childs, G. (2005). Clinical application of molecular profiling in breast cancer. *Future Oncol.*, **1**: 485–496.
- Tibshirani, R., Hastie, T., Narasimhan, B., and Chu, G. (2002). Diagnosis of multiple cancer types by shrunken centroids of gene expression. *Proc. Natl. Acad. Sci. U.S.A.*, **99**: 6567–6572.
- Tsai, C. A., Chen, C. H., Lee, T. C., Ho, I. C., Yang, U. C., and Chen, J. J. (2004). Gene selection for sample classifications in microarray experiments. *DNA Cell Biol.*, **23**: 607–614.

- van't Veer, L. J., et al. (2002). Gene expression profiling predicts clinical outcome of breast cancer. *Nature*, **415**: 530–536.
- Vapnik, V. (1995). *The Nature of Statistical Learning Theory*. New York: Springer.
- Williams, D. A. (1975). The analysis of binary responses from toxicological experiments involving reproduction and teratogenicity. *Biometrics*, **31**: 949–952.
- Yagi, T., et al. (2003). Identification of a gene expression signature associated with pediatric AML prognosis. *Blood*, **102**: 1849–1856.

MULTIDIMENSIONAL ANALYSIS AND VISUALIZATION ON LARGE BIOMEDICAL DATA*

Jinwook Seo

*Visual Processing Laboratory, School of Computer Science and Engineering,
Seoul National University, Gwanak-gu, Seoul, Korea*

Ben Shneiderman

*Human-Computer Interaction Lab & Department of Computer Science,
University of Maryland, College Park, MD, USA*

7.1 INTRODUCTION

The biomedical research community has witnessed tremendous advances in high-throughput biotechnologies in recent decades. We can take a snapshot of tens of thousands of genetic features in a clinical sample using a single gene chip. Moreover, we can examine millions of SNPs (single-nucleotide polymorphisms) on a single SNP chip. Consequently, more and more biomedical research studies utilize such high-throughput biotechnologies generating large datasets with many dimensions. Running high-throughput biomedical experiments with many samples is often a laborious and challenging process, but it often becomes even more challenging to analyze the quantitative data obtained from the experiment, which are large multidimensional datasets. Dealing with multidimensionality has been challenging to researchers in many disciplines due to the difficulty in humans comprehending more than three dimensions to discover meaningful features such as linear relationships, outliers, clusters, gaps, and so on. This difficulty is so well-recognized that it has a provocative name—“the curse of high dimensionality”—which signifies the difficulty in assessing the whole multidimensional space because of the exponential growth of volume as dimensionality increases.

Statistical analysis methods are certainly inevitable in the data analysis process. In previous chapters, we have covered important statistical concepts and methods that can be used in analyzing such large multidimensional datasets. However, these

*This chapter is a derivative work built upon a research paper published in the journal *Information Visualization* (Seo and Shneiderman, 2005).

statistical approaches often rely on certain summary values, which cannot provide insightful understanding and data exploration directly on such high-dimensional data. Therefore, in addition to these statistical techniques, multidimensional visualization techniques are critical to interrogate these high-dimensional, large biological data.

One of the commonly used methods to cope with multidimensionality is to use low-dimensional projections. Since human eyes and minds are effective in understanding one-dimensional (1D) histograms, two-dimensional (2D) scatterplots, and three-dimensional (3D) scatterplots, these representations are often used as a starting point. Users can begin by understanding the meaning of each dimension (since names can help dramatically, they should be readily accessible) and by examining the range and distribution (normal, uniform, erratic, etc.) of values in a histogram. Then experienced analysts suggest applying an orderly process to note exceptional features such as outliers, gaps, or clusters. Next, users can explore 2D relationships by studying 2D scatterplots and again use an orderly process to note exceptional features. Since computer displays are intrinsically 2D, collections of 2D projections have been widely used as representations of the original multidimensional data. This is imperfect since some features will be hidden, but at least users can understand what they are seeing and come away with some insights.

Advocates of 3D scatterplots argue that since the natural world is 3D, users can readily grasp 3D representations. However, there is substantial empirical evidence that for multidimensional abstract data (rather than 3D real objects such as chairs or skeletons) users struggle with occlusion and the cognitive burden of navigation as they try to find desired viewpoints. Advocates of higher dimensional displays have demonstrated attractive possibilities, but their strategies are still difficult to grasp for most users.

In this chapter, we will cover various visualization methods to help researchers understand large multidimensional datasets. We will start with traditional statistical low-dimensional projection methods that have been widely used to reveal patterns in multidimensional datasets, and then we will present a novel analysis framework to help users systematically explore 1D and 2D projections of large multidimensional datasets.

7.2 CLASSICAL MULTIDIMENSIONAL VISUALIZATION TECHNIQUES

Example 7.1 A biomedical researcher has a multidimensional dataset where he or she measured expression levels of 878 genes in 33 muscular dystrophy samples of three different types. He or she wants to visualize this multidimensional dataset to check the distribution patterns of the muscular dystrophy samples. If he or she is especially interested in patterns according to the variance in the dataset, what is an appropriate way to visualize the dataset?

7.2.1 Principal-Component Analysis

Most statistical packages and tools support principal-component analysis (PCA). Principal-component analysis itself is not a visualization method, but the

low-dimensional projections using two or three major principal components (i.e., eigenvectors) can make an efficient compact visualization for the original multidimensional dataset. The first eigenvector (with the largest eigenvalue) is the vector along which the original data items vary the most (i.e., the variance along the vector is the biggest). A famous statistical analysis tool called R has several routines that implement PCA. There are installation packages, tutorials, and manuals at <http://www.r-project.org/>. The following commands in R run a PCA and show a useful low-dimensional projection using two principal components as x - and y -axes:

```
> library(labdsv)
> data <- read.table("MDUA_R2.txt", header=TRUE,
  row.names=1)
> tdata <- t(data)
> result <- pca(tdata)
> plot(result)
> plot(result$scores[,1], result$scores[,2])
```

We first have to load the package (LabDSV) that has a PCA implementation. It should be noted that there are many other PCA implementations in R. Next, we load the dataset into R and then transpose the data matrix so that each muscular dystrophy sample takes each row in the matrix. After running a PCA on the transposed matrix, we can plot the result using the first two eigenvectors (having the biggest eigenvalues) as x - and y -axes (Fig. 7.1).

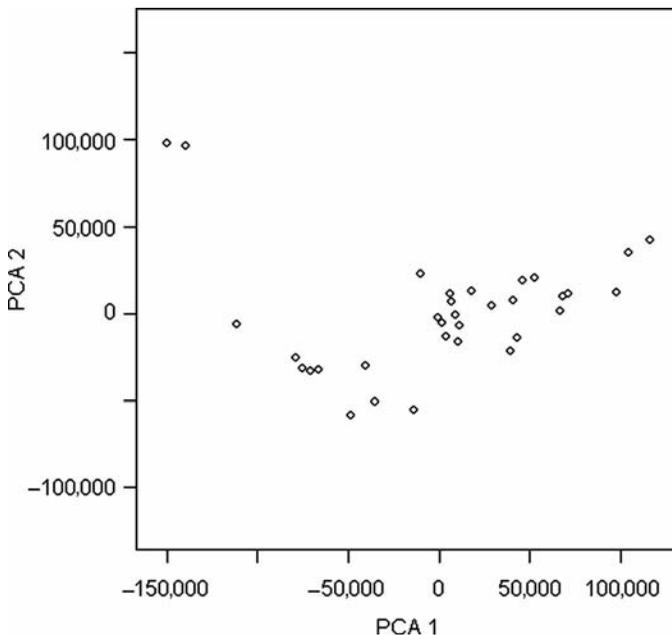


Figure 7.1 Scatterplot resulting from PCA for multidimensional dataset of 878 genes and 33 muscular dystrophy samples. The x -axis is the first principal component and the y -axis is the second principal component.

Example 7.2 The same researcher in Example 7.1 wants to visually examine the same multidimensional dataset, but now he or she is interested in a low-dimensional projection where the similarity/distance information in the original multidimensional dataset is preserved as much as possible. What is an appropriate way to visualize the dataset?

7.2.2 Multidimensional Scaling

Most statistical packages and tools also support multidimensional scaling (MDS). Multidimensional scaling has been used quite popularly to visualize multidimensional datasets (Fig. 7.2). We can use MDS to generate a 2D projection where distances/similarities among data items in the original multidimensional space are preserved as much as possible. In other words, MDS optimizes the following objective function: $F_{\text{MDS}} = \sum_{i \neq j} [d(i, j) - d'(i, j)]^2$, where $d(i, j)$ is the distance between i and j in the original multidimensional space and $d'(i, j)$ is the distance between i and j in the projected 2D space. R also has several routines that implement the multidimensional scaling. The following commands run an MDS and show a low-dimensional projection:

```
> euc_dist<- dist(tdata)
> mdsresult <- cmdscale(euc_dist, eig=FALSE, k=2)
> x <- mdsresult [,1]
> y <- mdsresult [,2]
> plot(x, y)
```

A slight change to the objective function for MDS will make the objective function for the *Sammon's mapping*, where the squared term in the objective function

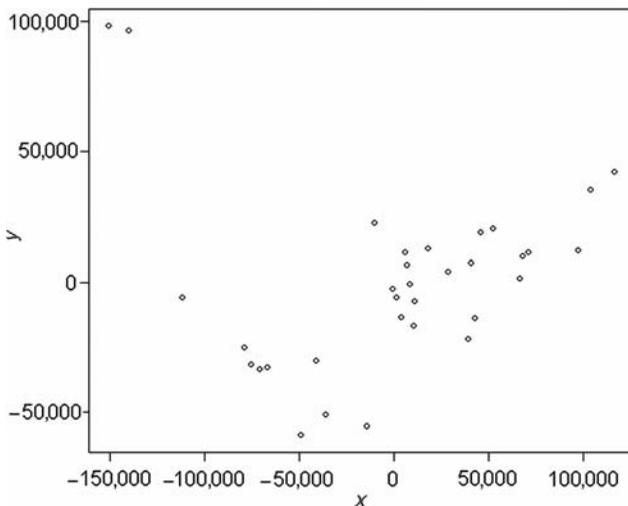


Figure 7.2 Scatterplot resulting from MDS analysis of 878 genes and 33 muscular dystrophy samples.

is divided by the distance in the original space in order to give more weight to the smaller distances in the original space. The function called `sammon` in R can be used to make Sammon's mappings for multidimensional datasets.

7.3 TWO-DIMENSIONAL PROJECTIONS

Two-dimensional projections have been utilized in many visualization tools and graphical statistics tools for multidimensional data analysis. As we have seen in the introduction of this chapter, projection techniques such as PCA (Hotelling, 1933), MDS (Torgerson, 1952), Sammon's mapping (Sammon, 1969), and parallel coordinates (Inselberg and Dimsdale, 1990) are used to find informative 2D projections of multidimensional datasets. It has been known that taking a look at only a single projection for a multidimensional dataset is not enough to discover all the interesting features in the original data since any projection may obscure some features (Friedman, 1987). Thus it is inevitable for users to scrutinize a series of projections to reveal the features of the dataset.

Since 2D presentations offer ample power while maintaining comprehensibility, many variations have been proposed. We can distinguish the three categories of 2D presentations by the way axes are composed:

1. Non-axis-parallel projection methods use a (linear/nonlinear) combination of two or more dimensions for an axis of the projection plane. Principal-component analysis is a well-established technique in this category.
2. Axis-parallel projection methods use existing dimensions as axes of the projection plane. One of the existing dimensions is selected as the horizontal axis, and another as the vertical axis, to make a familiar and comprehensible presentation. Sometimes, other dimensions can be mapped as color, size, length, angle, and so on.
3. Novel methods use axes that are not directly derived from any combination of dimensions. For example, the parallel coordinate presentation is a powerful concept in which dimensions are aligned sequentially and presented perpendicular to a horizontal axis (Inselberg and Dimsdale, 1990). Parallel coordinate visualization is available at several visualization tools such as TimeSearcher (<http://www.cs.umd.edu/hcil/timesearcher/>), Hierarchical Clustering Explorer (HCE, <http://www.cs.umd.edu/hcil/hce/>), and XmdvTool (<http://davis.wpi.edu/~xmdv/>). A recent survey of multi-dimensional visualization techniques belonging to category 3 is found in Ferreira de Oliveira and Levkowitz (2003).

7.3.1 Non-Axis-Parallel Projection Methods

Example 7.3 PCA and MDS enable users to visualize a multidimensional dataset by projecting the dataset onto a 2D space. However, they can show only one aspect of the original dataset. What can we try to visualize many more aspects of the original dataset?

Grand Tour and Projection Pursuit Projection methods in category 1, non-axis-parallel projection methods, were mostly developed by statisticians to enable researchers to see many more low-dimensional projects. The idea of projection pursuit (Friedman and Tukey, 1974) is to find the most interesting low-dimensional projections to identify interesting features in a multidimensional dataset. An automatic projection method known as the grand tour (Asimov, 1985) is a method for viewing multidimensional data via orthogonal projection onto a sequence of 2D subspaces. It changes the viewing direction, generating a movielike animation that makes a complete search of the original space. However, it might take several hours to complete a reasonably complete visual search even in four dimensions (Huber, 1985). An exhaustive visual search is out of the question as the number of dimensions grows.

Friedman and Tukey (1974) devised a method to automate the task of projection pursuit. They defined interesting projections as ones deviating from the normal distribution and provided a numerical index to evaluate the interestingness of the projection. When an interesting projection is found, the features on the projection are extracted and projection pursuit is continued until there is no remaining feature found.

Example 7.4 When biomedical researchers want to visually explore their multidimensional datasets with the most flexibility, are there any visual analysis tools that implement “grand tour” and “projection pursuit”?

XGobi (Cook et al., 1995) or *GGobi* (Swayne et al., 2001) is a widely used open-source statistical visualization tool that implements both grand tour and projection pursuit. We can download GGobi at <http://www.ggobi.org/>, where there are also tutorials and demonstrations.

There are also several clustering methods that utilize a series of low-dimensional projections in category 1. Among them, the HD-Eye system by Hinneburg et al. (1999) implements an interactive divisive hierarchical clustering algorithm built on a partitioning clustering algorithm, or OptiGrid (Hinneburg and Keim, 1999). They show projections using glyphs, color, or curve-based density displays to users so that users can visually determine low-dimensional projections where well-separated clusters are and then define separators on the projections.

These automatic projection pursuit methods using a series of low-dimensional projections have made impressive gains in the problem of multidimensional data analysis, but they have limitations. One of the most important problems is the difficulty in interpreting the solutions from the automatic projection pursuit. Since the axes are the linear/nonlinear combination of the variables (or dimensions) of the original data, it is hard to determine what the projection actually means to users. Conversely, this is one of the reasons that axis-parallel projections (projection methods in category 2) are used in many multidimensional analysis tools (Guo, 2003; Ward, 1994).

7.3.2 Axis-Parallel Projection Methods

Projection methods in category 2, axis-parallel projection methods, have been used by researchers in machine learning, data mining, and information visualization. In this category, 2D projections (visualized in “scatterplots”) are made by choosing one dimension for the x -axis and another dimension for the y -axis.

Example 7.5 When there are a relatively small number of dimensions (e.g., < 10) in a multidimensional dataset, what kind of visualizations are useful to show the overall relationships between all possible pairs of dimensions?

A *scatterplot matrix* is a well-known technique to show all possible scatterplots of a multidimensional dataset with a relatively small number of dimensions. The statistical package R has a version of scatterplot matrix visualization. The following command will show a scatterplot matrix visualization for the multidimensional dataset in Examples 7.1 and 7.2 using only the first five dimensions. The scatterplot matrix in Figure 7.3 shows univariate displays down the diagonal and each nondiagonal cell shows a scatterplot for the two corresponding dimensions.

```
> scatterplot.matrix(data[c(1, 2, 3, 4, 5)])
```

In machine learning and data mining, ample research has also been conducted to address the problems of using projections. Most work focuses on the detection of dimensions that are most useful for a certain application, for example, supervised classification. In this area, the term *feature selection* is a process that chooses an optimal subset of features according to a certain criterion (Liu and Motoda, 1998), where a feature simply means a dimension. Basically, the goal is to find a good subset of dimensions (or features) that contribute to the construction of a good classifier.

Unsupervised feature selection methods are also studied in close relation with unsupervised clustering algorithms. In this case, the goal is to find an optimal subset of features with which clusters are well identified. In pattern recognition, researchers want to find a subset of dimensions with which they can better detect specific patterns in a dataset.

In subspace-based clustering analysis, researchers want to find projections where it is easy to naturally partition the dataset. There are clustering algorithms based on axis-parallel projections of the multidimensional data. CLIQUE (Agrawal et al., 1998) partitions low-dimensional subspaces into regular hyperrectangles. It finds all dense units in each k -dimensional subspace using the dense units in $(k - 1)$ -dimensional subspaces and then connects these axis-parallel dense units to build a “maximal” set of connected dense units which will be reported in disjunctive normal form. PROCLUS (Aggarwal et al., 1999) does not partition subdimensions but instead finds a set of k -medoids drawn from different clusters together with appropriate sets of dimensions for each medoid. Then it assigns the data items to the medoids through a single pass over the database.

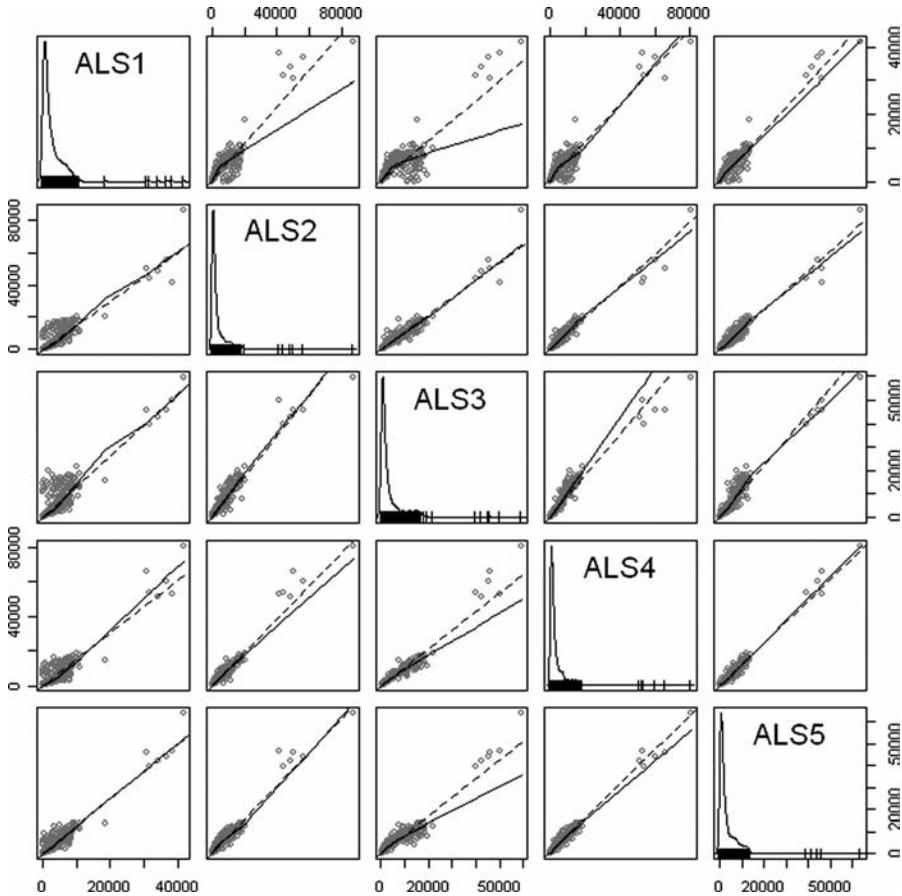


Figure 7.3 Scatterplot matrix visualization for first five dimensions of multidimensional dataset with 33 muscular dystrophy samples and 878 genes. (See color insert.)

In the early 1980s, Tukey, one of the prominent statisticians who foresaw the utility of computers in exploratory data analysis, envisioned a concept of “scagnostics” (a special case of “cognostics”—computer guiding diagnostics) (Tukey and Tukey, 1985). With high-dimensional data, it is necessary to use computers to evaluate the relative interest of different scatterplots or the relative importance of showing them and sort out such scatterplots for human analyses. He emphasized the need for better ideas on “what to compute” and “how” as well as “why.” He proposed several scagnostic indices such as the projection-pursuit clottedness and the difference between the classical correlation coefficient and robust correlation.

There are also some research tools and commercial products for helping users find more informative visualizations. Spotfire (trademark of TIBCO software) has a guidance tool called View Tip for rapid assessment of potentially interesting scatterplots, which shows an ordered list of all possible scatterplots from the one with highest correlation to the one with the lowest correlation.

Michael Friendly's Corrgram (Friendly, 2002) uses a color- and shape-coded scatterplot matrix display (Tukey and Tukey, 1981) to show correlations between variables. Variables are permuted so that correlated variables are positioned adjacently. Guo (2003) and Guo et al. (2003) also evaluated all possible axis-parallel 2D projections according to the maximum conditional entropy to identify ones that are most useful to find clusters. They visualized the entropy values in a matrix display called the entropy matrix (MacEachren et al., 2003) that is also a color-coded scatterplot matrix.

7.4 ISSUES AND CHALLENGES

Various visualization techniques for multidimensional datasets illustrated different perspectives that should be considered to facilitate visual understanding of the data. The difficulty in appreciating multiple dimensions has made researchers in different disciplines develop various methods to visualize multidimensional datasets. Although there are software tools for exploring and understanding multidimensional datasets (Cook et al., 1995; Ward, 1994), the utility of interactive interaction techniques has not been thoroughly explored.

Data mining and database research have suggested that clustering is a useful descriptive feature to reveal what the data look like and what the characteristics are. In this sense, the visualization of multidimensional data-clustering result has been an important area of multidimensional data visualization, where algorithmic work and visualization techniques can be combined to aid users to explore and understand the datasets. Among other clustering algorithms, the traditional hierarchical agglomerative algorithm is qualitatively effective (Eisen et al., 1998), and furthermore the visual representation of the clustering result (or dendrogram) is so intuitive and easy to understand that many researchers utilize it for understanding their datasets and presenting the result (Eisen et al., 1998; Seo and Shneiderman, 2002). Although Spotfire and some other tools provide tools for visualizing dendrograms, further work is necessary to incorporate interactive exploration methods into the understanding of the hierarchical clustering results.

Finding interesting axis-parallel 2D projections has been an important task for identifying useful features of the original multidimensional dataset. Most work for finding interesting 2D projections has focused on detecting 2D projections well suited for partitioning data. Most of them have one specific definition of what an "interesting" projection is. The definition of "interestingness" can be different from user to user or from application to application. For example, if users are interested in inferring why a group of items are clustered together in a hierarchical clustering, the most interesting projection would be the one that best separates the group from others. However, if users are seeking functional relationships between dimensions, the most interesting projection would be the one where all items are aligned on the diagonal.

Combining interactive tools with the powerful data-mining approaches, especially clustering analysis, is essential to help users effectively explore and understand multidimensional datasets, but at the same time it presents several challenges.

The design of the interactive interface for such tools should deal with the issues about how to naturally integrate dynamic interaction techniques into the exploration process and how to effectively provide sufficient contextual explanation about the analysis result (for example, in case of cluster analysis, why they are clustered together). Furthermore, it might be difficult to implement interactive visualization systems that practically combine the rapid, incremental updates of visualization with the computational requirements of data mining.

Although presentations in category 1, non-axis parallel, can show all possible 2D projections of a multidimensional dataset, they suffer from users' difficulty in interpreting 2D projections whose axes are linear/nonlinear combinations of two or more dimensions. For example, even though users may find a strong linear correlation on a projection where the horizontal axis is $3.7 \times \text{body weight} - 2.3 \times \text{height}$ and the vertical axis is $\text{waist size} + 2.6 \times \text{chest size}$, the finding is not so useful because it is difficult to understand the meaning of such projections.

Techniques in category 2, axis parallel, have a limitation that features can be detected in only the two selected dimensions. However, since it is familiar and comprehensible for users to interpret the meaning of the projection, these projections have been widely used and implemented in visualization tools. A problem with these category 2 presentations is how to deal with the large number of possible low-dimensional projections. If we have an m -dimensional dataset, we can generate $m(m - 1)/2$ two-dimensional projections using the category 2 techniques. We believe that the rank-by-feature framework offers an attractive solution to coping with the large numbers of low-dimensional projections and it provides practical assistance in finding features in multidimensional data.

Techniques in category 3 remain important. These techniques involve a unique method to map dimensions to some visual forms. However, users often have difficulty in grasping the meaning of the low-dimensional projections made by the techniques in category 3.

There have been many commercial packages and research projects that utilize low-dimensional projections for exploratory data analysis, including spreadsheets, statistical packages, and information visualization tools. However, users have to develop their own strategies to discover interesting projections and to display them. We believe that existing packages and projects, especially information visualization tools for exploratory data analysis, can be improved by enabling users to systematically examine low-dimensional projections.

7.5 SYSTEMATIC EXPLORATION OF LOW-DIMENSIONAL PROJECTIONS

A playful analogy may help clarify our goals (Seo and Shneiderman, 2004). Imagine you are dropped by parachute into an unfamiliar place—it could be a forest, prairie, or mountainous area. You could set out in a random direction to see what is nearby and then decide where to turn next. Or you might go toward peaks or valleys. You might

notice interesting rocks, turbulent streams, scented flowers, tall trees, attractive ferns, colorful birds, graceful impalas, and so on. Wandering around might be greatly satisfying if you had no specific goals, but if you needed to survey the land to find your way to safety, catalog the plants to locate candidate pharmaceuticals, or develop a wildlife management strategy, you would need to be more systematic. Of course, each profession that deals with the multifaceted richness of natural landscapes has developed orderly strategies to guide novices, to ensure thorough analyses, to promote comprehensive and consistent reporting, and to facilitate cooperation among professionals.

Our principles for exploratory analysis of multidimensional datasets have similar goals. Instead of wandering, analysts should clarify their goals and use appropriate techniques to ensure a comprehensive analysis. A good starting point is the set of principles put forth by Moore and McCabe, who recommended that statistical tools should (1) enable users to examine each dimension first and then explore relationships among dimensions and (2) offer graphical displays first and then provide numerical summaries (Moore and McCabe, 1999). We extend Moore and McCabe's principles to include ranking the projections to guide discovery of desired features and realize this idea with overviews to see the range of possibilities and coordination to see multiple presentations. An orderly process of exploration is vital, even though there will inevitably be excursions, iterations, and shifts of attention from details to overviews and back.

The rank-by-feature framework is especially potent for interactive feature detection in multidimensional data. To promote comprehensibility, we concentrate on axis-parallel projections; however, the rank-by-feature framework can be used with general geometric projections. Although 3D projections are sometimes useful to reveal hidden features, they suffer from occlusion and the disorientation brought on by the cognitive burden of navigation. On the other hand, 2D projections are widely understood by users, allowing them to concentrate on the data itself rather than being distracted by navigation controls.

Detecting interesting features in low dimensions (1D or 2D) by utilizing powerful human perceptual abilities is crucial to understand the original multidimensional dataset. Familiar graphical displays such as histograms, scatterplots, and other well-known 2D plots are effective to reveal features including basic summary statistics and even unexpected features in the dataset. There are also many algorithmic or statistical techniques that are especially effective in low-dimensional spaces. While there have been many approaches utilizing such visual displays and low-dimensional techniques, most of them lack a systematic framework that organizes such functionalities to help analysts in their feature detection tasks.

Our graphics, ranking, and interaction for discovery (GRID) principles are designed to enable users to better understand distributions in 1D or 2D and then discover relationships, clusters, gaps, outliers, and other features. Users work by viewing graphical presentations (histograms, boxplots, and scatterplots) and then choose a feature detection criterion to rank 1D or 2D axis-parallel projections. By combining information visualization techniques (overview, coordination, and dynamic query) with ranking, summaries and statistical methods users can systematically examine

the most important 1D and 2D axis-parallel projections. GRID principles are summarized as follows:

1. Study 1D, study 2D, then find features.
2. Ranking guides insight, statistics confirm.

Abiding by these principles, the rank-by-feature framework has an interface for 1D projections and a separate one for 2D projections. Users can begin their exploration with the main graphical display—histograms for 1D and scatterplots for 2D—and they can also study numerical summaries for more details.

The rank-by-feature framework helps users systematically examine low-dimensional (1D or 2D) projections to maximize the benefit of exploratory tools. In this framework, users can select an interesting ranking criterion. Users can rank low-dimensional projections (1D or 2D) of the multidimensional dataset according to the strength of the selected feature in the projection. When there are many dimensions, the number of possible projections is too large to investigate by looking for interesting features. The rank-by-feature framework relieves users from such burdens by recommending projections to users in an ordered manner defined by a ranking criterion that users selected. This framework has been implemented in our interactive visualization tool, HCE 3.0 (www.cs.umd.edu/hcil/hce/).

We present a systematic exploration strategy for large multidimensional data exploration using low-dimensional projections. To implement the strategy, we also present a conceptual interface framework for interactive feature detection named *rank-by-feature framework* to facilitate exploring a large number of low-dimensional projects for multidimensional datasets. We believe the systematic exploration strategy and the interface framework can help biomedical researchers analyze their large multidimensional biomedical datasets through intuitive user interfaces and efficient visual encodings.

In the rank-by-feature framework (the rank-by-feature framework interface for 2D scatterplots is shown at the bottom half of Fig. 7.4), users can select an interesting ranking criterion, and then all possible axis-parallel projections of a multidimensional dataset are ranked by the selected ranking criterion. Available ranking criteria are explained in following sections. The ranking result is visually presented in a color-coded grid (“score overview”) as well as a tabular display (“ordered list”) where each row represents a projection and is color coded by the ranking score. With these presentations users can easily perceive the most interesting projections and also grasp the overall ranking score distribution. Users can manually browse projections by rapidly changing the dimension for an axis using the item slider attached to the corresponding axis of the projection view (histogram and boxplot for 1D and scatterplot for 2D).

For example, assume that users analyze the U.S. counties data set with 17 demographic, health, and economic statistics available for each county. The dataset can be thought of as a 17-dimensional dataset. Users can choose “Pearson correlation coefficient” as a ranking criterion at the rank-by-feature framework if they are interested in linear relationships between dimensions. Then, the rank-by-feature framework calculates “scores” (in this case, Pearson correlation coefficient) for all

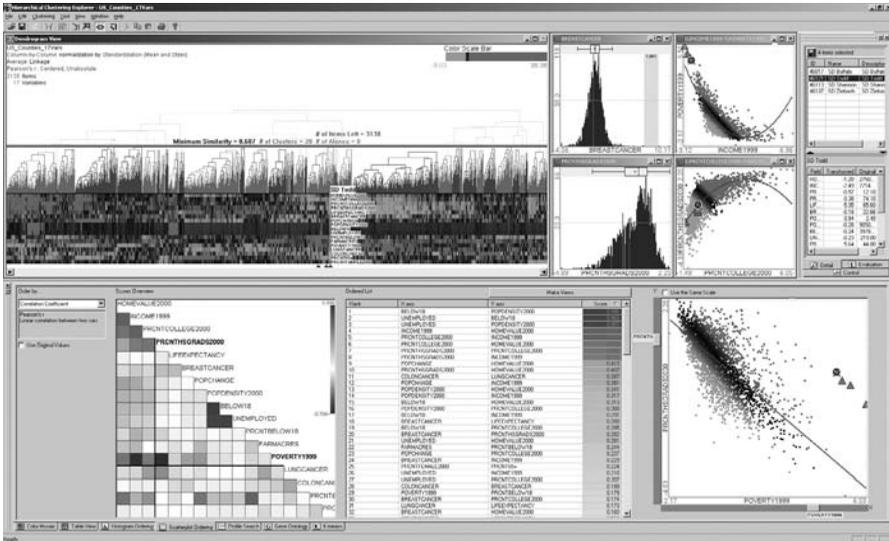


Figure 7.4 Rank-by-feature framework in HCE. (See color insert.)

possible pairs of dimensions and ranks all pairs according to their scores. Users could easily identify that there is a negative correlation between poverty level and percentage of high school graduates after they skim through the score overview (a color-coded grid display at the lower left corner of Fig. 7.4), where each cell represents the scatterplot for a pair of dimensions and it is color coded by the score value for the scatterplot. All possible pairs are also shown in the ordered list (a list control right next to the score overview at Fig. 7.4) together with the numerical score values in a column. The scatterplot is shown at the lower right corner of Figure 7.4. More details on the rank-by-feature framework are explained in the following sections.

The rank-by-feature framework was implemented in our interactive exploration tool for multidimensional data, HCE (Seo and Shneiderman, 2002) (Fig. 7.4), as two new tab windows (Histogram Ordering for 1D projections, and Scatterplot Ordering for 2D projections). The interactively coordinated displays in HCE 3.0 include dendrogram view, histogram views, scatterplot views, detailed view at the top right corner, and seven tabs (Color Mosaic, Table View, Histogram Ordering, Scatterplot Ordering, Profile Search, Gene Ontology, and K-means) at the bottom (Scatterplot Ordering tab is selected in Fig. 7.4), as shown in Figure 7.4. The dendrogram view at the top-left corner visualizes the hierarchical clustering result of a U.S. counties statistics dataset enabling users to interactively explore the clustering result. Among the seven tabs, Histogram Ordering and Scatterplot Ordering implement the rank-by-feature framework interface for 1D and 2D, respectively. In Figure 7.4, two histograms and two scatterplots are selected through the rank-by-feature interfaces and are shown as separate child windows to the right of the dendrogram view. Four selected U.S. counties are listed in the top half of the details view

and the statistics for one of the counties are shown in the bottom half. A 2D scatterplot ordering result using “Pearson correlation coefficient” as the ranking criterion is shown in the Scatterplot Ordering tab. Four counties that are poor and have a medium number of high school graduates are selected in the scatterplot browser and they are all highlighted in other views with triangles. By using the rank-by-feature framework, users can easily find interesting histograms and scatterplots and generate separate windows to visualize those plots. All these plots are interactively coordinated with other views (e.g., dendrogram and color mosaic views, tabular view, parallel coordinate view) in HCE 3.0. If users select a group of items in any view, they can see the selected items highlighted in all other views. Thus, it is possible to comprehend the data from various perspectives to get more meaningful insights.

The rank-by-feature framework was shown effective in facilitating users’ exploration of large multidimensional datasets from several different research fields including microarray data analysis (Seo and Shneiderman, 2006). We believe that it can be successfully applied to biomedical datasets that are very often large multidimensional datasets. In the following sections, we introduce visual interface frameworks and ranking criteria for 1D and 2D projections for multidimensional datasets.

7.6 ONE-DIMENSIONAL HISTOGRAM ORDERING

Users begin the exploratory analysis of a multidimensional dataset by scrutinizing each dimension (or variable) one by one. Just looking at the distribution of values of a dimension gives them useful insight into the dimension. The most familiar graphical display tools for 1D data are histograms and boxplots. Histograms graphically reveal the scale and skewness of the data and the number of modes, gaps, and outliers in the data. Boxplots are also excellent tools for understanding the distribution within a dimension. They graphically show the five-number summary (the minimum, the first quartile, the median, the third quartile, and the maximum). These numbers provide users with an informative summary of a dimension’s center and spread, and they are the foundation of multidimensional data analysis for deriving a model for the data or for selecting dimensions for effective visualization.

7.6.1 Graphical User Interface

The main display for the rank-by-feature framework for 1D projections shows a combined histogram and boxplot (Fig. 7.5). The interface consists of four coordinated parts: control panel, score overview, ordered list, and histogram browser. Users can select a ranking criterion from a combo box in the control panel, and then they see the overview of scores for all dimensions in the score overview according to the selected ranking criterion. All dimensions are aligned from top to bottom in the original order, and each dimension is color coded by the score value. By default, cells of high value have bright blue green colors and cells of low value have bright brown colors. The cell of middle value has the white color. As a value gets closer to the middle value, the color intensity attenuates. Users can change the colors for minimum, middle, and maximum values. The color scale and mapping are shown at the top-right corner of the overview in Figure 7.5*b*.

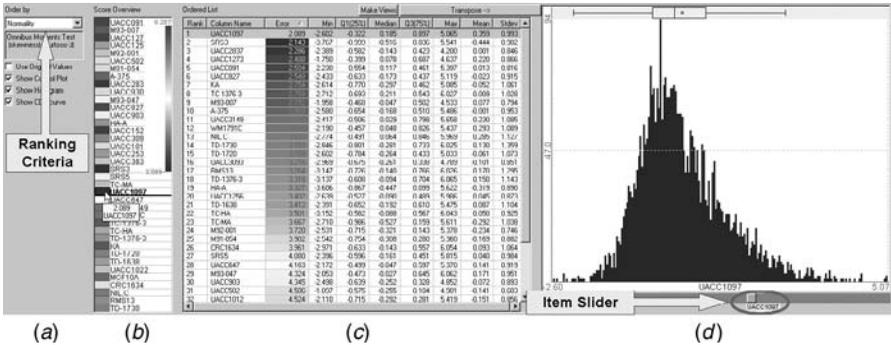


Figure 7.5 Rank-by-feature framework for 1D projections: histogram ordering. (See color insert.)

Users can easily see the overall pattern of the score distribution, and more importantly they can preattentively identify the dimension of the highest/lowest score in this overview. Once they identify an interesting row on the score overview, they can just mouse over the row to view the numerical score value and the name of the dimension is shown in a tooltip window (Fig. 7.5). The mouseover event is also instantaneously relayed to the ordered list and the histogram browser, so that the corresponding list item is highlighted in the ordered list and the corresponding histogram and boxplot are shown in the histogram browser. The score overview, the ordered list, and the histogram browser are interactively coordinated according to the change of the dimension in focus. In other words, a change of dimension in focus in one of the three components leads to the instantaneous change of dimension in focus in the other two components.

In the ordered list, users can see the numerical detail about the distribution of each dimension in an orderly manner. The numerical detail includes the five-number summary of each dimension and the mean and the standard deviation. The numerical score values are also shown at the third column whose background is color coded using the same color mapping as in the score overview.

While numerical summaries of distributions are very useful, sometimes they are misleading. For example, when there are two peaks in a distribution, neither the median nor the mean explains the center of the distribution. This is one of the cases for which a graphical representation of a distribution (e.g., a histogram) works better. In the histogram browser, users can see the visual representation of the distribution of a dimension at a time. A boxplot is a good graphical representation of the five-number summary, which together with a histogram provides an informative visual description of a dimension's distribution. It is possible to interactively change the dimension in focus just by dragging the item slider attached to the bottom of the histogram.

7.6.2 Ordering Criteria for Histograms

Since different users may be interested in different features in the datasets, it is desirable to allow users to customize the available set of ranking criteria. However, we have

chosen the following ranking criteria that we think fundamental and common for histograms as a starting point, and we have implemented them in HCE 3.0.

7.6.2.1 Normality of Distribution (0 to inf) Many statistical analysis methods such as t -test and analysis of variance (ANOVA) are based on the assumption that the dataset is sampled from a Gaussian normal distribution. Therefore, it is useful to know the normality of the dataset. Since a distribution can be nonnormal due to many different reasons, there are at least 10 statistical tests for normality including the Shapiro–Wilk and Kolmogorov–Smirnov tests. The omnibus moment test for normality was used in the current implementation. The test returns two values, skewness (s) and kurtosis (k). Since $s = 0$ and $k = 3$ for a standard normal distribution, $|s| + |k - 3|$ is calculated to measure how the distribution deviates from the normal distribution and ranks variables according to the measure. Users can confirm the ranking result using the histogram browser to gain an understanding of how the distribution shape deviates from the familiar bell-shaped normal curve.

7.6.2.2 Uniformity of Distribution (0 to Number of Bins) For the uniformity test, an information-based measure called entropy was used. Given k bins in a histogram, the entropy of a histogram, H , is defined as

$$\text{Entropy}(H) = - \sum_{i=1}^k p_i \log_2 p_i$$

where p_i is the probability that an item belongs to the i th bin.

High entropy means that values of the dimension are from a uniform distribution and the histogram for the dimension tends to be flat. While knowing a distribution is uniform is helpful to understand the dataset, it is sometimes more informative to know how far a distribution deviates from uniform distribution since a biased distribution sometimes reveals interesting outliers.

7.6.2.3 Number of Potential Outliers (0 to n) To count outliers in a distribution, we used the $1.5 \times \text{IQR}$ [interquartile range: the difference between the first quartile (Q_1) and the third quartile (Q_3)] criterion that is the basis of a rule of thumb in statistics for identifying suspected outliers (Moore and McCabe, 1999). An item of value d is considered as a suspected (mild) outlier if

$$d > (Q_3 + 1.5 \times \text{IQR}) \quad \text{or} \quad d < (Q_1 - 1.5 \times \text{IQR})$$

To get more restricted outliers (or extreme outliers), the $3 \times \text{IQR}$ range can be used. It is also possible to use an outlier detection algorithm developed in the data mining. Outliers are one of the most important features not only as noisy signals to be filtered but also as a truly unusual response to a medical treatment worth further investigation or as an indicator of credit card fraud.

7.6.2.4 Number of Unique Values (0 to n) At the beginning of the data analysis, it is useful to know how many unique values are in the data. Only a small number of unique values in a large set may indicate problems in sampling or data collection or transcription. Sometimes it may also indicate that the data are categorical values or the data were quantized. Special treatment may be necessary to deal with categorical or quantized variables.

7.6.2.5 Size of Biggest Gap (0 to Maximum Range of Dimensions) Gap is an important feature that can reveal separation of data items and modality of the distribution. Let t be a tolerance value, n be the number of bins, and f_{\max} be the maximum frequency. We define a gap as a set of contiguous bins b_k , where b_k

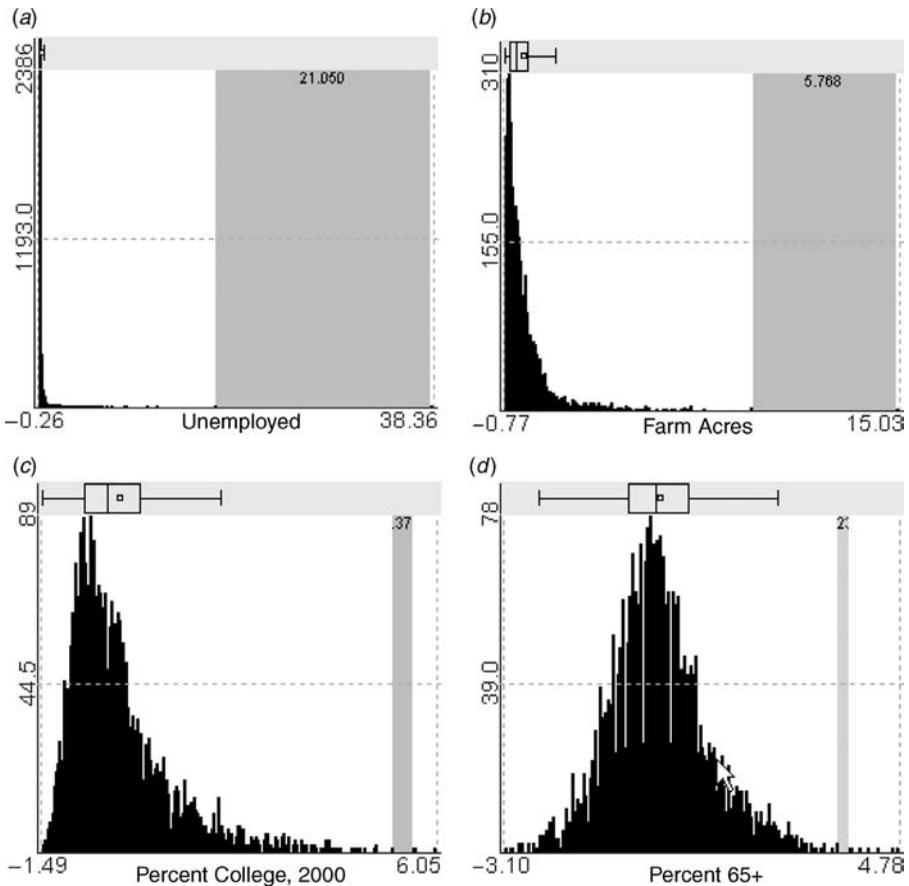


Figure 7.6 Four selected histograms ranging from big gap (a) to small gap (d). Gap detection was performed after standardizing each variable. The biggest gap is highlighted as a rectangle on each histogram. The bar to the right of the gap on (a) is for LA and the bar to the right of the gap on (b) is for Coconino, AZ.

($k = 0, \dots, n$) has less than tf_{\max} items. The procedure sequentially visits each bin and merges the satisfying bins to form a bigger set of such bins. It is a simple and fast procedure. Among all gaps in the data, histograms are ranked by the biggest gap size in each histogram (see Fig. 7.6). Since equal-sized bins are used, the biggest gap contains the most bins satisfying the tolerance value t .

For some of the ranking criteria for histogram ordering such as normality, there are many available statistical tests to choose from. We envision that many researchers could contribute statistical tests that could be easily incorporated into the rank-by-feature framework as plug-ins. For example, since outlier detection is a rich research area, novel statistical tests or new data-mining algorithms are likely to be proposed in the coming years, and they could be made available as plug-ins.

7.7 TWO-DIMENSIONAL SCATTERPLOT ORDERING

According to our fundamental principles for improving exploration of multidimensional data, after scrutinizing 1D projections, it is natural to move to 2D projections where pairwise relationships will be identified. Relationships between two dimensions (or variables) are conveniently visualized in a scatterplot. The values of one dimension are aligned on the horizontal axis, and the values of the other dimension are aligned on the vertical axis. Each data item in the dataset is shown as a point in the scatterplot whose position is determined by the values at the two dimensions. A scatterplot graphically reveals the form (e.g., linear or curved), direction (e.g., positive or negative), and strength (e.g., weak or strong) of relationships between two dimensions. It is also easy to identify outlying items in a scatterplot, but it can suffer from overplotting in which many items are densely packed in one area, making it difficult to gauge the density.

Scatterplots are used as the main display for the rank-by-feature framework interface for 2D projections. Figure 7.7 shows the interactive interface design for the rank-by-feature framework for 2D projections. Analogous to the interface for 1D projections, the interface consists of four coordinated components: control panel, score overview, ordered list, and scatterplot browser. Users select an ordering criterion in the control panel on the left, and then they see the complete ordering of all possible 2D projections according to the selected ordering criterion (Fig. 7.7a). The ordered list shows the result of ordering sorted by the ranking (or scores) with scores color coded on the background. Users can click on any column header to sort the list by the column. Users can easily find scatterplots of the highest/lowest score by changing the sort order to ascending or descending order of score (or rank). It is also easy to examine the scores of all scatterplots with a certain variable for the horizontal or vertical axis after sorting the list according to X or Y column by clicking the corresponding column header.

Users cannot, however, see the overview of entire relationships between variables at a glance in the ordered list. Overviews are important because they can show the whole distribution and reveal interesting parts of data. We have implemented a new version of the score overview for 2D projections. It is an m -by- m grid view where all dimensions are aligned in the rows and columns. Each cell of the score overview represents a scatterplot whose horizontal and vertical axes are dimensions

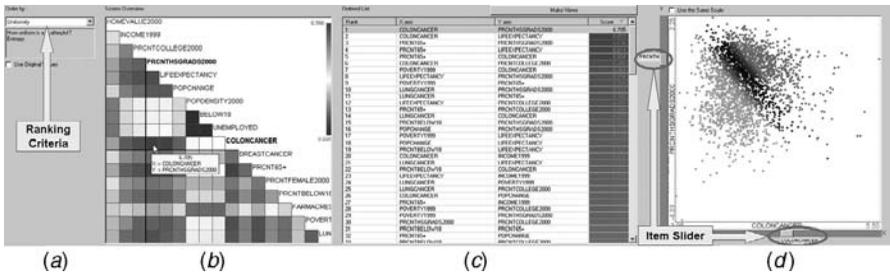


Figure 7.7 Rank-by-feature framework interface for scatterplots (2D): Scatterplot Ordering. All 2D scatterplots are ordered according to the current ordering criterion (a) in the ordered list (c). Users can select multiple scatterplots at the same time and generate separate scatterplot windows to compare them in a screen. The score overview (b) shows an overview of scores of all scatterplots. A mouseover event activates a cell in the score overview, highlights the corresponding item in the ordered list (c), and shows the corresponding scatterplot in the scatterplot browser (d) simultaneously. A click on a cell at the score overview selects the cell and the selection is fixed until another click event occurs in the score overview or another selection event occurs in other views. A selected scatterplot is shown in the scatterplot browser (d), where it is also easy to traverse scatterplot space by changing the x - or y -axis using item sliders on the horizontal or vertical axis. (The dataset shown is demographic- and health-related statistics for 3138 U.S. counties with 17 attributes.) (See color insert.)

at the corresponding column and row, respectively. Since this table is symmetric, we used only the lower triangular part for showing scores and the diagonal cells for showing the dimension names, as shown in Figure 7.7b. Each cell is color coded by its score value using the same mapping scheme as in 1D ordering. As users move the mouse over a cell, the scatterplot corresponding to the cell is shown in the scatterplot browser simultaneously, and the corresponding item is highlighted in the ordered list (Fig. 7.7c). The score overview, ordered list, and scatterplot browser are interactively coordinated according to the change of the dimension in focus as in the 1D interface.

In the score overview, users can preattentively detect the highest/lowest scored combinations of dimensions due to the linear color-coding scheme and the intuitive grid display. Sometimes, users can also easily find a dimension that is the least or most correlated to most other dimensions by just locating a whole row or column where all cells are the mostly bright brown or bright blue green. It is also possible to find an outlying scatterplot whose cell has distinctive color intensity compared to the rest of the same row or column. After locating an interesting cell, users can click on the cell to select, and then they can scrutinize it on the scatterplot browser and on other tightly coordinated views in HCE 3.0.

While the ordered list shows the numerical score values of relationships between two dimensions, the interactive scatterplot browser best displays the relationship graphically. In the scatterplot browser, users can quickly take a look at scatterplots by using item sliders attached to the scatterplot view. Simply by dragging the vertical or horizontal item slider bar, users can change the dimension for the horizontal or vertical axis. With the current version implemented in HCE 3.0, users can investigate multiple scatterplots at the same time. They can select several scatterplots in the

ordered list by clicking on them with the control key pressed. Then, clicking on the Make Views button on the top of the ordered list displays each selected scatterplot in a separate child window. Users can select a group of items by dragging a rubber rectangle over a scatterplot, and the items within the rubber rectangle are highlighted in all other views. On some scatterplots they might gather tightly together, while on other scatterplots they scatter around.

7.7.1 Ordering Criteria for Scatterplots

Again interesting ranking criteria might be different from user to user or from application to application. We have chosen the following six ranking criteria that we think are fundamental and common for scatterplots, and we have implemented them in HCE. The first three criteria are useful to reveal statistical (linear or quadratic) relationships between two dimensions (or variables), and the next three are useful to find scatterplots of interesting distributions.

7.7.1.1 Correlation Coefficient (-1 to 1) For the first criterion, we use Pearson's correlation coefficient (r) for a scatterplot (S) with n points defined as

$$r(S) = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^n (x_i - \bar{x})^2 \sum_{i=1}^n (y_i - \bar{y})^2}}$$

Pearson's r is a number between -1 and 1 . The sign tells us the direction of the relationship and the magnitude tells us the strength of the linear relationship. The magnitude of r increases as the points lie closer to the straight line. Linear relationships are particularly important because straight-line patterns are common and simple to understand. Even though a strong correlation between two variables does not always mean that one variable causes the other, it can provide a good clue to a true cause, which could be another variable. Moreover, dimensionality can be reduced by combining two strongly correlated dimensions, and visualization can be improved by juxtaposing correlated dimensions. As a visual representation of the linear relationship between two variables, the line of best fit or the regression line is drawn over scatterplots (Fig. 7.8).

7.7.1.2 Least-Square Error for Curvilinear Regression (0 to 1) This criterion is to sort scatterplots in terms of least-square errors from the optimal quadratic curve fit so that users can easily isolate ones where all points are closely/loosely arranged along a quadratic curve. Users are often interested to find nonlinear relationships in the dataset in addition to linear relationships. For example, economists might expect that there is a negative linear relationship between county income and poverty, which is easily confirmed by correlation ranking. However, they might be intrigued to discover that there is a quadratic relationship between the two, which can be easily revealed using this criterion.

7.7.1.3 Quadracity (0 to inf) If two variables show a strong linear relationship, they also produce a small error for curvilinear regression because the linear

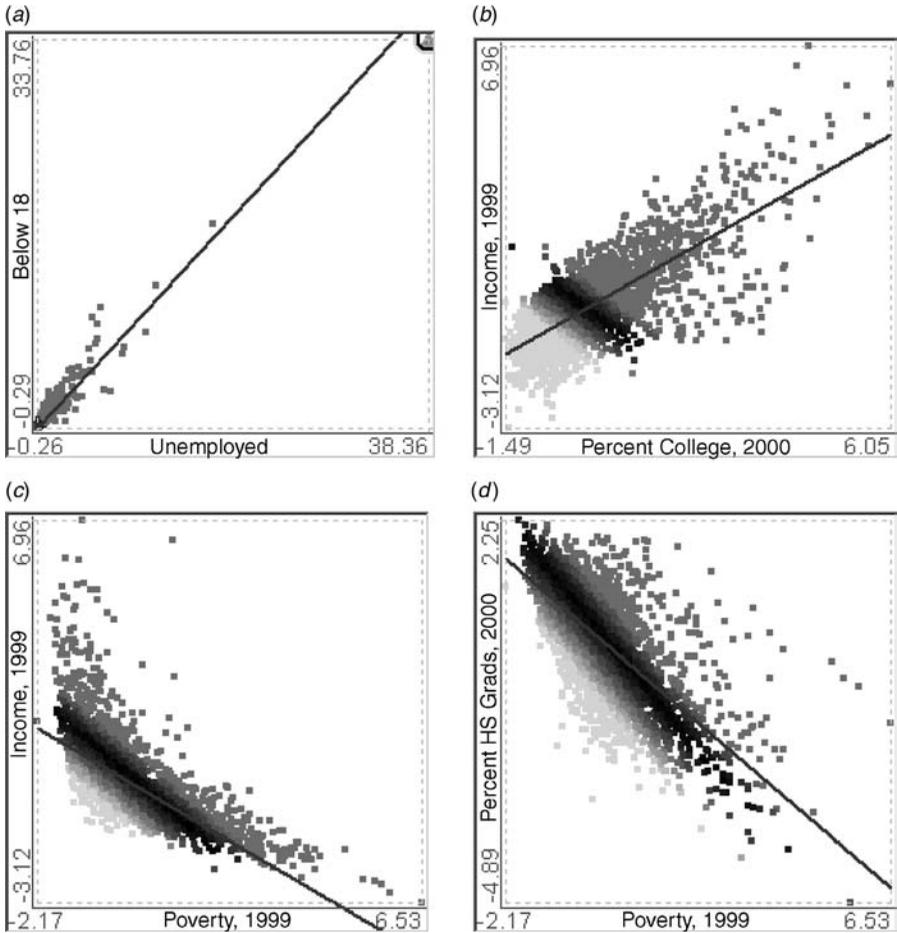


Figure 7.8 Four selected scatterplots ordered by correlation coefficient. The line of best fit is drawn as a blue line. (See color insert.)

relationship is a special case of the quadratic relationship, where the coefficient of the highest degree term (x^2) is zero. To emphasize the real quadratic relationships, we add the “quadracity” criterion. It ranks scatterplots according to the coefficient of the highest degree term, so that users can easily identify ones that are more quadratic than others. Of course, the least-square error criterion should be considered to find more meaningful quadratic relationships, but users can easily see the error by viewing the fitting curve and points at the scatterplot browser (Fig. 7.9).

7.7.1.4 Number of Potential Outliers (0 to n) Even though there is a simple statistical rule of thumb for identifying suspected outliers in 1D, there is no simple counterpart for 2D cases. Instead, there are many outlier detection algorithms developed by data-mining and database researchers. Among them, distance-based outlier

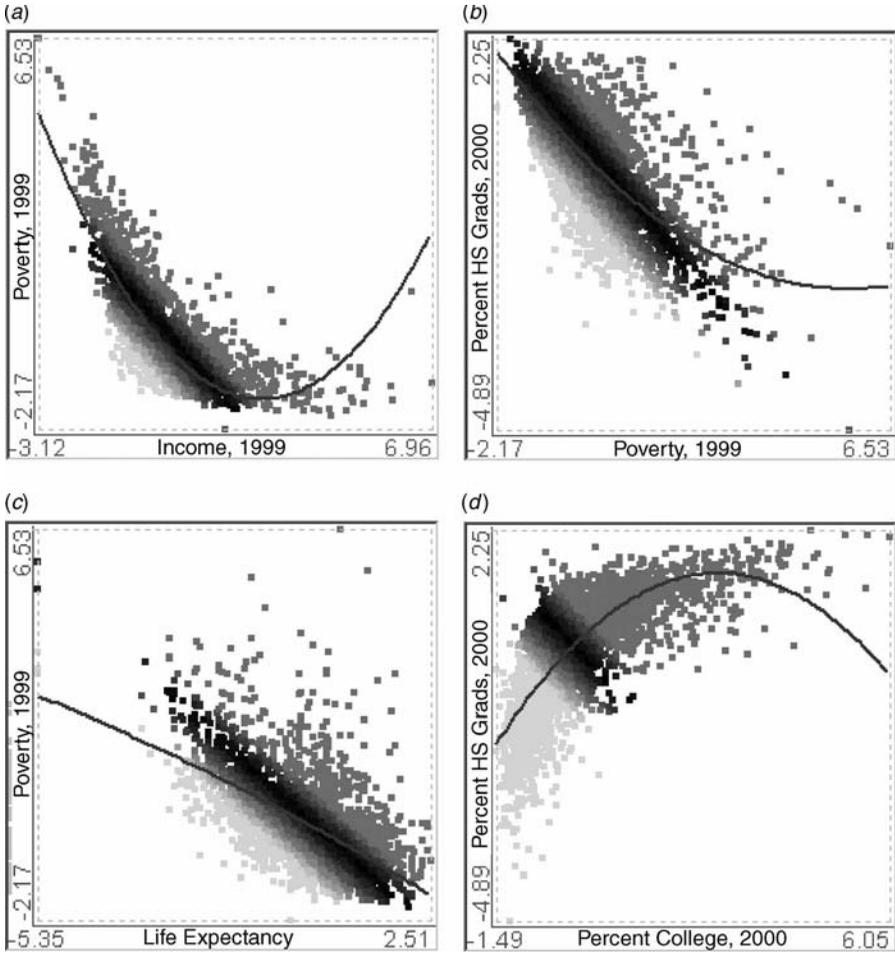


Figure 7.9 Quadraticity (coefficient of x_2 term). The regression curve is drawn as a blue parabola. (See color insert.)

detection methods such as DB-out (Knorr et al., 2000) define an object as an outlier if at least a fraction p of the objects in the dataset are apart from the object more than at a distance greater than a threshold value. Density-based outlier detection methods such as local outlier factor (LOF)-based method (Breunig et al., 2000) define an object as an outlier if the relative density in the local neighborhood of the object is less than a threshold, in other words the LOF of the object is greater than a threshold. Since the LOF-based method is more flexible and dynamic in terms of outlier definition and detection, we included the LOF-based method in the current implementation (Fig. 7.10).

7.7.1.5 Number of Items in Region of Interest (0 to n) This criterion is the most interactive since it requires users to specify a (rectangular, elliptical, or

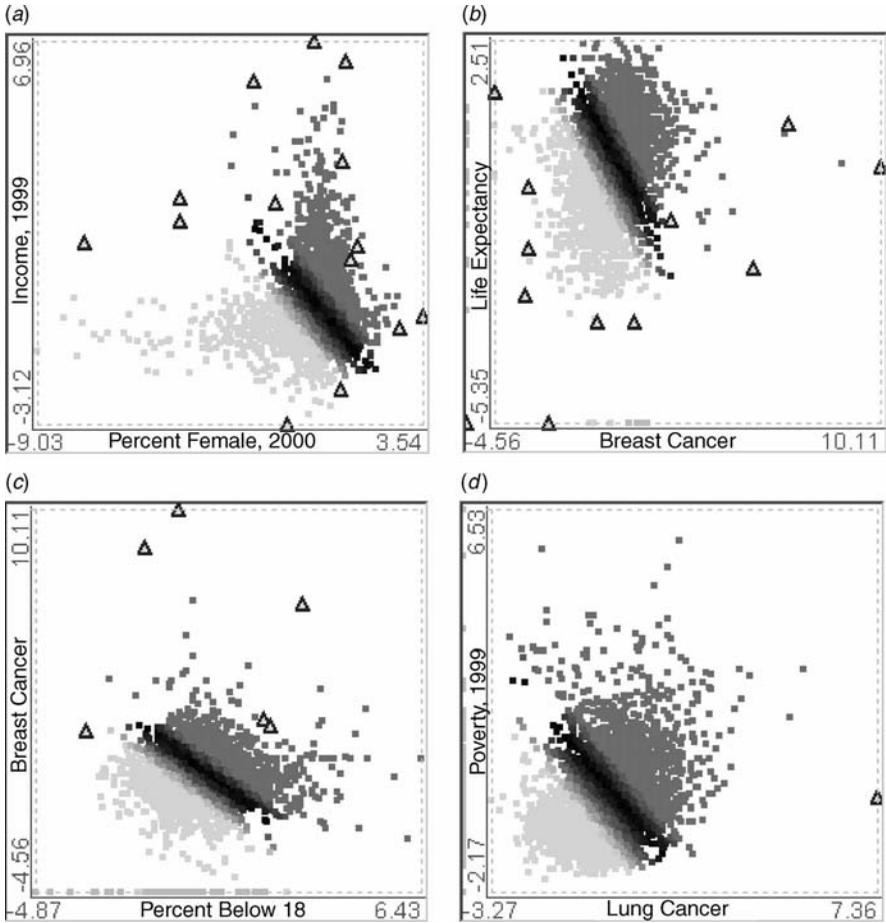


Figure 7.10 Number of outliers. Outliers whose LOF is greater than $(\text{minimum LOF} + \text{maximum LOF})/2$ are highlighted as triangles. (See color insert.)

free-formed) region of interest by dragging the mouse with the left button depressed on the scatterplot browser. Then the algorithm uses the number of items in the region to order all scatterplots so that users can easily find ones with largest/smallest number of items in the given 2D region. An interesting application of this ranking criterion is when users specify an upper left or lower right corner of the scatterplot. Users can easily identify scatterplots where most/least items have a low value for one variable (e.g., salary of a baseball player) and a high value for the other variable (e.g., the batting average). In this way, users can use this ranking criterion to learn properties of associations between variables.

7.7.1.6 Uniformity of Scatterplots (0 to Number of Cells) For this criterion, we calculate the entropy in the same way as we did for histograms, but this time we divide the 2D space into regular grid cells and then use each cell as a bin. For example,

if we have generated a k -by- k grid, the entropy of a scatterplot, S , is

$$\text{Entropy}(S) = - \sum_{i=1}^k \sum_{j=1}^k p_{ij} \log_2 p_{ij}$$

where p_{ij} is the probability that an item belongs to the cell at (i, j) of the grid. Since the more scattered a scatterplot is, the greater the entropy is, scatterplots of high entropy are ranked high according to this ranking criterion.

7.7.2 Transformations and Potential Ranking Criteria

Users sometimes want to transform the variable to get a better result. For example, log transformations convert exponential relationships to linear relationships, straighten skewed distributions, and reduce the variance. If variables have differing ranges, then comparisons must be done carefully to prevent misleading results; for example, a gap in a variable whose range is 0–1000 is not usually comparable to a gap in a variable whose range is 2–6. Therefore transformations such as standardization to common scales are helpful to ensure that ranking results are useful. In the current rank-by-feature framework, users can perform five transformations (natural log, standardization, normalization to the first column or to the median, and linear scaling to a certain range) over each column or row of the dataset when loading the dataset. Then when they use the rank-by-feature framework, the transformation results will apply to the transformed values. An improvement to the rank-by-feature framework would allow users to apply transformations during their analyses, not only at the data-loading time. More transformations, such as polynomial or sinusoidal functions, would also be useful.

We have implemented only a small fraction of possible ranking criteria in the current implementation. Among the many useful ranking criteria, we suggest three interesting and potent ones.

7.7.2.1 Modality If a distribution is normal, there should be one peak in a histogram. But sometimes there are several peaks. In those cases, different analysis methods (such as sinusoidal fitting) should be applied to the variable, or the variable should be partitioned to separate each peak (bell-shaped curve). In this sense, the modality is also an important feature. One possible score for the detection of multimodality could be the change of sign of the first derivative of the histogram curve. If there is one peak, there should be no change at the sign of the first derivative. If there are two peaks, the sign should change once.

7.7.2.2 Outlierness The number of outliers can be one of the informative features that contribute to making a better sense of underlying datasets. However, sometimes “outlierness,” the strength of the outliers in a projection, is a more informative feature than the number of outliers. The strongest outlier by itself can be a very important signal to users, and at the same time the axes of the projection where the outlier turns out to be a strong outlier can also be informative features because variables for

those axes can give an explanation of the outlier's strength. One possible score for the outlierness could be the maximum value of the LOF on a projection.

7.7.2.3 Gaps in 2D As we already saw in the 1D ordering cases, gaps are an informative feature in the dataset. Several researchers in other fields also have studied related problems such as the largest empty-rectangle problem (Chazelle et al., 1986; Edmonds et al., 2003) and the hole detection (Liu et al., 1997). The largest empty-rectangle problem is defined as follows: Given a 2D rectangular space and points inside it, find the largest axis-parallel subrectangle that lies within the rectangle and contains no points inside it. The hole detection problem is to find informative empty regions in a multidimensional space. The time complexity of the current implementations prevents exploratory data analysis. A more rapid algorithm could use the grid-based approach that was effective in the uniformity criteria. The projection plane can be divided into a relatively small number of grid cells (say 100 by 100), so that it becomes easy to find the biggest gap, similar to the method used for ranking 1D histogram gaps.

In addition to these ranking functions, we can also consider using various functions available in external statistical software packages such as R, Excel, SAS, and so on. In that case, the first method can be used, or it is possible to implement a wrapper function that has the prototype as in the second method and performs a filtering operation.

7.8 CONCLUSION

In this chapter, we presented famous traditional visualization techniques for multidimensional datasets with examples and visualization tools. To complement such traditional multidimensional visualization techniques, we also suggested that axis-parallel projects could play an essential role in helping users intuitively make meanings out of low-dimensional projects. The take-away message from the natural landscape analogy given early in this chapter is that guiding principles can produce an orderly and comprehensive strategy with clear goals. Even when researchers are doing exploratory data analysis, they are more likely to make valuable insights if they have some notion of what they are looking for.

We believe that the proposed strategy for multidimensional data exploration with room for iteration and rapid shifts of attention enables novices and experts to make discoveries more reliably. The graphics, ranking and interaction for discovery (GRID) principles are as follows:

1. Study 1D, study 2D, then find features.
2. Rank guides insight and confirm statistics.

The rank-by-feature framework enables users to apply a systematic approach to understanding the dimensions and finding important features using axis-parallel 1D and 2D projections of multidimensional datasets. Users begin by selecting a ranking criterion and then can see the ranking for all 1D or 2D projections. They can select

high- or low-ranked projections and view them rapidly or sweep through a group of projections in an orderly manner. The score overview provides a visual summary that helps users identify extreme values of criteria such as correlation coefficients or uniformity measures. Information visualization principles and techniques such as dynamic query-by-item sliders combined with traditional graphical displays such as histograms, boxplots, and scatterplots play a major role in the rank-by-feature framework.

As future work, various statistical tools and data-mining algorithms, including ones presented in the previous section, can be incorporated into our rank-by-feature framework as new ranking criteria. Just as geologists, naturalists, and botanists depend on many kinds of maps, compasses, binoculars, or global positioning systems, dozens of criteria seem useful in our projects. It seems likely that specialized criteria will be developed by experts in knowledge domains such as genomics, demographics, and finance. Other directions for future work include extending the rank-by-feature framework to accommodate 3D projections.

The concepts in the rank-by-feature framework and the current user interface might be difficult for many data analysts to master. However, our experience with a dozen biologists in gene expression data analysis tasks is giving us a better understanding of what training methods to use. Of particular importance is the development of meaningful examples based on comprehensible datasets that demonstrate the power of each ranking criterion. Since screen space is a scarce resource in these information-abundant interfaces, higher resolution displays (we use 3800×2480 -pixel display whenever possible) or multiple displays are helpful, as are efficient screen management strategies.

We hope the potent concepts in the rank-by-feature framework will be implemented by others with varied interfaces for spreadsheets, statistical packages, or biomedical information visualization tools. We believe that the GRID principles and the rank-by-feature framework will effectively guide biomedical researchers to understand dimensions, identify relationships, and discover interesting features.

REFERENCES

- Aggarwal, C. C., Wolf, J. L., Yu, P. S., Procopiuc, C., and Park, J. S. (1999). Fast algorithms for projected clustering. In *Proceedings of ACM SIGMOD International Conference on Management of Data*, S. B. Davidson and C. Faloutsos (Eds.). Philadelphia, PA: ACM Press, pp. 61–72.
- Agrawal, R., Gehrke, J., Gunopulos, D., and Raghavan, P. (1998). Automatic subspace clustering of high dimensional data for data mining applications. In *Proceedings of ACM SIGMOD International Conference on Management of Data*, A. Tiwary and M. Franklin (Eds.). Seattle, WA: ACM Press, pp. 94–105.
- Ankerst, M., Berchtold, S., and Keim, D. A. (1998). Similarity clustering of dimensions for an enhanced visualization of multidimensional data. In *Proceedings of IEEE Symposium on Information Visualization*, G. Wills and J. Dill (Eds.). Research Triangle Park, NC: IEEE Computer Society Press, pp. 19–20.
- Asimov, D. (1985). The grand tour: A tool for viewing multidimensional data. *SIAM J. Scie. Statist. Comput.*, **6**(1): 128–143.
- Breunig, M. M., Kriegel, H. P., Ng, R. T., and Sander, J. (2000). LOF: Identifying density-based local outliers. *Sigmod Record*, **29**(2): 93–104.

- Chazelle, B., Drysdale, R. L., and Lee, D. T. (1986). Computing the largest empty rectangle. *SIAM J. Comput.*, **15**(1): 300–315.
- Cook, D., Buja, A., Cabrera, J., and Hurley, C. (1995). Grand tour and projection pursuit. *J. Computat. Graphic. Statist.*, **4**(3): 155–172.
- Edmonds, J., Gryz, J., Liang, D. M., and Miller, R. J. (2003). Mining for empty spaces in large data sets. *Theoret. Comput. Sci.*, **296**(3): 435–452.
- Eisen, M. B., Spellman, P. T., Brown, P. O., and Botstein, D. (1998). Cluster analysis and display of genome-wide expression patterns. *Proc. Natl. Acad. Sci. U.S.A.*, **95**(25): 14863–14868.
- Ferreira de Oliveira, M. C., and Levkowitz, H. (2003). From visual data exploration to visual data mining: A survey. *IEEE Trans. Visualizat. Comput. Graphics*, **9**(3): 378–394.
- Friedman, J. H. (1987). Exploratory projection pursuit. *J. Am. Statist. Assoc.*, **82**(397), 249–266.
- Friedman, J. H., and Tukey, J. W. (1974). Projection pursuit algorithm for exploratory data-analysis. *IEEE Trans. Comput.*, **C23**(9): 881–890.
- Friendly, M. (2002). Corrgrams: Exploratory displays for correlation matrices. *Am. Statist.*, **56**(4): 316–324.
- Guo, D. (2003). Coordinating computational and visual approaches for interactive feature selection and multivariate clustering. *Informat. Visualizat.*, **2**: 232–246.
- Guo, D., Gahegan, M., Peuquet, D., and MacEachren, A. (2003). Breaking down dimensionality: An effective feature selection method for high-dimensional clustering. In *Proceedings of the Third SIAM International Conference on Data Mining, Workshop on Clustering High Dimensional Data and Its Applications*. San Francisco, CA: USA. pp. 29–42.
- Hinneburg, A., and Keim, D. A. (1999). Optimal grid-clustering: Towards breaking the curse of dimensionality in high-dimensional clustering. In *Proceedings of the 25th International Conference on Very Large Data Bases*, M. P. Atkinson, M. E. Orłowska, P. Valduriez, S. B. Zdonik, and M. L. Brodie (Eds.). Edinburgh, Scotland: Morgan Kaufmann, pp. 506–517.
- Hinneburg, A., Keim, D. A., and Wawryniuk, M. (1999). HD-Eye: Visual mining of high-dimensional data. *IEEE Comput. Graphics Applicat.*, **19**(5): 22–31.
- Hottelling, H. (1933). Analysis of a complex of statistical variables into principal components. *J. Ed. Psychol.*, **24**: 417–441.
- Huber, P. J. (1985). Projection pursuit. *Ann. Statist.*, **13**(2): 435–475.
- Inselberg, A., and Dimsdale, B. (1990). Parallel coordinates: A tool for visualizing multi-dimensional geometry. In *Proceedings of IEEE Conference on Visualization*, A. Kaufman (Ed.). San Francisco, CA: IEEE Computer Society Press, pp. 361–378.
- Knorr, E. M., Ng, R. T., and Tucakov, V. (2000). Distance-based outliers: Algorithms and applications. *Vldb J.*, **8**(3–4): 237–253.
- Liu, B., Ku, L. P., and Hsu, W. (1997). Discovering interesting holes in data. In *Proceedings of International Joint Conference on Artificial Intelligence*. Nagoya, Japan: Morgan Kaufmann, pp. 930–935.
- Liu, H., and Motoda, H. (1998). *Feature Selection for Knowledge Discovery and Data Mining*. Boston: Kluwer Academic.
- MacEachren, A., Xiping, D., Hardisty, F., Guo, D., and Lengerich, E. (2003). Exploring high-d spaces with multiform matrices and small multiples. In *Proceedings of IEEE Symposium on Information Visualization*, T. Munzner and S. North (Eds.). Seattle, WA: IEEE Computer Society Press, pp. 31–38.
- Moore, D. S., and McCabe, G. P. (1999). *Introduction to the Practice of Statistics*, 3rd ed. New York: W. H. Freeman.
- Sammon, J. W. (1969). A nonlinear mapping for data structure analysis. *IEEE Trans. Comput.*, **C18**(5): 401–409.
- Seo, J., and Shneiderman, B. (2002). Interactively exploring hierarchical clustering results. *Computer*, **35**(7): 80–86.
- Seo, J., and Shneiderman, B. (2004). A rank-by-feature framework for unsupervised multidimensional data exploration using low dimensional projections. In *Proceedings of IEEE Symposium on Information Visualization*, M. Ward and T. Munzner (Eds.). Austin, TX: IEEE Computer Society Press, pp. 65–72.
- Seo, J., and Shneiderman, B. (2005). A rank-by-feature framework for interactive exploration of multidimensional data. *Informat. Visualiz.*, **4**(2): 99–113.
- Seo, J., and Shneiderman, B. (2006). Knowledge discovery in high-dimensional data: Case studies and a user survey for the rank-by-feature framework. *IEEE Trans. Visualiz. Comput. Graphics*, **12**(3): 311–322.

- Swayne, D. F., Lang, D. T., Buja, A., and Cook, D. (2001). GGobi: XGobi redesigned and extended. In *Proceedings of the 33rd Symposium on the Interface of Computing Science and Statistics*, E. J. Wegman, A. Braverman, A. Goodman, and P. Smyth (Eds.). Fairfax Station, VA: Interface Foundation of North America, pp. 64–76.
- Torgerson, W. S. (1952). Multidimensional scaling: I. theory and method. *Psychometrika*, **17**: 401–419.
- Tukey, J. W., and Tukey, P. A. (1985). Computer graphics and exploratory data analysis: An introduction. In *Proceedings of Annual Conference and Exposition: Computer Graphics*, Vol. 3. Silver Spring, MD: National Micrographics Association, pp. 773–785.
- Tukey, P. A., and Tukey, J. W. (1981). Graphical display of data sets in three or more dimensions. In *Interpreting Multivariate Data*. Chichester: Wiley, pp. 189–275.
- Ward, M. O. (1994). XmdvTool: Integrating multiple methods for visualizing multivariate data. In *Proceedings of IEEE Conference on Visualization*, R. D. Bergeron and A. E. Kaufman (Eds.). Washington, DC: IEEE Computer Society Press, pp. 326–333.

STATISTICAL MODELS, INFERENCE, AND ALGORITHMS FOR LARGE BIOLOGICAL DATA ANALYSIS

Debashis Ghosh

*Departments of Statistics and Public Health Sciences, Penn State University,
University Park, Pennsylvania, USA*

Seungyeoun Lee

Department of Applied Statistics, Sejong University, Seoul, Korea

Taesung Park

Department of Statistics, Seoul National University, Seoul, Korea

8.1 INTRODUCTION

With the completion of large-scale genome projects and the development of high-throughput technologies for assaying biological data, this century has seen the dawn of a revolution in terms of the amount of genomic information that is available to the user. This has spawned fields that are collectively known as “-omics.” What has been remarkable is that there are now technologies whose components are miniaturized so much that they allow for measurement of -omics data. This represents a major advance in data generation so that more underlying biological phenomena can be discovered and understood. Here are some examples of -omics fields, taken from the website Wikipedia (<http://en.wikipedia.org/wiki/-omics>).

1. Transcriptomics: The study of transcriptome, the mRNA complement of an entire organism, tissue type, or cell.
2. Epigenomics: The study of changes in phenotype (appearance) or gene expression caused by mechanisms other than changes in the underlying DNA sequence.
3. Proteomics: The study of the proteome, the protein complement of an entire organism, tissue type, or cell.

4. **Metabolomics:** The study of metabolome, the totality of metabolites in an organism.
5. **Lipidomics:** The study of lipidome, the totality of lipids.
6. **Glycomics:** The study of glycome, the totality of glycans, carbohydrate structures of an organism, a cell or tissue type.
7. **Interactomics:** The study of interactome, the totality of the molecular interactions in an organism.
8. **Spliceomics:** The study of the spliceome (see spliceosome), the totality of the alternative splicing protein isoforms.
9. **ORFeomics:** The study of the ORFeome refers to the totality of DNA sequences that begin with the initiation codon ATG, end with a nonsense codon, and contain no stop codon.

These technologies have permeated most medical and scientific research. In the area of human medicine, such technologies hold the potential for identification of new biomarkers useful in the clinical management of diseases as well as further advancing our knowledge in the etiology of disease. As an example, we consider the recent finding by Tomlins et al. (2005), who identified a gene fusion in prostate cancer. They discovered it by making the following observation. For certain genes, only a fraction of samples in one group were overexpressed relative to those in the other group; the remaining samples showed no evidence of differential expression. They used a method called cancer outlier profile analysis to rank such genes using expression data generated by microarrays. Using such a score identified one of the genes involved in the fusion event.

A drawback with the generation of such large datasets is a “data overflow” problem; in particular, it becomes much more difficult to make sense of such high-dimensional datasets. More crucially than ever, substantial statistical input is needed to make sense of the data being generated by -omics technologies. In this chapter, we will discuss the role of statistical inference in the analyses of -omics datasets. We will refer to this as genomic data analyses, although the term is meant to be quite generic. In thinking of the role of statistics, it is useful to break it down into three parts:

1. Statistical models
2. Statistical estimation
3. Numerical algorithms

In this chapter, we will discuss what each of these terms means. What is crucial to note is that they are all crucial to a statistical analysis. There are choices that need to be made for each of these components, which leads to a panoply of methods available to the data analyst. We will describe the major options for each, while assuming a level of knowledge of probability distributions at the level of that in Chapter 2. No mathematical derivation is presented; rather simple examples are used to illustrate the major ideas and concepts that are to be learned here. Finally, we will conclude discussion with an overview of the major analytical issues and methods that arise in the analysis of -omics data.

8.2 STATISTICAL/PROBABILISTIC MODELS

8.2.1 Motivation

In many instances, biological considerations or scientific theory posits the formulation of mechanistic models to describe biological phenomenon. One very famous example comes from Michaelis–Menten kinetics. This model describes the behavior of enzymes, in particular the relationship between an enzyme and the substrate it acts upon. If one assumes that these molecules are in a steady-state equilibrium, then the relationship among the substrate concentration ($[S]$), the enzyme concentration ($[E]$), and the concentration of enzyme that has substrate bound, that is, $[ES]$, is given by the equation

$$[ES] = \frac{k_1[E][S]}{k_{-1} + k_2}$$

where k_1 , k_{-1} , and k_2 are unknown constants that have to be estimated using experimental data. This is an example of a mechanistic or mathematical model. The key feature to note is that we have a set of parameters (here, k_1 , k_{-1} , and k_2) that represent rate constants in the forward and backward direction of the chemical reaction between the enzyme ($[E]$) and substrate ($[S]$). These parameters have a biological interpretation that is of scientific interest. It is this feature of the model that we will eventually like to make (a) an “estimate,” or guess, at its value using the collected data and (b) inference about in a so-called hypothesis-testing framework. A statistical or probabilistic model is very similar in spirit. What is different is a statistical attempt to explicitly take into account the idea that the phenomenon is stochastic in nature or that the observed measurements represent the true value plus or minus some noise distribution. In this example, there are several ways we can make the model above statistical. The first is to assume that the left-hand side of the equation is equal to the right-hand side plus a probabilistic error term:

$$[ES] = \frac{k_1[E][S]}{k_{-1} + k_2} + e$$

where e is an error term that is a random variable with an associated distribution with mean zero. This is an example of a so-called nonlinear statistical model, because k_1 , k_{-1} , and k_2 enter the model in a nonlinear way.

Another method would be to consider the constants in the model to be random variables, each with their own associated distribution. To do this, we would need measurements on several enzyme–substrate concentrations: for $i = 1, \dots, n$, where i indexes the experiment performed,

$$[ES]_i = \frac{k_{1i}[E]_i[S]_i}{k_{-1i} + k_{2i}} + e_i$$

where $(k_{1i}, k_{-1i}, k_{2i})$ are independently and identically distributed observations from some distribution function F of the parameters $(k_1, k_{-1}$, and $k_2)$. We refer this new model to a hierarchical model. Such a structure is very commonly used in statistics,

as it provides a nice and natural means for pooling information across experiments: Notice that this hierarchical model has three levels. At the top level are the parameters of the Michaelis–Menten model. At the bottom are the observed data, which are the enzyme, substrate, and enzyme–substrate complex measurements from $n = 3$ experiments, each one represented separately in a box. The second level, termed *intermediate parameters*, represents parameters that can be obtained from the individual experiments. In particular, we can estimate the rate constants for each experiment separately. If these parameters are “related” to each other, then we can combine information across the experiments to come up with an estimate of the “master” rate constants (k_1 , k_{-1} , and k_2). While we are presenting the model in the context of enzyme kinetics, we will see later that such a hierarchical specification turns out to provide a useful framework for the analysis of genomic data. Also there is no requirement that there be only one intermediate level. In principle, we can have more than one, depending on the scientific context.

8.2.2 Statistical Models

In terms of models, they fall into three categories: *parametric*, *nonparametric*, and *semiparametric*. A parametric model only contains a finite number of parameters. One of the simplest examples of a parametric statistical model is the following:

Model 1 X_1, \dots, X_n are independent and identically distributed observations from a Bernoulli distribution with probability p . These Bernoulli random variables take the value 0 and 1.

In model 2, we have exactly one parameter (p). At the other extreme is a nonparametric model, in which there are an infinite number of parameters. Here is an example of a nonparametric model:

Model 2 X_1, \dots, X_n are independent and identically distributed observations from a distribution with cumulative distribution function $F(x) = \Pr(X \leq x)$.

The reason that model 2 is a nonparametric model is because F is unknown so the value of F at every real number, x , represents an unknown parameter. Models that are neither parametric nor nonparametric are said to be semiparametric. An example of a semiparametric model is the following:

Model 3 X_1, \dots, X_n are independent and identically distributed observations from a distribution with an unknown mean $m = E(X)$ and the cumulative distribution function $F(x) = \Pr(X \leq x)$.

Model 3 is semiparametric because it contains a parameter of interest (m) as well as the infinite-dimensional parameter $F(x)$ for every value of x .

Intuitively, parametric models tend to yield estimates that are less variable than nonparametric and semiparametric models. However, parametric models are more sensitive to *model misspecification*. What model misspecification means is that if the distribution of the data does not match the model assumed, then the answers gotten from the model will tend to be quite wrong or, in statistical jargon, biased. By contrast, nonparametric and semiparametric models tend to make fewer

assumptions so that they will be less sensitive to model misspecification. Such a property is known as *robustness*.

Summary

- A statistical model incorporates stochasticity in order to describe a biological phenomenon and will contain parameters that are of interest to the investigator.
- Models can be parametric, nonparametric, or semiparametric.
- The advantage of parametric models is that estimates obtained from such models will tend to have lower variability than those obtained from semiparametric and nonparametric models.
- The advantage of semiparametric and nonparametric models is that they have more robustness than parametric models. If the true data model does not match the model being fit by the investigator, then estimates from the semiparametric and nonparametric models will have less associated bias than that from a parametric model.

8.3 ESTIMATION METHODS

Once we have specified a statistical model, containing unknown parameters, the next step is to figure out how to come up with a method of estimating the parameters based on the observed data. Parameter estimates represent our “best guess” at the values using the data we have collected. Put another way, we wish to figure out what values of the parameter space are the most likely to have generated the data that we are observing. As might be expected, there are many different definitions of how to define “what is likely.”

Perhaps the most common method for doing this is known as *maximum likelihood*. In one sense, maximum likelihood can be thought of as the dual to a probability density function specification. Let us take the example of model 1 from the previous section. There, X_1, \dots, X_n were assumed to be independent and identically distributed observations from a Bernoulli distribution with probability p . The maximum-likelihood estimate poses the following question: Given the assumption of a Bernoulli model and the fact that I have observed the values X_1, \dots, X_n , what is the most likely value of p that is consistent with the data? Mathematically, what maximum likelihood tries to do is maximize the function $L(p|x_1, \dots, x_n)$ as a function of p . The function L is identical to the probability model for X_1, \dots, X_n (i.e., the Bernoulli model). Some calculus and algebra can be used to show that the maximum-likelihood estimate of p in this model is the average of X_1, \dots, X_n , which seems like a sensible estimate. More generically, the way maximum likelihood works is to maximize

$$L(\text{parameters}|\text{data}) \tag{8.1}$$

where *parameters* are the parameters in the probabilistic or stochastic model being specified and *data* are the observed data collected by the investigator.

The advantage of the maximum-likelihood approach is that it will make full use of the collected information; this property is known as *efficiency*. Typically in functional genomic experiments, the number of independent samples collected by investigators is small relative to the dimensionality of the data being examined. Thus, it behooves investigators to have available analytical methods that are efficient.

The major disadvantage to maximum-likelihood estimation is that it crucially relies on the correct function L being specified in Eq. (8.1). By correct, what we mean is that the model being fit to the data is a close approximation to the true data-generating mechanism. If this is not happening, then the answers from the maximum likelihood will be wrong and lead to very misleading answers for the investigator.

The other dominant approach for estimation is known as Bayesian estimation. We do not attempt to give a detailed exposition of the topic here; much greater detail can be found in Gelman et al. (2004) and Carlin and Louis (2000). In this approach, the parameters are treated as random variables themselves. This is in contrast to the maximum-likelihood estimation procedure, in which the parameters are treated as unknown constants that only take one possible value. In the Bayesian framework, the goal is to make inference about the parameters. This is done in the following way. The parameters themselves have a certain probabilistic model that we call a prior distribution. We then construct a posterior probability model for the parameters by the use of Bayes's rule:

$$f(\text{parameters}|\text{data}) = \frac{L(\text{parameters}|\text{data})g(\text{parameters})}{\int L(\text{parameters}|\text{data})g(\text{parameters})} \quad (8.2)$$

We make some comments about the terms in Eq. (8.2). The left-hand side of (8.2) is referred to as the posterior distribution of the parameters given the observed data. The function g corresponds to the probability model we wish to specify for the parameters of the model. It is known as the prior distribution. Both the likelihood and prior distributions are needed to perform Bayesian inference. As more and more data are collected, the influence of the prior distribution relative to the likelihood function diminishes. The integral in the denominator represents a sum taken over all possible values of the parameters that can go into the function. The denominator serves as a so-called normalizing constant so that the area under the curve for the posterior distribution is equal to 1.

As a simple example, we consider the following model. Let X_1, \dots, X_n be independent and identically distributed observations from a normal distribution with mean m and variance s^2 .

Thus, there are two parameters in the model. Furthermore, let m have a normal distribution with mean zero and variance t^2 and suppose that the true value of s^2 is known. This effectively makes our model have only one parameter, m . We can plug these models into the Bayes rule machinery, as embodied in Eq. (8.2); some algebraic calculations would reveal that the posterior distribution of m given the data has a closed-form solution. It is normal with mean $nt^2\bar{x}/(nt^2 + s^2)$ and variance $(nt^2 + s^2)/s^2t^2$. In the mean, the bar over the x denotes the average of the observations.

Since there is now a normal probabilistic model for m given the data, one can generate all different sorts of features from the model summarizing variability, confidence about parameter values, and so on. We also point out in passing that the mean of the posterior distribution is actually a weighted average of the observed mean (\bar{x}) and the mean of the prior distribution (zero). This is known as a shrinkage effect and demonstrates how the Bayesian approach can pool the current data collected with prior information.

In practice, the models we will wish to fit will be much more complicated than the normal–normal example that is presented here. For those situations, it will often be the case that (8.1) will not have a closed-form solution. This will necessitate the use of so-called Monte Carlo algorithms to approximate the posterior distribution. We will discuss these algorithms in the next section.

Summary

- In terms of estimation procedures, the two dominant paradigms currently are the maximum-likelihood method and the Bayesian framework.
- The advantage of the maximum likelihood is that it will be efficient if the likelihood function matches the true data-generating model. If it does not, then the answers can be very misleading.
- The other dominant approach is known as Bayesian inference. In this approach the parameters have probabilistic models themselves. By Bayes’s rule, we calculate the posterior distribution of the parameters given the data, which will be used for inference. The Bayesian approach requires specification of a prior distribution.

8.4 NUMERICAL ALGORITHMS

In the previous section, we discussed various modes of inference about the parameters in the models that we specify for the data. We now wish to discuss the actual numerical computations necessary to obtain the parameter estimates. It should be noted that while maximum likelihood and Bayesian inference are modes of estimation, the choice of the numerical algorithm is a separate decision.

Let us start with the maximum-likelihood estimation procedure. In the ideal case, L in (8.1) will be a “nice” function in that it will be a differentiable and smooth function. For such cases, one can employ a Fisher scoring, or more generally a Newton–Raphson method, which effectively uses gradient searches to find the optima of the function. Such algorithms will tend to have very fast convergence rates. If the function L is not a nice function, then one will have to use more complicated algorithms. Examples include Nelder–Mead optimization, simulated annealing, majorization algorithms, and constrained programming methods.

Certain probabilistic models have a special structure in that there is “missing” data. If the missing data were not present, then optimization would be easy. For such problems, what is recommended is an algorithm known as the expectation–maximization (EM) algorithm. An excellent review of the algorithm can be found

in the text by McLachlan and Krishnan (2002). The basic idea of the EM algorithm is to iterate between the following two steps until convergence:

1. Expectation—fill in the missing data with a guess of them, namely their expectation given the observed data.
2. Maximization—using the observed data and the filled-in data from step 1, maximize the “likelihood” function.

In step 1 of the algorithm, the quotes around *likelihood* means it is no longer the likelihood function of the observed data but rather a likelihood function that would have been obtained if the investigator had also observed the missing data. While we have described the EM algorithm in a fairly general way here, the list of models that can be estimated with this type of algorithm is extensive. Broadly speaking, any sort of so-called mixture model (Lindsay, 1995) can be estimated with the EM algorithm. Unlike the Newton–Raphson method, the convergence based on the EM algorithm tends to be slower. As an example, a very popular model to fit in bioinformatics is called a hidden Markov model. For a monograph describing its application to sequence analysis problems, the reader is referred to Durbin et al. (1998). The forward–backward algorithm used for the estimation of hidden Markov models is a special case of the EM algorithm.

We now move to estimation in the Bayesian inferential paradigm. As mentioned in the previous section, the objective is to be able to calculate the posterior distribution, defined in Eq. (8.2). For most models, it will not be possible to get a closed-form solution, unlike the normal–normal example given in the previous section. We resort to a class of algorithms known as Monte Carlo sampling algorithms. The idea is to derive conditional distributions corresponding to the posterior distribution and to simulate from these conditionals in order to compute the posterior distribution. A detailed text on Monte Carlo sampling algorithms can be found in Robert and Casella (2004). It should be pointed that, while these algorithms tend to be quite useful for Bayesian inference, they can also be used for maximum-likelihood estimation. Again, the choices of model, estimation, and algorithm are all orthogonal components in a statistical modeling procedure.

8.5 EXAMPLES

In the previous sections, statistical models are introduced in describing biological phenomena along with the statistical estimation procedure and numerical algorithm. We now combine the ideas from the previous sections to discuss a class of models that is commonly used in the statistical literature, known as regression models. To make the example concrete, suppose that we are performing a microarray experiment. Microarray expression profiling has been widely applied to biological studies because of its ability to simultaneously examine tens of thousands of gene expression patterns. Microarray experiments have also proven to be quite useful for investigating the associations between genes and complex mechanisms of many human diseases. Several statistical modeling approaches have also been used to analyze the gene

expression data along with other response variables such as treatment group variables (Park et al., 2003), physiological response variables, and survival times (van de Vijver et al., 2002). In this section, we provide three types of statistical models: regression model, analysis-of-variance (ANOVA) model, and survival analysis model.

8.5.1 Regression Models

Regression models are commonly used to model the functional relationship among variables. The Michaelis–Menten kinetics model is an example of a regression model which describes the behavior of enzymes and in particular the relationship between an enzyme and the substrate it acts upon. In regression models, there are two types of variables: dependent variable (response variable or outcome variable) and independent variables (explanatory variable or predictor variable). In a regression model, the dependent variable is modeled as a function of one or more independent variables. When this function is a linear combination of one or more model parameters, called regression coefficients, the model is called a linear regression model. If it is a nonlinear function, then the model is called a nonlinear regression model. The parameters are estimated by maximum-likelihood estimation described in Section 8.3 or least-squares estimation to give a best fit of the data.

Regression models can be used for predicting outcomes and hypothesis testing. In order to illustrate regression models, consider a real example of docetaxel chemosensitivity and microarray gene expression data on the NCI-60 cancer cell lines (Cho et al., 2007). The NCI-60 cell line panel consists of nine cancer subtypes: lung, colon, breast, ovarian, leukemia, renal, melanoma, prostate, and central nervous system cancers. Docetaxel is one of the most widely used antineoplastic chemotherapeutic compounds to treat various tumors such as breast, non-small-cell lung, gastrointestinal (stomach), and prostate cancers (Developmental Therapeutic Program NCI/NIH). Major target genes of docetaxel are known to be *BCL2* and *TUBB1*. In this illustration, we use in vitro drug activity data of docetaxel on the NCI-60 cancer cell line panel, so-called GI50 (50% growth inhibition dose concentration in two-day assays), together with NCI-60 genomewide expression profiling data of Affymetrix HG-U133A oligonucleotide microarrays. Let the dependent variable Y be the log transformed value of GI50. Let X_1, \dots, X_p be p different gene expression values from Affymetrix HG-U133A. In order to identify genes that are strongly correlated with the GI50 values of docetaxel on NCI-60, we consider the following regression model:

$$E(Y_i) = \alpha_g + \beta_g X_{gi} \quad (8.3)$$

where index g is for gene on the Affymetrix HG-U133A array and the index i is for the arrays. In Eq. (8.3), Y denotes the gene expression measurement for the g th gene from the i th array, α_g denotes the gene-specific intercept of the log-transformed GI50, and β_g denotes the effect of gene g on GI50. This right-hand side of model contains corresponding *regression parameters* α_g and β_g . Note that this model can be written using an *error term* e_i as follows:

$$Y_i = \alpha_g + \beta_g X_{gi} + e_i \quad (8.4)$$

where the error term e_i represents the unexplained variation in the dependent variable. This statistical representation of the regression model is more commonly used than (8.3). The difference between Eqs. (8.3) and (8.4) is that there is no e_i term in (8.3); this was removed because of the presence of the expectation operator E on the left-hand side of (8.3). By rewriting (8.3) into (8.4), we have now made the model probabilistic in an explicit fashion. If we specify the distribution or density of e_i in (8.4), then we can use maximum likelihood to estimate the parameters (α_g, β_g) . A very popular model to use for the error term is to assume it has a normal distribution with mean zero and some variance. Then the maximum-likelihood estimation of (8.4) proceeds by the least-squares method; in fact, the maximum-likelihood estimators can be found in closed form.

Based on the regression coefficient β_g , the significance test can be performed to identify genes which are either positively ($\beta_g > 0$ with a significant p -value) or negatively ($\beta_g < 0$ with a significant p -value) highly correlated with the GI50 values.

Note that this model contains a single independent variable. In addition to gene expression values, some other independent variable Z can be included in the regression model, such as the cell type or cancer subtype. Then, the model becomes

$$Y_i = \alpha_g + \beta_g X_{gi} + \gamma Z_i + e_i$$

where γ denotes the effect of Z on GI50. Sometimes, more than one gene may influence a dependent variable simultaneously. Two genes may have a joint effect on the response variable GI50 values of docetaxel on NCI-60 in our example. The regression model can be easily extended to include more than one gene. For example, the model with two genes g and g' along with an additional independent variable Z is given by

$$Y_i = \alpha_g + \beta_g X_{gi} + \beta_{g'} X_{g'i} + \gamma Z_i + e_i$$

In regression models, the independent variables are usually quantitative or continuous variables. When the independent variables consist of all qualitative (grouped or categorical) variables, the model is the ANOVA model. When the independent variables consist of both qualitative variables and quantitative variables, the model is the analysis of covariance (ANCOVA) model. The next section illustrates the ANOVA model.

A linear regression model need not be a linear function of the independent variable: *linear* in this context means that the conditional mean of Y is linear in the parameters but it may not be nonlinear in independent variables. A nonlinear regression model is nonlinear in the parameters.

8.5.2 ANOVA Models

We now introduce a new class of models that is commonly used in the statistical literature, the ANOVA models, which are special regression models using all dummy variables. Suppose that we are performing a gene expression experiment in which we are profiling samples using Affymetrix U95A oligonucleotide microarrays.

Suppose that there are 100 samples, 25 from four different groups, numbered 1, 2, 3, and 4. For this situation, a very popular model to fit to the data is the following:

$$E(Y_{gi}) = \alpha_g + \beta_{1g}I(\text{Group}_i = 2) + \beta_{2g}I(\text{Group}_i = 3) + \beta_{3g}I(\text{Group}_i = 4) \quad (8.5)$$

As introduced in the regression model, the index g is for the gene on the Affymetrix array, $g = 1, \dots, 22,283$, and the index i is for the individual, $i = 1, \dots, 100$. In Eq. (8.5), Y denotes the gene expression measurement for the g th gene from the i th sample, α_g denotes the gene-specific mean of the expression of the g th gene, $(\alpha_g, \beta_{1g}, \beta_{2g}, \beta_{3g})$ is the effect of the group in which the i th subject is in. The I in Eq. (8.5) is known as an indicator function and takes a value of 1 if the event is true and 0 otherwise. What model (8.5) does is to partition the variability in the gene expression measurements for the g th gene into the variability due to the groups from which the samples were collected from. The β parameters are interpreted as the difference in average expression for the g th gene relative to the baseline group, which in this case was taken to be group 1.

In order to estimate the parameters in Eq. (8.5) by using the maximum-likelihood estimation procedure in Section 8.3, we first note that Eq. (8.5) can be rewritten as

$$Y_{gi} = \alpha_g + \beta_{1g}I(\text{Group}_i = 2) + \beta_{2g}I(\text{Group}_i = 3) + \beta_{3g}I(\text{Group}_i = 4) + e_{gi} \quad (8.6)$$

where e_{gi} is a random variable with mean zero. By rewriting (8.5) into (8.6), we now have made the model probabilistic in an explicit fashion. If we specify the distribution or density of e_{gi} in (8.6), then we can use maximum likelihood to estimate the parameters $(\alpha_g, \beta_{1g}, \beta_{2g}, \beta_{3g})$, as in regression models. A very popular model to use for the error term is to assume it has a normal distribution with mean zero and some variance.

Although the maximum-likelihood approach is a standard procedure to estimate parameters in regression models and ANOVA models, its application as -omics data may not be straightforward. One aspect of the experiment is that there are 22,283 genes, while there are only 100 samples. The maximum-likelihood procedure described in the previous paragraph has to be fit 22,283 times, once for each gene. It might be desirable to attempt to pool information across genes so as to reduce the variation in the gene-specific parameter estimates. For this, the hierarchical model paradigm discussed in Section 8.2 can be very useful. This requires us to assume that $(\alpha_g, \beta_{1g}, \beta_{2g}, \beta_{3g})$ has a distribution across genes, or equivalently satisfies a probabilistic model with parameters θ . In the notation of Figure 8.1, the observed data are Y and the group indicators $(\alpha_g, \beta_{1g}, \beta_{2g}, \beta_{3g})$ are the intermediate parameters, and the hierarchical parameter is θ . The equations would be as follows:

$$Y_{gi} = \alpha_g + \beta_{1g}I(\text{Group}_i = 2) + \beta_{2g}I(\text{Group}_i = 3) + \beta_{3g}I(\text{Group}_i = 4) + \varepsilon_{gi} \\ \times (\alpha_g, \beta_{1g}, \beta_{2g}, \beta_{3g}) \sim F_\theta \quad (8.7)$$

The two equations specified in (8.7) are an example of a hierarchical model. In terms of estimation, one approach would be maximum-likelihood estimation. It turns out that in (8.7), the vector $(\alpha_g, \beta_{1g}, \beta_{2g}, \beta_{3g})$ can be treated as missing

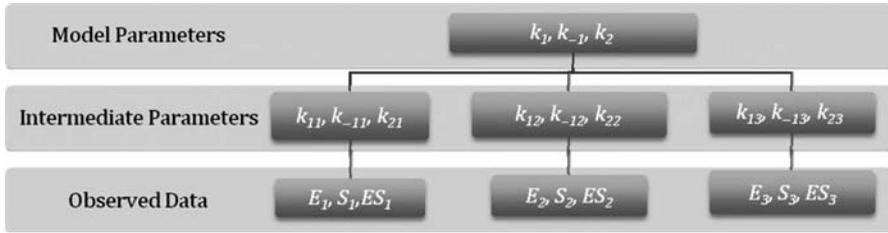


Figure 8.1 Hierarchical model paradigm.

data so that maximum-likelihood estimation can be performed using the EM algorithm. Alternatively, one could adopt a Bayesian approach in which the goal would be the posterior distribution of $(\alpha_g, \beta_{1g}, \beta_{2g}, \beta_{3g})$ given the data. In this case (8.7) specifies a hierarchical model in which F_θ plays the role of the prior distribution. Unless F has a very restrictive form, the posterior distribution will not have a closed form so that Monte Carlo sampling methods will be necessary in order to approximate the posterior distribution by simulation.

8.5.3 Survival Models

Recently, many studies have been developed in linking gene expression profiles to survival data, such as patients’ overall survival time or time to relapse. For example, suppose that there were gene expression measurements of 20,000 and 200 patients with lung cancer. Among 200 patients, 120 patients died of lung cancer and the remaining 80 patients were still alive at the end of the study. The special characteristic of the survival data is that the survival time cannot be completely observed but can be “censored” due to the end of study or “lost to follow up” due to withdrawal or drop-out during the study. In this example, the survival times for 120 patients were completely observed but those for 80 patients were censored at the end of the study. For this situation, in order to identify important genes that are related to the survival time, the most popular method is to fit the Cox regression model (Cox, 1972), specified as follows:

$$\lambda(t|X) = \lambda_0(t) \exp(\beta_1 X_1 + \beta_2 X_2 + \dots + \beta_p X_p) = \lambda_0(t) \exp(\beta'X) \quad (8.8)$$

Here $\lambda(t|X)$ is the hazard function for given predictors, $X = \{X_1, X_2, \dots, X_p\}$, $\lambda_0(t)$ is an unspecified baseline hazard function and $\beta_1, \beta_2, \dots, \beta_p$ are the regression coefficients of the predictors, $X = \{X_1, X_2, \dots, X_p\}$, which is a vector of gene expression measurements with the corresponding sample values of $x_i = \{x_{i1}, \dots, x_{ip}\}$ for the i th sample. The index i is for the individual, $i = 1, \dots, 200$, and the index p is for the gene expression profiles, $p = 1, \dots, 20,000$. The model in (8.8) can be rewritten like a regression model in (8.3) by taking the log function as follows:

$$\log \frac{\lambda(t|X)}{\lambda_0(t)} = \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_p X_p \quad (8.9)$$

The left-hand side in (8.9) is the log hazards ratio, which plays a role of $E(Y_i)$ in (8.3) and denotes the log odds ratio of having the gene expression profiles of $X = \{X_1, X_2, \dots, X_p\}$ relative to $X = \{0, 0, \dots, 0\}$, and $\{\beta_1, \beta_2, \dots, \beta_p\}$ is the effect of the gene expression profiles on the log odds ratio. The β_j parameter is interpreted as the log relative risk when the gene expression level of X_j is increased by unit 1 for the other gene expression levels fixed. The positive value of β_j represents that the relative risk increases as the level of X_j increases while the negative value of β_j represents that the relative risk decreases as the level of X_j increases. Therefore, the positive value of β_j has the effect of shortening the survival time while the negative value of β_j makes the survival time prolonged.

As the next step, in order to estimate the regression coefficients, $\beta_1, \beta_2, \dots, \beta_p$, in model (8.8), Cox's partial likelihood is maximized with respect to the regression coefficients, which provides the efficient estimators. Cox's partial likelihood (Cox, 1975) was proposed to provide the very efficient estimators which are also easily obtained. Cox's partial likelihood is specified as follows:

$$L(\beta|\text{data}) = \prod_{i \in D} \frac{\exp(\beta'x_i)}{\sum_{j \in R_i} \exp(\beta'x_j)}$$

Here D denotes the set of indices of the events (e.g., deaths) and R_i denotes the set of indices of the individuals at risk at time $t_i - 0$.

However, due to the very high dimensional space of the predictors, that is, the genes with expression levels measured by microarray experiments, the standard maximum Cox partial-likelihood method cannot be applied directly to estimate the parameters. As mentioned in the previous sections, in genomic experiments, the number of independent samples is very small relative to the dimensionality of the data being examined. In this example, the sample size is 200 while the number of predictors is 20,000 gene expression levels. Besides the high dimensionality, the expression profiles of some genes are highly correlated, which yields the problem of multicollinearity. To solve a multicollinearity problem, the most popular approach is to use the penalized partial likelihood, which includes both the L_2 penalized estimation, which is called the ridge estimation, and the L_1 penalized estimation, which is called the least absolute shrinkage and selection operator (Lasso) estimation (Tibshirani, 1997). Comparing these two methods, the L_2 penalized estimation uses all gene expression profiles and does not provide the subset selection of the relevant genes for prediction of the survival time. On the other hand, the L_1 penalized estimation is maximized by the quadratic programming procedure to estimate the parameter and cannot be applied directly to the settings when the sample size is much smaller than the number of predictors. Many methods have been proposed to modify these penalized estimation procedures in selecting the important predictive genes for survival using the microarray gene expression profiles.

Alternatively, the methods for reducing the high dimensionality have been proposed by considering the correlation structure within predictors as well as the correlation between predictors and survival times. The most effective method for reducing the high-dimensional data is principal-component analysis (PCA), in which the correlation between gene expression profiles is captured. However, PCA

has a limitation that the latent variable identified by the first principal component may not be relevant to the survival time because including all genes in finding the principal component by PCA yields noise and poor predictive performance. To overcome this shortcoming of PCA, supervised PCA (SPCA) has been proposed by Bair and Tibshirani (2004), which estimates the principal components from a selected subset of gene expression profiles instead of all genes. The selected subset of genes is obtained by the supervised procedure in which the survival time is used to select the subset of genes. It is shown that the estimation of the principal components from a subset of genes improves the prediction accuracy in the SPCA algorithm compared to the PCA algorithm without the gene-screening procedure.

The other possible method for reducing the high dimensionality of predictors is to consider the cluster structure in which clusters consist of coregulated genes with coordinated function. A large number of gene expression profiles are divided into clusters by using the various clustering methods such as the K -means and hierarchical methods. The optimal number of clusters is often determined by the gap statistic (Tibshirani et al., 1999). As a result, the large number of genes can be reduced to the relatively small number of clusters and the mean expression levels or median expression levels are used as representative measures for each cluster. Then the important clusters are selected for predicting the survival time by using the standard Lasso method at the cluster level. The rationale of taking into account the cluster structure in selecting important genes is that complex diseases such as cancer and HIV may be caused by mutations in gene pathways, instead of individual genes. In general, however, the biologically functional clusters and the statistical clusters are not perfectly matched, although they tend to have correspondence. In many cases, the selected clusters consist of a large number of individual genes, but some genes within clusters may not be related to the survival time. To build a more parsimonious and predictive model, the supervised group Lasso (SGLasso) method is proposed in which the selection procedure is performed in two steps. In the first step, important genes within each cluster are identified using the Lasso method, and in the second step, important clusters are selected using the group Lasso in which each cluster is regarded as an individual variable and the Lasso method is applied at the cluster level. Since the within-cluster gene selection is taken into account in the first step, SGLasso yields more parsimonious models as well as identifies the important individual genes.

We introduced many methods for selecting the relevant genes from the high-dimensional data in the Cox regression model. As explained, these methods are deeply involved in numerical algorithms such as quadratic programming, penalization and dimension reduction methods, including the PCA and clustering methods. Hence it is also very essential to develop these numerical algorithms more effectively for identifying important genes to predict the survival time.

8.6 CONCLUSION

In this chapter, we have attempted to describe the various parts that go into a statistical modeling procedure. In particular, one needs a model, an estimation method, and a numerical algorithm. There are many options one can specify at each stage.

A useful skill for the reader when surveying the bioinformatics literature is to find each of the three components when reading about a particular analysis tool. In particular, there have been many algorithms developed in the bioinformatics community in which all that is proposed is an algorithm without a formal stochastic model. By not specifying a model, several limitations exist:

- One does not know if there is a meaningful biological interpretation to the output from the algorithm.
- One does not know about the necessary assumptions/probabilistic model under which the proposed method will have any optimality properties.
- Assessing variability of the procedure may be problematic as well because the nature of the stochasticity is not explicitly described.

A key goal for future research in this area is that if there are new methods that appear to be successful but are simply motivated as algorithms, then statisticians ought to seek to develop modeling frameworks in which the methods can be evaluated.

In terms of a practical data analysis guide for -omics data, the article by Allison et al. (2006) is highly recommended. The references provide further resources for those who wish to learn the details of the methodologies alluded to in the chapter.

REFERENCES

- Allison, D. B., Cui, X., Page, G. P., and Sabripour, M. (2006). Microarray data analysis: From disarray to consolidation and consensus. *Nat. Rev. Genet.*, **7**(1): 55–65.
- Bair, E., and Tibshirani, R. (2004). Semi-supervised methods to predict patient survival from gene expression data. *PLOS BIOL.*, **2**: 511–522.
- Carlin, B. P., and Louis, T. A. (2000). *Bayes and Empirical Bayes Methods for Data Analysis*, 2nd ed. Boca Raton, FL: Chapman and Hall/CRC Press.
- Casella, G., and Robert, C. (2004). *Monte Carlo Statistical Methods*, 2nd ed. New York: Springer-Verlag.
- Cho, H., et al. (2007). Induction of dendritic cell-like phenotype in macrophages during foam cell formation. *Physiol. Genomics*, **29**: 149–160.
- Cox, D. R. (1972). Regression models and life-tables. *J. R. Statist. Soc. Ser. B (Methodol.)*, **34**(2): 187–220.
- Cox, D. R. (1975). Partial likelihood. *Biometrika*, **62**(2): 269–276.
- Durbin, R., Eddy, S., Krogh, A., and Mitchison, G. (1998). *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge, UK: Cambridge University Press.
- Gelman, A., Carlin, J. B., Stern, H. S., and Rubin, D. B. (2004). *Bayesian Data Analysis*, 2nd ed. Boca Raton, FL: Chapman and Hall/CRC Press.
- Lindsay, B. G. (1995). *Mixture Models: Theory, Geometry and Applications*. Institute of Mathematical Statistics.
- McLachlan, G. J., and Krishnan, T. (2002). *The EM Algorithm and Extensions*. New York: Wiley.
- Park, T., et al. (2003). Statistical tests for identifying differentially expressed genes in time-course microarray experiments. *Bioinformatics*, **19**(6): 694–703.
- Tibshirani, R. (1997). The Lasso method for variable selection in the Cox model. *Statist. Med.*, **16**: 385–395.
- Tibshirani, R., Hastie, T., Eisen, M., Ross, D., Botstein, D., and Brown, P. (1999). Clustering methods for the analysis of DNA microarray data. Available: <http://www-stat.stanford.edu/~tibs/ftp/jcgs.ps>.
- Tomlins, S. A., et al. (2005). Recurrent fusion of TMPRSS2 and ETS transcription factor genes in prostate cancer. *Science*, **28**;310(5748): 644–648.
- van de Vijver, M. J., et al. (2002). A gene-expression signature as a predictor of survival in breast cancer. *N. Engl. J. Med.*, **19**;347(25): 1999–2009.

EXPERIMENTAL DESIGNS ON HIGH-THROUGHPUT BIOLOGICAL EXPERIMENTS

Xiangqin Cui

Department of Biostatistics, University of Alabama, Birmingham, Alabama, USA

Experimental design is a term used to describe the planning of an experiment with the goal of making the experiment more efficient (obtaining the most information with the least amount of effort and resources). It is an important part of any experiment regardless of the nature of the experiment. Statisticians have been studying the design of experiments that involve variation and uncertainty and came up with some guiding principles (Fisher, 1926; Kuehl, 2000). Although these principles were obtained from low-throughput experiments, most of them are applicable to high-throughput technologies. In this chapter, we will discuss these design principles and examine their applications in high-throughput biotechnology using microarray technology as an example. We will also discuss some unique design aspects of microarray experiments which can be similarly utilized in other types of high-throughput biological experiments

9.1 RANDOMIZATION

The first principle of experimental design is randomization. It dictates that the experimental subjects should be randomly assigned to the treatments or conditions to be studied (Fisher, 1935). The purpose of randomization is to eliminate unknown factors that potentially affect results. For example, if we want to compare two diets for mice in weight gain using 10 gender-, age-, and weight-matched inbred mice, our randomization of mice to both diets can eliminate the influence of some other factors, such as health condition and litter, which are potentially influential in weight gain.

Randomization is commonly achieved by first numbering the subjects and then randomly drawing the numbers on a computer or tossing a coin. In our mouse example, the mice labeled with the first five randomly picked numbers can be assigned to one diet and the rest can be assigned to the other diet. A common misconception is that we will obtain a random mouse if we put all the mice in a case and randomly catch

one. Chances are that the one caught first is the slowest, weakest, and least healthy. If we give the first five caught mice one diet and the rest another diet, the difference between the two observed diets could be due to the diet or health condition of the mice. In this case, the diet effect is confounded with the effect of the mouse's health condition.

High-throughput experiments, such as microarray experiments, are often complicated as they involve many steps. Besides randomizing samples to the treatments, there are also other steps where we should randomize to avoid bias. For example, we can randomize the arrays in respect to the samples to avoid array order or array batch bias. We can also randomize the order of performing labeling, hybridizations, and scanning to avoid the process timing effect. However, in some cases, it may not be possible to randomize samples to treatments or condition. For example, when we compare mutant-versus-control mice, we are not able to randomize the genotype across mice because a mutant mouse is mutant and a wild-type mouse is a wild type. In other cases, there are known varying factors that prevent us from randomizing samples in the experiment. For example, we know there is a large processing batch effect, but we cannot process all arrays in one batch. In this case, we need to treat the processing batches as blocks and randomize within blocks (details described later).

9.2 REPLICATION

9.2.1 General Principle

Replication is another basic principle of experimental design. The definition of replication is the independent repetition of the same experimental process and/or the independent acquisition of biological observations, so that their error variability can be estimated for evaluating the statistical significance of the observed phenomenon (Kuehl, 2000). Thus, such error estimation is essential for applying statistical analysis and inference techniques. Whenever we have doubts about whether a replicate is a true one, we just need to check whether we are independently repeating the whole process of creating the phenomenon. Replication makes it possible to estimate the variability associated with the results. If we do not have replicates, we have no way to know whether the experiment has variability and how much variability it possesses. The result we obtain is just the result for that one occurrence. We cannot make any prediction or inference about what would happen if we would do it again. With the estimated variation from replicates, we can make inference about not only the results we obtained but also the results we would obtain if we did the experiment again and again using the same settings.

There is a clear distinction between replications and *repeated measurements*. The latter refers to taking several measurements from a single occurrence of a phenomenon. To fully understand what a true replicate is, we need to understand the term *experimental unit*. A true replicate is simply a replicated experimental unit. An experimental unit is defined as the unit that is directly exposed and randomly assigned to the treatment independent of other units (Kuehl, 2000). In our mouse diet example, the treatments we want to study and compare are the two diets. The units to which

these treatments are directly applied are the individual mice. Replication at the mouse level is true replication. If we study the effects of two types of plant growth hormones on seedling leaves and decide to spray the seedlings grown in pots, then the seedling pot is the experimental unit because we do not spray individual seedlings independently; instead we spray the whole pot. Any individual seedling or leaf is not an experimental unit. Any measurements on them would be repeated measurements. If we rely on the repeated measurements to estimate the variability and make an inference, we can only infer about the particular sample we used for the experiment. We do not have information about the population. Another example is if we want to know the height of men and women in the United States but we just pick one man and one woman and measure each 50 times. We will know the variability of our measuring technique and obtain very accurate measurements of this particular man and woman. However, the results do not tell us anything about the men and women in the United States. On the other hand, if we randomly pick 50 men and 50 women within the United States, we will have a good idea about the height of men and women in the United States.

In microarray technology, researchers often use biological and technical replicates to distinguish replication at different levels (Churchill, 2002; Yang and Speed, 2002). Biological replicates are considered true replicates while technical replicates are replicates at a lower level than the biological ones, often measurements from the same RNA sample (Yang and Speed, 2002; Simon et al., 2003). Because of the complexity of microarray technology, there are potentially many different levels of technical replicates. For example, Figure 9.1 shows a microarray experiment that has three levels of replication: the first level is at the cell culture level. Four independent cell cultures were established and each receives one of two treatments. Two RNA samples are obtained from each cell culture and each sample is measured with two one-color arrays. In this experiment, the independent cell cultures are biological replicates. The replicates at RNA samples and arrays are technical replicates which are similar to the repeated measurements. They are less useful for identifying significantly expressed genes between the two treatments. However, technical replicates are essential in experiments designed for evaluating the technology and in identifying the sources of variation (Zakharkin et al., 2005). The variability between

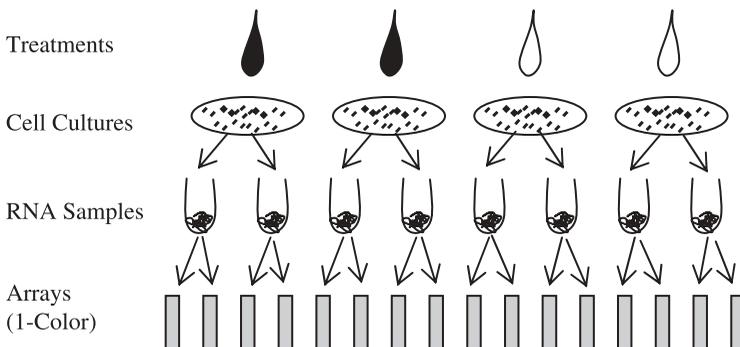


Figure 9.1 Different levels of replication in a microarray experiment.

the duplicated arrays estimates the variability of the procedure after RNA extraction and the variability between the duplicated RNA samples estimates the variability from both RNA extraction and the array hybridization.

9.2.2 Sample Size and Power

We know that it is important to have replicates and that there are different types of replicates so the next question we would naturally ask is how many replicates are needed. *Sample size* is the number of replicates in the experiment. It is an important aspect to consider for all experiments because it directly affects the amount of resource and effort needed to complete the experiment. There are many factors that affect the decision on the sample size. One is *power* ($1 - \beta$); the probability of declaring a positive if it is indeed a positive; β is called as type II error or *false-negative error rate*. It is similar to the concept of *sensitivity*. If we want our experiment to have a lot of power (very sensitive), we would naturally need a large sample size. Another factor that affects the choice of sample size is *effect size* (δ), which represents how big the actual difference is between the two treatments. In addition, variability (σ^2) and type I error rate (α) also affect sample size where the latter is the *false-positive error rate* or the probability of declaring a positive by chance even though it is a (random) negative observation. Sample size has to go up if the effect size decreases and/or variability increases and/or the type I error rate decreases (Kuehl, 2000). When the errors of the replicates or summary statistics of the replicates (e.g., sample mean) are closely distributed as a normal distribution, the relationship between sample size and these factors can be summarized using the following formula, where z is the standard normal distribution:

$$n = 2[z_{\alpha/2} + z_{\beta}]^2 \left(\frac{\sigma}{\delta}\right)^2$$

Another factor often considered in deciding sample size is the degree of freedom (df) for error. It is a measure of the number of independent pieces of information based on when the variability is estimated. In most experiments with limited sample sizes, the standard normal distribution in the above equation is actually a *t*-distribution, which is flatter than a standard normal distribution. The error df determines the flatness of the *t*-distribution. The larger the df, the closer the *t*-distribution is to a standard normal distribution. Too few degrees of freedom for the error can lead to an inaccurate estimate of the amount of variability in the experiment, resulting in a very flat *t*-distribution, which makes it harder to make a significant claim. Therefore, most experiments strive to have decent numbers of the df for error. The calculation for df is simple; it is the number of observations subtracted by the number of parameters estimated. For example, we may want to examine two factors (two types of diets and two growing environments) that affect mouse weight gain at the same time using a 2×2 factorial design (all four combinations of diets and environments). If we would like to estimate the diet effect, the environment effect, and the interaction of these two factors, Table 9.1 shows the degree of freedom for estimating the variability when different sample sizes (number of mice per group) are used. When the

TABLE 9.1 Degrees of Freedom for 2×2 Factorial Experiment

	$n = 1$	$n = 2$	$n = 3$
Overall mean	1	1	1
Diet	1	1	1
Environment	1	1	1
Diet and environment	1	1	1
Error	0	4	8
Total	4	8	12

Note: n represents the sample size.

sample size is 1, there is no df to estimate the variability. When the sample size is 2 and 3, there are four and eight df for estimating the variability. In general, at least five df is desired.

9.2.3 Software Packages for Microarray Sample Size Calculation

The sample size calculation described above is for traditional low-throughput experiments. In a microarray experiment, the observations from each gene are equivalent to one traditional experiment. However, there are thousands of genes which have different variability. The sample size calculation based on one gene does not fit all other genes. One strategy is to compute the sample size for each gene and then decide the overall sample size based on the overall power. For example, the `ssize` package (Table 9.2) in Bioconductor, which is freely available constructed using the free statistical language environment R, is based on the sample size calculation for each gene using a formula similar to the one shown above (Warnes and Liu, 2006). Given the required parameters, the software will calculate the sample size, power, or log fold change difference for each gene and provide graphs to summarize the sample size and power for all genes. Users can download the package from Bioconductor, install the package in R, and follow the script examples in the `ssize.pdf` document accompanying the package to run the package. There are also other packages in Bioconductor for calculating sample size and power for microarray experiments, such as `OCplus` and `sizepower`. The `OCplus` package computes the sample size based on the setting of the proportion of differentially expressed genes, the fold change, and the proportion of top-ranked genes (Pawitan et al., 2005). The `sizepower` package computes the sample size and power based on the setting of effect size, number of nondifferential genes, and expected number of false positives (Lee and Whitmore, 2002). Because these packages are based on different models and methods for sample size calculation, it is not always straightforward to compare with each other.

Here we provide some simple examples for sample size calculation using these packages. We assume that there are 10,000 genes and we want 90% power for detecting genes with a fold change of 2 (1 on \log_2 scale) in a two-group comparison experiment. Assume the standard deviation of the difference between two means is 0.5. Also assume that 95% of genes are nondifferential.

TABLE 9.2 Comparison of Software Tools for Sample Size Calculation

Parameter	ssize	OCplus	sizepower	PowerAtlas
Implementation	R/Bioconductor	R/Bioconductor	R/Bioconductor	Web tool
Source	www.bioconductor.org	www.bioconductor.org	www.bioconductor.org	www.poweratlas.org
Main functions	ssize	sampleSize	sampleSize.randomized, sampleSize.matched	
Inputs	sd of reference group, group mean difference, significance level	Proportion of nonDifferential genes, standardized group mean difference, % top genes to take	Expected false positives, none—differential gene proportion, group mean difference, sd of difference	<i>p</i> -Values, or custom pilot data, or select similar data from GEO
Outputs	Sample size	FDR for different Sample size	Sample size	Expected discovery rate, true positive rate, sample size
References	Warnes and Liu, 2006	Pawitan et al., 2005	Lee and Whitmore, 2002	Page et al., 2006

The sample size calculation using `ssize` is as follows:

```
>library(ssize)
>library(xtable)
>library(gdata)
>ssize(sd=0.6, delta=1, sig.level=0.05, power=0.9)
Output: 8.649245 (sample size)
```

The input `sd`, `delta`, and `sig.level` are the standard deviation from the reference group, difference between two group means on \log_2 scale, and significance level with Bonferonni correction, respectively. The obtained sample size can be rounded up to 9. Note that the input `sd` is often a vector of all the estimated variances from all genes instead of one value as shown here.

The sample size calculation using `OCplus` is as follows:

```
>library(OCplus)
>sampleSize(p0=0.95, D=2, crit=0.01)
Output:
          FDR_0.01      fdr_0.01
5  1.348339e-01    2.020542e-01
10 1.928410e-03    4.361612e-03
15 2.970820e-05    8.049273e-05
```

The input arguments `p0`, `D`, and `crit` are the proportion of nondifferential genes, difference between two group means on \log_2 scale in the unit of its standard deviation, and proportion of top-ranked genes to be examined for false discovery rate (FDR). The output is the FDR for the top 1% genes with different sample size.

The sample size calculation using the `sizepower` package is as follows:

```
>library(sizepower)
>sampleSize.randomized(ER0=1, G0=9500, power=0.9,
  absMu1=1, sigmad=0.5)
Output: n=7
```

The input arguments `ER0`, `G0`, `absMu1`, and `sigmad` are the expected false positives, number of true none-differential genes, absolute mean difference between the two groups, and the standard deviation of `absMu1`, respectively. The obtained sample size using this program is 7.

A less traditional way to calculate sample size and power for microarray experiments is based on the modeling of the collection of p -values obtained from all genes. For an experiment comparing different groups of treatments, if none of the genes is differentially expressed, the collection of p -values from all genes will follow a uniform distribution. If some genes are differentially expressed, their p -values tend to be small. Therefore, the collection of all p -values from all genes will have excess small p -values compared to a uniform distribution. This distribution of all p -values can be modeled as a mixture of two distributions, one uniform and one beta, and the mixing proportion of each component can then be estimated (Allison et al., 2002). Based on mixture modeling, an online power calculation tool, `PowerAtlas`

(<http://www.poweratlas.org/>) (Table 9.2), was established to use existing public data as preliminary data or provided p -values to calculate power and sample sizes (Page et al., 2006). Instead of calculating the traditional power (probability for a true positive being identified as a positive), they use an expected discovery rate (proportion of true positives being called positive) and a true positive rate (proportion of genes called positive are true positive), which is 1-False positive rate. Users can visit the website and select the proper organism and array type. The tool will show the available public data for users to choose an experiment that is close to their experiment. Users can also upload their own preliminary data. Because the tool is based on the p -value distribution following the mixture of normal and beta distributions, it is important to check the p -value distributions before using their own preliminary data. Bad p -value distributions are common and they do not work with this tool.

9.2.4 Multilevel Replication and Resource Allocation

As shown in Figure 9.1, there could be multiple levels of replications in one experiment. Although we emphasize that biological replicates are the most important replication for identifying differentially expressed genes, in certain circumstances, increasing technical replicates can be a wise choice. For example, if we test the treatment effect on a strain of expensive transgenic mice, we are unable to afford many replicates of the transgenic mouse. Increased technical replicates will help us accurately measure the gene expression of a relatively small number of transgenic mice, especially when the technical variability is larger than the biological variability (Cui and Churchill, 2003). If we assume that the cost of each mouse is C_M and the cost of each array is C_A , the total material cost for the experiment can be simply calculated as the sum of the mouse cost and the array cost, $\text{Cost} = mC_M + mnC_A$ for m mice and n arrays per mouse. For a simple design using one-color microarrays, we can calculate n as $n = (\sigma_A^2/\sigma_M^2)^{1/2}(C_M/C_A)^{1/2}$ to achieve minimum overall cost. Based on the median estimates of mouse and array variance components, $\sigma_A^2 = 0.0168$ and $\sigma_M^2 = 0.009$, from the project normal dataset (Pritchard et al., 2001), the optimal number of arrays for one mouse is $n = (0.0168/0.009)^{1/2}(15/600)^{1/2} = 0.2$, which rounds up to one array per mouse, if the array and mouse prices are \$600 and \$15, respectively (Table 9.3). On the other hand, if the mouse price is \$1500, the optimum number of arrays per mouse is calculated as $n = (0.0168/0.009)^{1/2}(1500/600)^{1/2} = 2.2$, which indicates that two arrays per mouse will be more efficient. The increased

TABLE 9.3 Number of Arrays per Mouse Related to Mouse Price

Mouse Price	Array Price	Number of Arrays per Mouse
\$15	\$600	1
\$300	\$600	1
\$1500	\$600	2

technical replicates will help to fully utilize the expensive mice to minimize the overall cost of the experiment. Note that this calculation only considers mouse and array costs. Other costs such as mouse maintenance and tissue extraction can also affect the decision if the costs are substantial.

9.3 POOLING

Pooling the biological replicates has the potential to reduce biological variability by measuring the average instead of individuals. Therefore, it is a very appealing process to investigators when biological samples are inexpensive and readily available. Theoretically, pooling can reduce the biological variability to $1/n$, where n is the number of individual biological replicates in each pool. However, this ideal ratio is almost never achieved. One reason is that the pooling process has variability itself (Zhang et al., 2007; Shih et al., 2004). The pool is never a perfect mix of equal amounts for each individual. More importantly, pooling is at the original scale (RNA or tissue level), while variance is often calculated on the log signal intensity level (Kendzierski et al., 2005; Zhang et al., 2007). Despite the imperfection of pooling, it can be beneficial in improving the power of the experiment with the same numbers of arrays (Zhang et al., 2007; Kendzierski et al., 2003). The benefit depends on several factors:

1. The ratio of biological variance to the technical variance. Only when this ratio is large, that is, the biological error is much bigger than the technical error, does the benefit become substantial.
2. The design of the experiment. Pooling is beneficial when a small number of arrays and a small number of pools are used.
3. The availability and acquisition of sample materials for experiment.

If there are a large number of arrays per treatment, pooling individuals for each array does not bring much improvement unless pooling a very large number of individuals, which may offset the financial benefit of pooling. Pooling is more beneficial when a small number of arrays and a small number of pools are used for each treatment; however, too small a number of pools make the estimate of biological variability unreliable, which causes some misleading results. For example, if we have 20 subjects in each treatment, should we make 2 pools with 10 subjects in each pool or 10 pools with 2 subjects in each, or 5 pools with 4 subjects in each? Assume that we will use one array per pool; the choice of 2 pools of 10 will use the smallest number of arrays but the estimate of biological variability will be very unreliable—therefore it is not a good choice. The option of 10 pools of 2 is the most powerful but will use 10 arrays per treatment, which may dramatically increase the cost. The option of 5 pools of 4 may be the best choice. Of course, the decision is based not only on the reduction of the number of arrays but also on the cost of individual biological replicates, the ratio of biological and technical variances, and other limits. In certain studies, pooling is necessary to obtain enough RNA for one microarray hybridization, such as working with *drosophila*.

9.4 BLOCKING

Another experimental design principle that is heavily applied to microarray technology is blocking. Blocking is a way to reduce the effect of some varying factors that are identified but uninteresting. Experimental units are grouped into blocks. Within a block, the experimental units are homogeneous but across blocks the experimental units vary. The treatments or conditions are therefore compared within a block. Blocking was originated from field experiments in agriculture where different areas of the field differ in many aspects. To test crop varieties for yield, Fisher (1926) came up with a blocking strategy to divide the field into blocks within which the field was uniform. Different varieties were then planted and compared within each block. For blocking design, the number of experimental units in each block is called *block size*. The ideal situation is that the block size is equal to the number of varieties/treatments to be compared. However, sometimes the block size is less than the number of varieties/treatments to be compared. Depending on whether the block size is equal to or less than the number of treatments, different block designs are implemented.

9.4.1 Complete and Incomplete Block Design

9.4.1.1 Complete Block Design When the block size is the same as the number of treatments to be compared, we can compare all the treatments in each block. This design is called complete block design. For example, we have cages of mice with three mice in each cage and we want to compare three diets using these mice. If we treat the cage as a blocking factor, then the block size (3) is the same as the number treatments (3). We can compare all three diets in each cage and use multiple cages of mice to have replicates in our experiment. Complete block design does not make comparisons across blocks. In complete block designs, randomization is carried within each block.

9.4.1.2 Incomplete Block Designs When the block size is smaller than the number of treatments (or treatment combinations in multifactorial design) we want to compare, we have an incomplete block design. In this case, not all comparisons can be made within each block. In our mouse example above, if we have six instead of three diets to compare, we cannot compare all of them using one cage (three) mice. In this situation, we try to balance the treatments with the blocks by using an equal number of mice for each diet and pairing all six diets equally into cages which would result in a *balanced incomplete block design* (BIB). Suppose that we want to use 10 cages of mice, we could assign the six diets (1–6) into the 10 cages (C_1 to C_{10}) of mice. We would then have five replicates for each diet. We can arrange the treatments to the mice in the 10 cages as $C_1(1,2,5)$, $C_2(1,2,6)$, $C_3(1,3,4)$, $C_4(1,3,6)$, $C_5(1,4,5)$, $C_6(2,3,4)$, $C_7(2,3,5)$, $C_8(2,4,6)$, $C_9(3,5,6)$, $C_{10}(4,5,6)$. Constructing a BIB is a complicated mathematical problem. There are some designs and designing methods for constructing a BIB design (Kuehl, 2000; Cox, 1958; Yates, 1936a, b). The design of a BIB experiment is usually first constructed using code such as the

1–6 representing our six diets and C_1 to C_{10} representing the cages. The codes are then connected to the samples and treatments. Randomization needs to be carried out when assigning the six actual treatments to numbers 1–6 as well as assigning the treatments to the three mice in each cage.

9.4.2 Blocking in Microarray Technology

The application of blocking design has been fully explored in the two-color microarray platform because each array is a natural block of size 2 for any gene. From the very beginning of the microarray technology, investigators have fully realized the variability from spot to spot across arrays due to the variability in the loading of DNA quantity, spot morphology, hybridization condition, and scanning settings for the cDNA arrays. Therefore, they did not want to compare the intensity of one spot to that of another for the same gene on a different array. Instead, they paired two samples labeled with different dyes on one array to enable within-array comparison because the two channels of the same spot are more comparable (homogeneous). This feature of the two-color microarray fits right into a block design with a block size of 2 (Churchill, 2002; Kerr and Churchill, 2001a, b). When there are only two samples to compare, we have a complete block design. If we have more than two samples to compare, we have an incomplete block design.

The two dyes (Cy3 and Cy5) enable the comparison within a spot (block); however, each dye does have gene-specific bias. Some genes show a higher affinity to one dye than the other. When the two samples labeled with different dyes are compared, the higher signal intensity from one dye at a spot may not necessarily mean a higher level of expression of the corresponding gene in the sample labeled by that dye. It could come from that gene's higher affinity to that dye. It was estimated that about 40% of genes showed significant gene-specific dye bias (Dobbin et al., 2005). The gene-specific dye bias cannot be removed through whole-chip normalization; it can only be removed by balancing the dyes with respect to the treatments for testing the treatment effect. If we are interested in comparing two samples, we should label each sample with both dyes and pair the two samples of two arrays with dye labeling reversed. This pairing strategy is called dye swapping.

9.4.2.1 Reference Designs For complex experiments, the most intuitive design is the reference design where all the samples are labeled with one dye and compared to a universal reference labeled with the other dye (Fig. 9.2). The ratios of all samples to the reference sample are analyzed for treatment or condition effect. Dye bias is eliminated in this design because all the samples of interest are labeled with the same dye. The reference sample is often a universal reference or the pool of all other samples. However, it can be individual biological replicates from one (most often the baseline) condition such as in a time-series experiment discussed below. In this case, there will be different choices of schemes of reference designs that fit different experimental goals (Yang and Speed, 2002; Steibel and Rosa, 2005).

9.4.2.2 Loop Designs Although the use of a universal reference in the reference design helps to block out the effect of spots, it does not contribute information to the

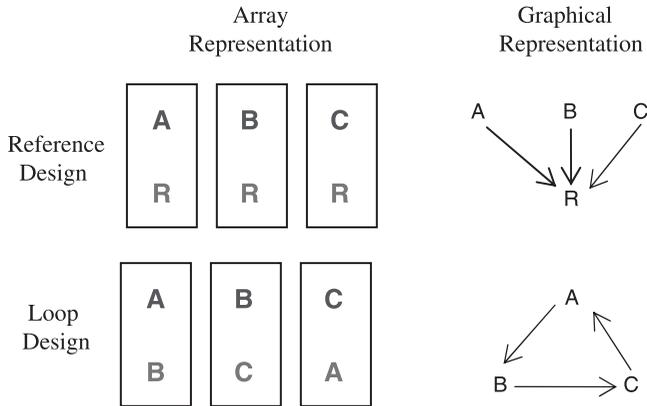


Figure 9.2 Array and graphical representations of reference design and loop designs for comparing three samples. In the array representation, the arrays are represented by rectangles. The samples used for hybridization on the array are indicated by letters in the rectangles. The dyes are represented by colors of the letters, green for Cy3 dye, red for Cy5 dye. In the graphical representation, the array is represented by arrow with head pointing to the Cy3 dye and tail pointing to the Cy5 dye. The samples are the nodes of the graphs. (See color insert.)

measurement of the treatment effect, which is often the goal of the experiment. Recognizing this inefficiency and the similarity in the blocking structure between microarray and conventional field trials, Kerr and Churchill (2001a) proposed the loop design for comparing multiple samples (Fig. 9.2). This design does not involve a reference sample; instead it just pairs the biological samples on arrays. To make all possible comparisons of all samples, they connected all the samples in a loop and also established a graphic representation of a microarray experiment using an arrow to represent an array with the head pointing to one dye and the tail pointing to the other. The loop design can balance the dyes and achieve a higher efficiency than a reference design when multiple samples are compared. If we have different treatments and multiple biological replicates for each treatment, constructing a reference design is straightforward but constructing a loop design can be complicated. As shown in Figure 9.3, one strategy is to use one loop for each set of biological replicates when there are equal numbers of biological replicates across treatments (Churchill, 2002; Altman and Hua, 2006).

Comparisons between reference and loop designs showed that both have advantages and disadvantages. The reference design is easy to handle in the laboratory and is easy to extend. The comparison between any two samples in a reference design is within two steps; therefore, the comparisons are equally efficient. The disadvantage of reference design is that it has half of the measurements wasted on the uninteresting sample; therefore, it is not very efficient. In contrast, the overall efficiency of a loop design is higher than that of a reference design (Kerr and Churchill, 2001a; Churchill, 2002; Yang and Speed, 2002; Kerr, 2003; Wit et al., 2005; Altman and Hua, 2006). For example, for the two designs shown in Figure 9.2, the average variance of the reference design is 2, but it is only 0.67 for the loop design (Yang and

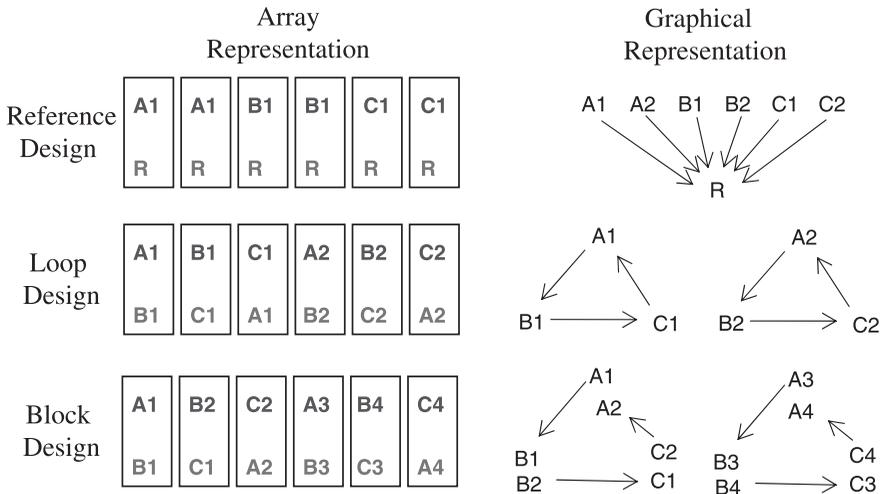


Figure 9.3 Array and graphical representations of designs with biological replicates. The three treatments are represented by three letters. Biological replicates within treatments are represented by the combination of letters and numbers. In the array presentation, arrays are represented by rectangles and dyes are represented by colors, green for Cy3 dye, red for Cy5 dye. In the graphical representation, arrays are represented by arrows with head pointing to the Cy3 dye and tail pointing to the Cy5 dye. For the same number of arrays, the balanced block design can accommodate four biological replicates while the other two designs can only accommodate two biological replicates. (See color insert.)

Speed, 2002). However, the efficiency of comparisons between different samples could be different in a loop design. When the loop gets larger, the efficiency of some comparison can be low (Kerr and Churchill, 2001a; Dobbin et al., 2003). In addition, it is rather complicated to construct a loop design to achieve optimal efficiency when there are multiple samples (Kerr and Churchill, 2001a; Wit et al., 2005; Altman and Hua, 2006). It is not obvious how to extend a loop design although there are ways to achieve it (Altman and Hua, 2006).

9.4.2.3 BIB Design Another design for complex experiment is the BIB design where each biological replicate is labeled once and samples from different treatments are paired on arrays with dyes balanced with respect to treatments (Fig. 9.3). Similar to the loop design, a reference sample is not used in the BIB design. The difference between a BIB and loop design is how many times a biological sample is labeled. In a block design, each sample is labeled once. The balance of dyes is achieved at the treatment level. In a loop design, each sample is labeled twice (by both dyes) and the balance of dyes is achieved at the sample level. Because balanced block design can incorporate more biological samples without using technical replicates for the same number of arrays, it can be more efficient in testing for the treatment effect than a loop design. However, block design cannot be used for classifying individual samples because the cross-array comparison is often not possible due to

the large variation (Dobbin and Simon, 2002; Dobbin et al., 2003). Loop design and reference design are good for classifying individual samples.

9.4.2.4 Other Potential Blocking Factors Besides the pairing of samples on the two-color microarrays, there are other applications of blocking in microarray experiments. For example, if we have to use arrays from different lots in the experiment, we can treat the array lot as a blocking factor by hybridizing an equal number of samples from all treatments to arrays from one lot and the rest of the samples to arrays from the other lot. If we cannot process all arrays in one batch, we can treat the batch as a block by processing the same number of arrays from all treatments in one batch and the leftovers in another batch. If we have two technicians working on the same experiment, we can treat the technician as a blocking factor and have each person process a full set of samples from all treatments. In these cases, the variation of array lots, processing batches, and technicians will be blocked out and the results will be less biased.

9.5 DESIGN FOR CLASSIFICATIONS

Microarray technology is also commonly used in clustering and classification studies, especially in the classification of tumors (Quackenbush, 2006; Simon, 2003). A clustering study is often used to examine the gene expression level across individual samples and to assess similarities or dissimilarities across genes and across individuals. Classification studies are used to predict the classes of samples based on the expression of some genes which are selected from the training set of samples. The experimental designs for these types of applications are different from those used for identifying differentially expressed genes across treatments/conditions/classes. In clustering and classification studies, there are often no biological replicates and individual samples are the main interest. In addition, there are often many more individual samples to be compared.

The design for clustering and classification is often very simple when a one-color microarray is used, just one array for each sample. For two-color microarrays, there is the consideration of different pairing schemes. Unlike the experiments to compare classes/treatments, the application of loop and block designs is limited in clustering and classification experiments (Dobbin and Simon, 2002). Reference design is the main design due to its practical advantage and extendibility. The loop design is applicable although it can be complicated and inefficient. The BIB design is not applicable at all due to the lack of dye balance and confounding with array effects when individual samples are considered.

Sample size calculation for a clustering study is rarely utilized because clustering is an exploratory and screening tool, the sample size often depending on the sample availability and budget. On the other hand, sample size calculation for the training set of a classification experiment is critical in order to establish the classifier genes (Dobbin et al., 2008; Dobbin and Simon, 2007; Mukherjee et al., 2003). It is quite different from sample size calculation in experiments aimed at identifying differentially expressed genes. The number of samples from each known class in the

training set depends not only on the fold change, accuracy, and type I error but also on the ratio of the two classes in the general population and the number of genes on the array (Dobbin et al., 2008; Dobbin and Simon, 2007). For a common classification experiment, 20–30 samples per class in the training set are probably sufficient. However, the required number of samples in each class could dramatically increase when there are no genes with large differences between the classes (Dobbin and Simon, 2007). There are some tools for the sample size calculation for the training set at <http://linus.nci.nih.gov/brb/samplesize/samplesize4GE.html>.

9.6 DESIGN FOR TIME COURSE EXPERIMENTS

Profiling gene expression at different times or developmental stage (time course experiment) is important to reveal the dynamics of gene expression. Many experiments have been conducted to reveal gene expression over time (Bar-Joseph, 2004). The design of a time-series microarray experiment has some similarity to the design of experiments for comparing different classes of samples. The reference design, loop design, and direct comparison are the building blocks of designs for a time-series experiment. However, selecting designs for a time course experiment depends more on interested comparisons. If the interested comparison is the consecutive time point in the series, direct comparison of neighboring points will be the most efficient use of the arrays (Yang and Speed, 2002). On the other hand, if comparison of the initial time point to all other time points is the most important comparison, direct comparison between the initial time point (as a reference) and the remaining points is beneficial. This design is very similar to a reference design except that the reference is of interest and biological replicates are desired for this reference. In addition, dye balance needs to be considered between this reference and other samples. This becomes an alternative reference design (Steibel and Rosa, 2005). If the interest is for all comparisons, alternative reference design, interwoven loop design, and the combination of both (carriage wheel design) are some choices (Khanin and Wit, 2005). The alternative reference design has the advantage of equal efficiency for any comparisons between the baseline point and the others or among the others, although it may be less efficient than other more direct comparisons in overall efficiency. The interwoven loop design uses multiple loops to connect the samples to avoid time points that are too far away in one loop. The carriage wheel design is especially suitable for experiments that are intended to compare the adjacent time points and the comparison between the initial point and all other points. This design uses the initial point as the reference and connects the rest of the time points consecutively into a loop (Khanin and Wit, 2005).

9.7 DESIGN FOR eQTL STUDIES

A more recent application of microarray is in the quantitative trait locus (QTL) mapping area. The conventional analysis of QTL is on phenotypic traits such as blood pressure and body weight, which are measured from each individual in a genetic segregating family of individuals resulting from crosses or pedigrees. In recent

years, microarray has been used to profile the gene expression of each individual in a QTL mapping population (de Koning and Haley, 2005; Gibson and Weir, 2005). The expression of each gene is treated as a quantitative trait for QTL mapping. The design of this type of experiment is more complex. It consists of design issues related to the QTL mapping aspect and design issues related to the microarray aspect. For the microarray aspect, it is simple if a one-color microarray platform is used. One array is often used for one individual. If a two-color microarray platform is used, the pairing of samples on each array can be mathematically studied. Although the type of QTL population can vary from study to study, such as model organism back cross, F_2 , recombinant inbred lines, or family pedigrees, the principles for pairing the samples on arrays are the same. The first thing to be clear about is the objective of the experiment and to identify the main interest effects to be mapped. The second is to pair the most dissimilar samples regarding the interested effect on the same array in a block design. The most dissimilar samples can often be identified based on the marker genotypes. The reference and loop designs are also applicable but are less efficient for mapping genetic factors controlling the expression of each gene (Bueno Filho et al., 2006; Fu and Jansen, 2006).

REFERENCES

- Allison, D. B., et al. (2002). A mixture model approach for the analysis of microarray gene expression data. *Computat. Statist. Data Anal.*, **39**: 1–20.
- Altman, N. S., and Hua, J. (2006). Extending the loop design for two-channel microarray experiments. *Genet. Res.*, **88**: 153–163.
- Bar-Joseph, Z. (2004). Analyzing time series gene expression data. *Bioinformatics*, **20**: 2493–2503.
- Bueno Filho, J. S., Gilmour, S. G., and Rosa, G. J. (2006). Design of microarray experiments for genetical genomics studies. *Genetics*, **174**: 945–957.
- Churchill, G. A. (2002). Fundamentals of experimental design for cDNA microarrays. *Nat. Genet.*, **32**(Suppl. 2): 490–495.
- Cox, D. R. (1958). *Planning of Experiments*. New York: Wiley.
- Cui, X., and Churchill, G. A. (2003). How many mice and how many arrays? Replication of cDNA microarray experiment. In *Methods of Microarray Data Analysis*, Vol. 3, M. L. Simon and T. A. Emily (Eds.). New York: Kluwer Academic Publishers.
- de Koning, D. J., and Haley, C. S. (2005). Genetical genomics in humans and model organisms. *Trends Genet.*, **21**: 377–381.
- Dobbin, K., Shih, J. H., and Simon, R. (2003). Statistical design of reverse dye microarrays. *Bioinformatics*, **19**: 803–810.
- Dobbin, K., and Simon, R. (2002). Comparison of microarray designs for class comparison and class discovery. *Bioinformatics*, **18**: 1438–1445.
- Dobbin, K. K., Kawasaki, E. S., Petersen, D. W., and Simon, R. M. (2005). Characterizing dye bias in microarray experiments. *Bioinformatics*, **21**: 2430–2437.
- Dobbin, K. K., and Simon, R. M. (2007). Sample size planning for developing classifiers using high-dimensional DNA microarray data. *Biostatistics*, **8**: 101–117.
- Dobbin, K. K., Zhao, Y., and Simon, R. M. (2008). How large a training set is needed to develop a classifier for microarray data? *Clin. Cancer Res.*, **14**: 108–114.
- Fisher, R. A. (1926). The arrangement of field experiments. *J. Ministry Agric. Great Br.*, **33**: 503–513.
- Fisher, R. A. (1935). *The Design of Experiments*. Edinburgh: Oliver and Boyd.
- Fu, J., and Jansen, R. C. (2006). Optimal design and analysis of genetic studies on gene expression. *Genetics*, **172**: 1993–1999.

- Gibson, G., and Weir, B. (2005). The quantitative genetics of transcription. *Trends Genet.* **21**: 616–623.
- Kendzioriski, C., Irizarry, R. A., Chen, K. S., Haag, J. D., and Gould, M. N. (2005). On the utility of pooling biological samples in microarray experiments. *Proc. Natl. Acad. Sci. U.S.A.*, **102**: 4252–4257.
- Kendzioriski, C. M., Zhang, Y., Lan, H., and Attie, A. D. (2003). The efficiency of pooling mRNA in microarray experiments. *Biostatistics*, **4**: 465–477.
- Kerr, M. K. (2003). Design considerations for efficient and effective microarray studies. *Biometrics*, **59**: 822.
- Kerr, M. K., and Churchill, G. A. (2001a). Experimental design for gene expression microarrays. *Biostatistics*, **2**: 183–201.
- Kerr, M. K., and Churchill, G. A. (2001b). Statistical design and the analysis of gene expression microarray data. *Genet. Res.*, **77**: 123–128.
- Khanin, R., and Wit, E. (2005). Design of large time-course microarray experiments with two channels. *Appl. Bioinformatics*, **4**: 253–261.
- Kuehl, R. O. (2000). *Design of Experiments: Statistical Principles of Research Design and Analysis*. Pacific Grove, CA: Duxbury.
- Lee, M. L., and Whitmore, G. A. (2002). Power and sample size for DNA microarray studies. *Stat. Med.*, **21**: 3543–3570.
- Mukherjee, S., et al. (2003). Estimating dataset size requirements for classifying DNA microarray data. *J. Comput. Biol.*, **10**: 119–142.
- Page, G. P., et al. (2006). The PowerAtlas: a power and sample size atlas for microarray experimental design and research. *BMC Bioinformatics*, **7**: 84.
- Pawitan, Y., Michiels, S., Koscielny, S., Gusnanto, A., and Ploner, A. (2005). False discovery rate, sensitivity and sample size for microarray studies. *Bioinformatics*, **21**: 3017–3024.
- Pritchard, C. C., Hsu, L., Delrow, J., and Nelson, P. S. (2001). Project normal: Defining normal variance in mouse gene expression. *Proc. Natl. Acad. Sci. U.S.A.*, **98**: 13266–13271.
- Quackenbush, J. (2006). Microarray analysis and tumor classification. *N. Engl. J. Med.*, **354**: 2463–2472.
- Shih, J. H., Michalowska, A. M., Dobbin, K., Ye, Y., Qiu, T. H., and Green, J. E. (2004). Effects of pooling mRNA in microarray class comparisons. *Bioinformatics*, **20**: 3318–3325.
- Simon, R. (2003). Diagnostic and prognostic prediction using gene expression profiles in high-dimensional microarray data. *Br. J. Cancer*, **89**: 1599–1604.
- Simon, R., Korn, E. L., McShane, L. M., Radmacher, M. D., Wright, G. W., and Zhao, Y. (2003). *Design and Analysis of DNA Microarray Investigations*. New York: Springer, pp. 11–35.
- Steibel, J. P., and Rosa, G. J. (2005). On reference designs for microarray experiments. *Stat. Appl. Genet. Mol. Biol.*, **4**: Article36.
- Warnes, G. R., and Liu, P. (2006). Sample size estimation for microarray experiments. Technical Report 06/06. University of Rochester, Department of Biostatistics and Computational Biology, Rochester, NY.
- Wit, E., Nobile, A., and Khanin, R. (2005). Near-optimal designs for dual channel microarray studies. *Appl. Statist.*, **54**: 817–830.
- Yang, Y. H., and Speed, T. (2002). Design issues for cDNA microarray experiments. *Nat. Rev. Genet.*, **3**: 579–588.
- Yates, F. (1936a). A new method of arranging variety trials involving a large number of varieties. *J. Agric. Sci.*, **26**: 424–455.
- Yates, F. (1936b). Incomplete randomized blocks. *Ann. Eugen.*, **7**: 121–140.
- Zakharkin, S. O., et al. (2005). Sources of variation in Affymetrix microarray experiments. *BMC Bioinformatics*, **6**: 214.
- Zhang, W., Carriquiry, A., Nettleton, D., and Dekkers, J. C. (2007). Pooling mRNA in microarray experiments and its effect on power. *Bioinformatics*, **23**: 1217–1224.

**STATISTICAL RESAMPLING
TECHNIQUES FOR LARGE
BIOLOGICAL DATA ANALYSIS**

Annette M. Molinaro and Karen Lostritto

Yale School of Public Health, Yale University, New Haven, Connecticut, USA

10.1 INTRODUCTION

The massive amounts of data generated by the advent of information-rich technologies provide an opportunity for deciphering complex biological processes. Among the most relevant circumstances where this applies is tumorigenesis, where cell types and activities are completely transformed. Microarray-based reagents are available for investigating RNA and protein expression, chromosomal amplification and deletions, epigenetic changes, DNA–protein interactions, and tissue histopathology. Each technology offers its own unique insight into underlying cellular processes and has the potential to revolutionize our understanding of carcinogenesis and cancer therapy. A translational link between bench and bedside exists in the ability to sort through large and complex datasets to arrive at mechanistic understandings of cancer that will lead to improved diagnostics or therapeutics.

Thus far, a common approach to extracting potential molecular markers from high-throughput technologies is to perform searches without identifying targets a priori, that is, without a hypothesis, and therefore labeled “discovery based.” In this scenario, one purpose is to identify a subset of targets for inference and future hypothesis testing, while a second purpose is to build a rule which discriminates groups of patients with different diagnoses, prognoses, or responses to therapy. Although the findings from such studies can be promising, as the following examples exhibit, discovery-based research can be severely limited by the analysis and study design or lack thereof, resulting in nonreproducible findings. One of the most significant reasons for nonreproducibility is due to overfitting (Ransohoff, 2004). Overfitting is to blame when perfect or nearly perfect discrimination can be found based solely on chance associations and, in turn, the results cannot be replicated in an independent dataset.

The most noted recent example of near-perfect discrimination was reported in a 2002 *Lancet* article that describes a cluster pattern of proteomic spectra generated by

mass spectroscopy with almost perfect classification of the study's 116 patients as having ovarian cancer or not, that is, 100% sensitivity and 95% specificity (Petricoin et al., 2002; Pollack, 2004). Given the lack of effective early detection of ovarian cancer, the findings gained national attention, and plans for a commercial test were soon underway. However, the test was subsequently delayed by the U.S. Food and Drug Administration due to questions concerning the reproducibility of the results.

In addition to others, researchers at the M. D. Anderson Cancer Center attempted to replicate the analysis on the publicly available data to no avail and concluded that the published method “performed no better than chance” (Baggerly et al., 2005; Ransohoff, 2005). In this example, the pattern for categorization was determined based on a particular group of patients and then used to classify the same patients (hereafter referred to as training set or resubstitution error). Had the findings been validated in an independent dataset, it would have been clear that the markers were not readily generalized to other groups of patients.

A second example of attaining overly optimistic results based on overfitting can be found on the front page of the December 19, 2002, edition of *The New York Times*, which reads, “Breast Cancer: Genes Are Tied To Death Rates” (Kolata, 2002). That week's *New England Journal of Medicine* published an article depicting the prognosis for several groups of women based on RNA expression profiles (van de Vijver et al., 2002). Of utmost importance was the ability of the profiling to separate women with a “poor” prognosis (e.g., only a 50% chance of survival at five years) from those with a “good” prognosis (90% survival at five years). However, it was subsequently determined that the authors had included some of the patients in the training set to evaluate the predictive ability of their findings, leading to an exaggerated estimate of the profiling accuracy. When a truly independent group of patients was used to evaluate the profiles, the difference in survival probability between the two groups dropped from the initial 40% to only 20%, signifying the effect of overfitting on prediction accuracy.

These two examples illustrate that building rules and assessing their predictive ability with the same data lead to results that are difficult to replicate in other samples or, more importantly, generalize to the broader population. In both examples, the outcome (ovarian cancer vs. no ovarian cancer and good vs. poor prognosis) is a categorical response. Regardless of whether the outcome is categorical or continuous, the most definitive way to demonstrate reproducible results is to evaluate the findings in a completely independent test set. However, due to small samples or insufficient study designs, this is not always feasible.

As described and demonstrated in this chapter, resampling methods provide us the tools to avoid overfitting when a completely independent test set is not available. In the following, we will outline estimation of prediction error with the goals of model selection and performance assessment as well as the most common techniques for resampling. Each will be demonstrated on a publicly available dataset for purposes of clarification and comparison. Analogous pseudocode can be found at the end of the chapter. Additionally, for the purposes of inference, we will also provide the resampling framework for ascertaining uncertainty in models and parameter estimation. As such, we will introduce the reader to the bootstrap and Markov chain Monte Carlo methods.

10.2 RESAMPLING METHODS FOR PREDICTION ERROR ASSESSMENT AND MODEL SELECTION

10.2.1 Prediction Error

In the prediction problem, we are interested in building and evaluating the performance of a rule or procedure fitted to n independent observations, corresponding to the n independent subjects in a study. Accordingly, we observe a random sample $X = \{x_1, \dots, x_n\}$, where $x_i = (y_i, z_i)$ contains an outcome y_i and a collection of p measured explanatory variables or features $z_i = (z_{i1}, \dots, z_{ip})'$ for the i th subject. For example, in microarray experiments, z_i includes RNA or protein expression, chromosomal amplification and deletions, or epigenetic changes, while in proteomic data, it includes the intensities at the mass-over-charge (m/z) values. The collection of features may also contain explanatory variables measured in the clinic and/or by histopathology such as a patient's age or tumor stage. The variables which constitute z can be measured on a continuous, ordinal, or categorical scale. The outcome y may be a continuous measure such as months to disease, a categorical or ordinal measure such as stage of disease, or a binary measure such as disease status.

The outcome y can be predicted from the collection of features z with the use of various statistical models. Subsequently a given prediction rule can be written as $r_x(z)$, where x reflects the dependence of the built rule on the observed random sample (see Fig. 10.1).

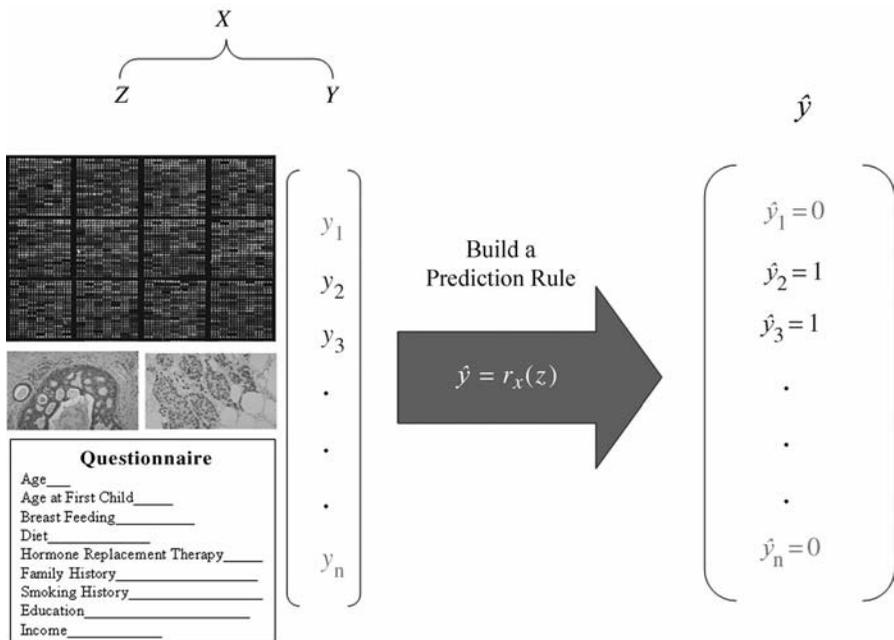


Figure 10.1 Building a prediction rule from the observed sample.

To assess how well a given rule predicts, loss functions may be employed. For each observation, a loss function $L(y_i, r_x(z_i))$ simply measures the error of the rule by determining the difference between what is predicted by the rule and the truth, that is, the difference between $r_x(z_i)$ and y_i . The choice of loss function is dependent on the scale of y . If y is continuous, frequent choices of $L(y_i, r_x(z_i))$ are as follows:

1. $(y_i - r_x(z_i))^2$, squared error.
2. $|y_i - r_x(z_i)|$, absolute error.

If y is binary, the most common choice is the indicator loss function, also referred to as the misclassification of the prediction rule:

$$L(y_i, r_x(z_i)) = I(y_i \neq r_x(z_i)).$$

Here the indicator function, $I(y_i \neq r_x(z_i))$, equals unity if the rule's prediction is incorrect and zero if it is correct. A more thorough discussion of loss function options is left for the end of the chapter, including the scenario when y is a time to event which is not always observed, that is, censored.

An average of the loss function over all observations in the sample can be calculated and is referred to as the *error rate* of a given rule. The quantity we are interested in estimating is the *true error* of a given rule, defined as the expected prediction error over an independent test set, $x_0 = (y_0, z_0)$. The *true error* of the rule r_x is

$$\text{Err} = E[L(y_0, r_x(z_0))]$$

where x is the observed random sample on which the prediction rule is built and E indicates the expectation, or average, over the independent test set.

10.2.2 Model Selection and Performance Assessment

There are two impetuses for evaluating the error rate of a prediction rule: model selection and performance assessment. As the number of explanatory variables included in a rule increases, so does the model complexity allowing the adaptation of the complex underlying structures in the data. Conversely, the more a model accommodates the underlying structure specific to the data it is built on, the less likely it will accurately predict an independent test set increasing the estimated error. Figure 10.2 illustrates this phenomenon. Hence, in model selection, the goal is to find the one rule or model which minimizes the error rate over a collection of potential models. Subsequent to model selection, in performance assessment, the goal is to estimate the error rate for a given rule, that is, assess how well it predicts the outcome of an observation not included in the observed data.

In an ideal setting, independent datasets would be available for the purposes of model selection and estimating the true error rate. In the next best scenario, we would have a substantial sample size in order to split the data into three parts: a training set to fit the model, a validation set to estimate the prediction error for the purpose of model selection, and a test set for assessment of the prediction accuracy of the final chosen model. Many factors influence how to divide the data into three sets, including the sample size, the signal-to-noise ratio in the data, the chosen algorithm for rule building, and the complexity of the rule. One split may allocate 50% to the training set and 25% each in the validation and test set. This scenario is depicted in Figure 10.3.

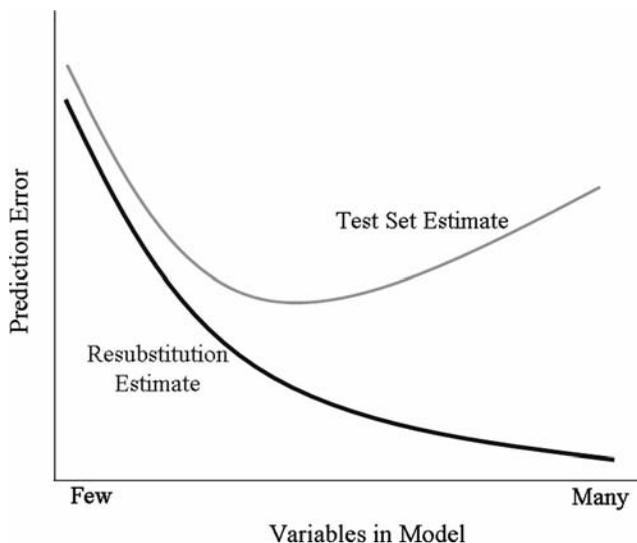


Figure 10.2 Effect of model complexity (as measured by number of variables in the model) on two different estimates of prediction error.

Frequently, however, we are faced with inadequate samples sizes to split the data into three sets and one must use the observed sample for model building, selection, and performance assessment. Again, the determination of how to allocate the data for each step is based on the aforementioned factors. The simplest method for estimating the prediction error rate is with the training set, known as the *resubstitution* or *apparent* error. Here all n observations are used for constructing, selecting, and, subsequently, evaluating the prediction error of the rule r_x . For example, with a continuous outcome y and squared-error loss function, the error rate of r_x is given as $n^{-1} \sum_{i=1}^n [y_i - r_x(z_i)]^2$. From the notation, we can observe that the prediction rule is built on the n observations

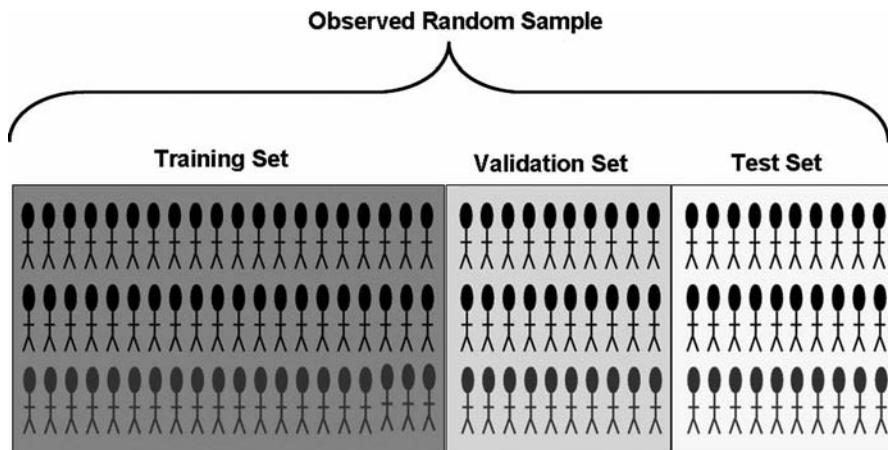


Figure 10.3 Large observed sample split into three sets: training for model building, validation for model selection, and test for assessment of prediction accuracy. (See color insert.)

and subsequently assessed with the same n observations. As a result, the resubstitution estimate tends to underestimate the true error (Efron, 1983; McLachlan, 1992). As can be seen in Figure 10.2, the resubstitution error estimate decreases with model complexity, potentially going to zero and indicating a complete overfit. And, also seen in Figure 10.2, as the estimated error drops, the generalization to the independent test set is increasingly worse.

To alleviate this biased estimation, resampling methods, such as cross-validation and bootstrapping, can be employed to more accurately estimate prediction error. In the next sections, these techniques are described as well as the implications of their use in the framework of model selection and performance assessment.

10.2.3 Resampling Framework

In the absence of a large, independent test set, there are numerous techniques for model selection and performance assessment achieved by implementing a form of partitioning or resampling of the original observed data. Each of these techniques involves dividing the data into a training set and a test set, as shown in layer 1 of Figure 10.4. For purposes of model selection, depicted in layer 2 of Figure 10.4, the

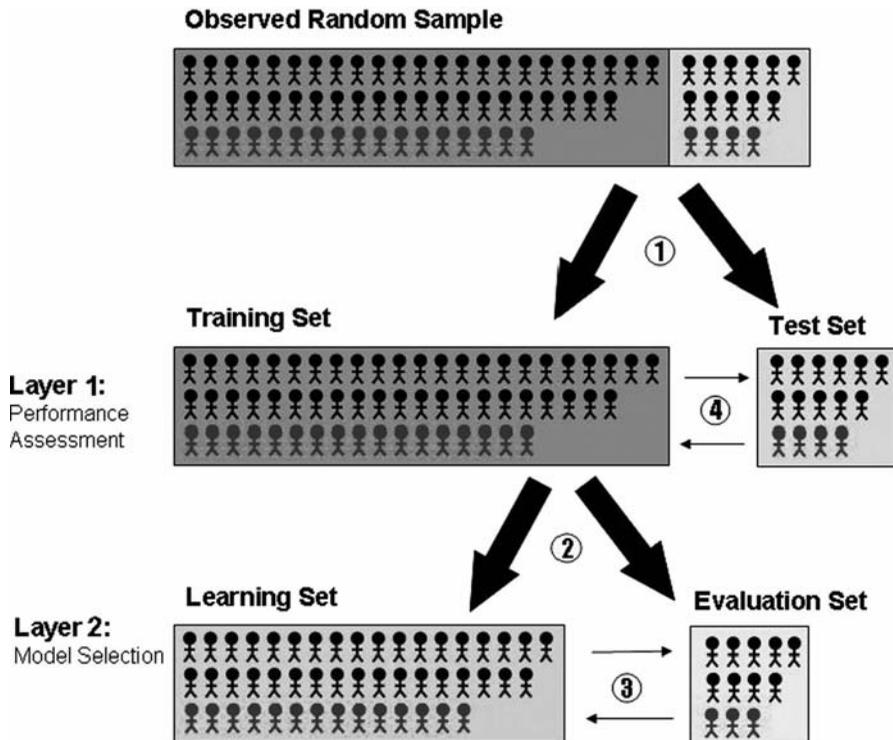


Figure 10.4 Flow chart for allocating data into training, learning, evaluation, and test sets. (See color insert.)

training set may further be divided into a learning set and an evaluation set (Molinari et al., 2004; Dudoit and van der Laan, 2005; Varma and Simon, 2006). Importantly, the interpretation of the results for either layer is applied to the intact observed random sample. For example, if a rule with six explanatory variables is chosen in the second layer and the corresponding prediction error is calculated as 0.38 in the first layer, the interpretation is that the best size model for the intact observed random sample is with six features and, for that model, the true error estimate is 0.38.

For ease of explanation, in the following descriptions of resampling methods, only the splitting of the observed sample into a training and test set will be discussed. However, the identical resampling methods and considerations can be used for further partitioning the training set into learning and evaluation sets, as depicted in layer 2 of Figure 10.4.

There are several considerations when selecting a resampling method. The foremost is the sample size n . Additional considerations are on the proportion of the observations reserved for the test set and the number of times the error estimate is calculated. We address these considerations in the following sections and also refer the reader to supplementary discussions in McLachlan (1992) and Davison and Hinkley (1997).

Two precautions are worth noting. First, regardless of the resampling method chosen, if the outcome is categorical and the prediction rule is built by a classification algorithm which relies on majority vote, for example, classification and regression trees (CART) or RandomForest (Breiman et al., 1984; Breiman, 2001), *there should be equal representation of each category of the outcome in the training/test and learning/evaluation sets*. This stratification will prevent a bias toward the most represented category or level of the outcome, particularly when majority vote is employed (Quackenbush, 2004).

Second, given the high-dimensional structure of each dataset, feature selection is typically an essential step before running any of the algorithms. Including too many noisy variables reduces accuracy of the prediction and may lead to overfitting of data, resulting in promising but often nonreproducible results (Ransohoff, 2004). Within each resampling method, feature selection must be based on the training set for prediction error and the learning set for model selection; otherwise additional bias is introduced. *This correct approach to feature selection within cross-validation has been referred to as honest or complete*. For additional discussions see Simon et al. (2003) and Quackenbush (2004). Pseudocode for estimating prediction error via cross-validation is included at the end of the chapter.

10.3 FEATURE SELECTION

In addition to improving prediction performance, feature selection offers the benefits of facilitating data visualization and interpretation as well as reducing computational intensity. Approaches to feature selection typically fall into one of the following three categories: filter methods, wrapper methods, or embedded methods. *Filter methods* encompass preprocessing techniques which rely on evaluating the predictor variables separately to assess their individual importance in explaining the outcome. As such, these methods ignore the combined effects, or synergism, of subsets of variables.

The appeal is the ease of computation and applicability. The resulting statistics or p -values for the chosen filter method are then ranked and a cutoff chosen to select the most significant features. Examples of filter methods are t -tests, Wilcoxon rank-sum or signed-rank tests, Pearson correlation estimates, log-rank tests, and univariate regression techniques such as linear, logistic, or Cox proportional hazards.

Wrapper methods, on the other hand, evaluate subsets of predictor variables with the intention of elucidating a meaningful understanding of the various predictors by taking interactions into account. A common approach is to also use the selected rule-building algorithm for delineating which variables to include. Examples of wrapper methods include the forward, backward, and stepwise variable-selection techniques in regression analysis. *Embedded methods* inherently perform feature selection as part of the user-selected rule-building algorithm and are typically specific to a chosen machine algorithm. Examples are classification and regression trees, CART (Breiman et al., 1984), and regularization techniques such as lasso (Tibshirani, 1996).

10.4 RESAMPLING-BASED CLASSIFICATION ALGORITHMS

For building rules, various algorithms are available based on the scale of y . If the outcome y is continuous, then the rule, or predictor, can be built via regression (linear and non-) or recursive binary partitioning such as CART (Breiman et al., 1984). Consequently, $\hat{y} = r_x(z_1, \dots, z_p)$ denotes the predicted outcome based on a function of the observed features in X . If the outcome is categorical or binary, it assumes one of K -values. The rule, or predictor, partitions the explanatory variable space into K disjoint and exhaustive groups G_k , where $k = 1, \dots, K$, such that $\hat{y} = k$ if the observed features fall into G_k . Standard statistical algorithms include linear discriminant analysis and diagonal discriminant classifiers, logistic regression, nearest neighbors (NN), neural networks, support vector machines, and CART as well as aggregate classifiers. Thorough discussions of available algorithms can be found in Breiman et al. (1984), McLachlan (1992), Ripley (1996), and Hastie et al. (2003).

10.5 PRACTICAL EXAMPLE: LYMPHOMA

The resampling methods presented in the following will be demonstrated on a lymphoma dataset. This publicly available microarray data set of Rosenwald et al. (2002) focuses on diffuse large-B-cell lymphoma. In this study there are 240 patients with 7399 genes measured on each microarray. For the purposes of this analysis, the outcome variable y is binary and represents the lymphoma subtype: $y = 0$ for activated B-cell and $y = 1$ for germinal-center B-cell. In this example, there is a moderate signal-to-noise ratio as the subgroups do not separate perfectly based on the microarray observations (Wright et al., 2003).

For feature selection, we used univariate t -statistics to identify the 10 most differentially expressed genes and k nearest neighbors (k -NN) to build the rule based

on these 10 genes. It is important to note that k , the number of neighbors, is inversely related to model complexity. That is, for small k , the rule may better accommodate the underlying data structure, while, for larger k , this will not hold true. As frequently encountered in research, there was not an independent test set to validate the results; thus, we will use the observed 240 patients for the purposes of model building, selection, and performance assessment.

To compare different resampling methods, five measures are reported: the prediction error for the test set; sensitivity, defined here as the proportion of germinal-center B-cell patients correctly classified as such; specificity, defined as the proportion of activated B-cell subtype patients correctly classified as such; positive predictive value (PPV), defined as the probability that a patient has germinal-center B-cell subtype given a classification as such; and the negative predictive value (NPV), defined as the probability that a patient has activated B-cell subtype given a classification as such. An estimate of the standard deviation is provided along with each of the five measures based on 20 repetitions of the resampling method.

10.6 RESAMPLING METHODS

10.6.1 Split Sample

Split sample is a popular resampling method, also referred to as the *learning-test split* or *holdout method* (McLachlan, 1992), which entails a single partition of the observed data into a training set and a test set. The size of the two sets is based on a predetermined proportion p for the test set. For example, if $p = \frac{1}{3}$, this method allots two-thirds of the observed data to the training set and one-third to the test set. Figure 10.5 illustrates this resampling method.

The main advantage of the split-sample method is ease of computation. In addition, since the prediction rule is developed only once, a completely specified algorithm for classifier development need not be available; thus, the development can be more informal and subjective. Regardless of the computational ease, there

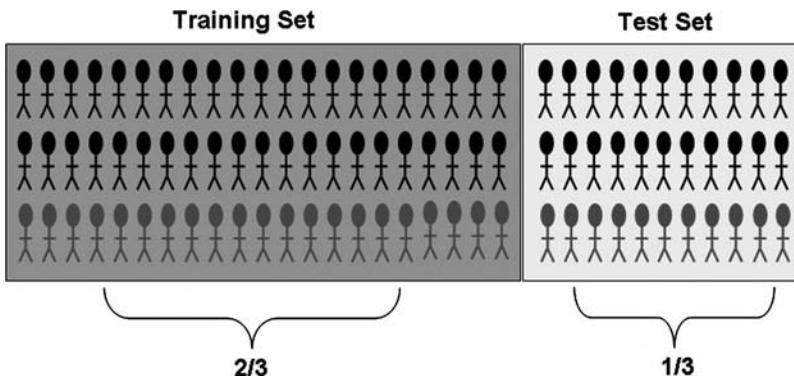


Figure 10.5 Depiction of split sample separated in a training ($2/3$ of observed data) and a test (remaining $1/3$) set. (See color insert.)

are two potential sources of bias inherent in this method. First, each individual contributes to only one of the two sets: the training or the test. Second, both feature selection and prediction rule construction are based on a small training set. Because the training set is smaller than the full data set, the corresponding test set error will tend to overestimate the unknown true error rate for a model or rule built on the full data set. This is especially true when the observed sample size is small. Consequently, the split-sample method is severely impaired by such a reduced training set size for feature selection, model fitting, and performance assessment.

To illustrate this approach, we split the 240 patients in the lymphoma dataset into a training and test set (once with two-thirds in the training set and once with half); subsequently each training set was split into a learning and evaluation set based on the same two-thirds training and one-third test set or half training and half test set split in order to perform model selection. For the k -NN algorithm, model selection entails selecting the value of k resulting in the minimum prediction error based on the evaluation set. Once k has been selected, the training set is reassembled, and k -NN is performed on the training set and evaluated with the test set using the selected value of k from the model selection step. The results for applying the split-sample approach to model selection and performance assessment in the lymphoma data set are in Table 10.1.

10.6.2 Cross-Validation Methods

Cross-validation (CV) is the most frequently used method for assessing prediction error. As mentioned previously, in the ideal setting we would have a large enough sample to set aside a portion for an independent test set to estimate the accuracy of our developed rule. When confronted with the smaller samples, we can use CV to build the rule on a subset of the data and assess it on an entirely different subset. Several different variations of CV are explained in the following.

10.6.2.1 v -Fold Cross-Validation In v -fold CV the n observations in the sample are randomly assigned to one of v partitions such that the partitions are of approximately equal size. As a result, for a total of v -folds, the training set contains all but one of the partitions, labeled the test set. Therefore, each partition acts as the test set exactly once. Figure 10.6 illustrates this resampling method. For each of the v -folds, the prediction rule is built on the training set, and the error rate is estimated on the single partition labeled as the test set. This is repeated v times, and the reported

TABLE 10.1 Lymphoma Data Results Using Split-Sample Approach for Model Selection and Performance Assessment

Proportion	Prediction					
	error	Sensitivity	Specificity	PPV	NPV	k -Value
$p = \frac{1}{2}$	0.154 (0.03)	0.918 (0.04)	0.912 (0.04)	0.906 (0.05)	0.924 (0.04)	5.4 (3.2)
$p = \frac{1}{3}$	0.161 (0.06)	0.909 (0.03)	0.899 (0.03)	0.892 (0.03)	0.915 (0.02)	6.3 (2.7)

Note: Numbers in parentheses are standard deviations.

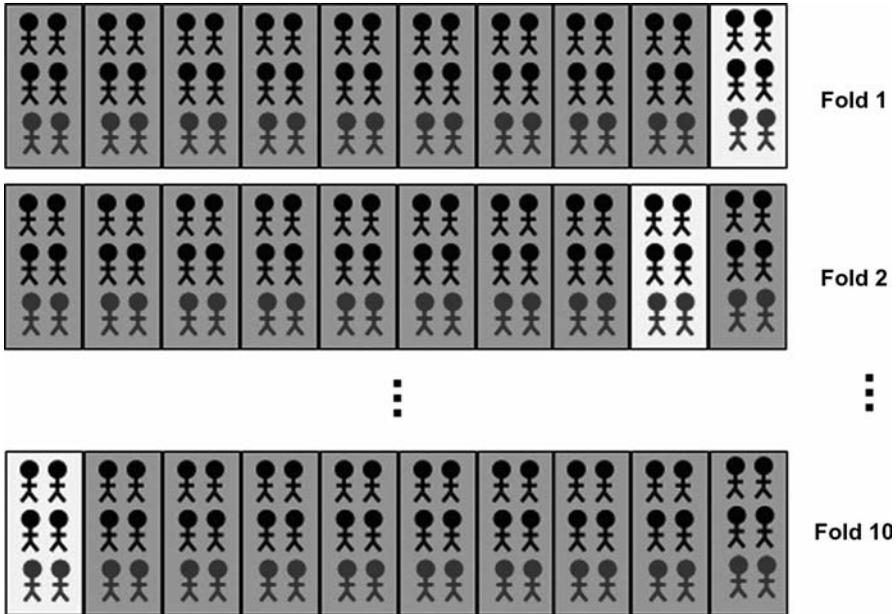


Figure 10.6 Depiction of training and test sets for the iterations of 10-fold CV. (See color insert.)

cross-validation error estimate is the average over the v test sets. The proportion of observations in each test set is approximately equal to $k = 1/v$, leaving approximately $1 - k$ in each training set.

Both k and v can adversely or positively affect this estimate of the true error. For example, a larger v (e.g., $v = 10$) results in a smaller proportion of k in the test set; thus, a higher proportion in the training set decreases the bias. An additional way to decrease the bias inherent in this estimate is by performing *repeated v -fold cross-validation*. As such, the entire process is repeated v times, each time initially assigning the observations to different partitions, implementing v -fold CV as described above and subsequently averaging the v error estimates.

For model selection, the prediction rule with the smallest CV error averaged over the v evaluation sets is labeled as the “best” model. Nonetheless, in practice a 1-SE rule is used, which suggests selection of the most parsimonious model whose error is no more than one standard error above that of the best model’s.

To illustrate this approach, we split the 240 patients in the lymphoma dataset into a training and test set for each of 2-fold, 5-fold, and 10-fold CV. Each training set was subsequently split into a learning and evaluation set based on the same v -fold in order to perform model selection. For the k -NN algorithm, model selection entails selecting the value of k resulting in the minimum prediction error based on the evaluation set. Once k has been selected, the training set is reassembled, and k -NN is performed on the training set and evaluated with the test set using the selected value of k from the model selection step. Additionally, we performed repeated CV, where v was set to 10, and the CV was repeated either 5 or 10 times.

TABLE 10.2 Lymphoma Data Results Using v -Fold CV Approach for Model Selection and Performance Assessment

Folds	Prediction					
	error	Sensitivity	Specificity	PPV	NPV	k -Value
$v = 2$	0.157 (0.02)	0.900 (0.02)	0.900 (0.02)	0.893 (0.02)	0.908 (0.02)	5.6 (2.0)
$v = 5$	0.147 (0.02)	0.884 (0.01)	0.894 (0.01)	0.885 (0.01)	0.894 (0.01)	7.2 (0.9)
$v = 10$	0.149 (0.01)	0.885 (0.01)	0.898 (0.01)	0.888 (0.01)	0.894 (0.01)	6.8 (0.8)
$v = 10, V = 5$	0.152 (0.01)	0.884 (0.01)	0.900 (0.01)	0.890 (0.01)	0.894 (0.01)	7.1 (0.5)
$v = 10, V = 10$	0.153 (0.01)	0.888 (0.00)	0.902 (0.01)	0.893 (0.01)	0.897 (0.00)	7.1 (0.2)

Note: Numbers in parentheses are standard deviations.

The results, displayed in Table 10.2, indicate that with a sample size of 240 the prediction error rates are very similar between the variations of v -fold CV, although a reduction in variability is achieved by using more folds. The number of neighbors k selected in two-fold CV is similar to that in a split sample with $p = \frac{1}{2}$; additionally, it is smaller than for the other folds of CV as well as repeated CV for which k ranges from 6 to 7. The estimates for sensitivity, specificity, PPV, and NPV are similar across the different folds of CV, although a reduction in variability is achieved by using repeated CV.

10.6.2.2 Leave-One-Out Cross-Validation Leave-one-out cross-validation (LOOCV) is the most extreme case of v -fold CV. In this method each observation is individually assigned to the test set, that is, $v = n$ and $k = 1/n$ (Lachenbruch and Mickey, 1968; Geisser, 1975; Stone, 1974, 1977). This method is depicted in Figure 10.7. LOOCV and the corresponding $k = 1/n$ represent the best example of a bias-variance trade-off in that it tends toward a small bias with an elevated variance. The reason for the high variance is due to the fact that the n test sets are so similar to one another. In model selection, LOOCV has performed poorly compared to

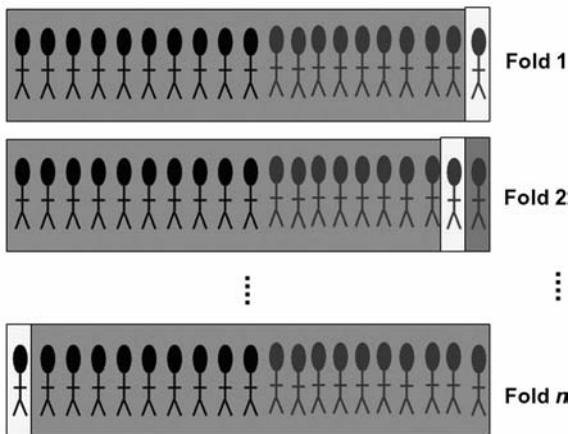


Figure 10.7 Depiction of training and test set splits for LOOCV. (See color insert.)

TABLE 10.3 Lymphoma Data Results Using LOOCV Approach for Model Selection and Performance Assessment

Folds	Prediction					
	error	Sensitivity	Specificity	PPV	NPV	k-Value
$v = n$	0.169 (0.01)	0.887 (0.00)	0.904 (0.00)	0.895 (0.00)	0.897 (0.00)	7.3 (0.13)

Note: Numbers in parentheses are standard deviations.

10-fold CV (Breiman and Spector, 1992). Due to the computational burden, LOOCV has not been a favored method for large samples and its behavior in estimating generalization error has not been thoroughly studied.

The results from implementing LOOCV in the lymphoma dataset are shown in Table 10.3. Leave-one-out cross-validation is the most computationally intensive method for this dataset, as for each of the 240 training\test set pairs, the training set, consisting of 239 observations, is then split into a learning set of 238 observations and an evaluation set of 1 observation for a total of 239 times in order to perform model selection.

Generalized Cross-Validation Of the cross-validation variations, LOOCV is appealing due to its inherent low bias; however, its variance is high and it is computationally expensive. One alternative to reduce the computational complexity while achieving a low bias is to implement *generalized cross-validation* (GCV) for approximating LOOCV. GCV can be used when the outcome of interest is continuous, y , and the squared error loss function is chosen to estimate the error rate of r_x . The GCV approximation is given by

$$\text{GCV} = \frac{1}{n} \sum_{i=1}^n \left(\frac{y_i - r_x(z_i)}{1 - k/n} \right)^2$$

where n is the sample size and k is the number of parameters, or coefficients, estimated by the algorithm.

10.6.2.3 Monte Carlo Cross-Validation The last variation is frequently referred to as Monte Carlo cross-validation (MCCV). Unlike the previous forms of CV, this approach repeatedly and randomly splits the sample into a training and test set, for example, 20, 50, or 1000 iterations (see Table 10.4). For each split

TABLE 10.4 Lymphoma Data Results Using the Monte Carlo CV Approach for Model Selection and Performance Assessment

Iterations	Prediction error	Sensitivity	Specificity	PPV	NPV	k-Value
20	0.155 (0.02)	0.883 (0.01)	0.900 (0.01)	0.890 (0.01)	0.893 (0.01)	6.8 (0.5)
50	0.154 (0.01)	0.886 (0.01)	0.901 (0.01)	0.892 (0.01)	0.895 (0.01)	7.2 (0.3)
100	0.153 (0.01)	0.887 (0.01)	0.891 (0.01)	0.883 (0.01)	0.896 (0.01)	7.6 (0.2)

Note: Numbers in parentheses are standard deviations.

$nk = n(1/v)$ of the observations are labeled as the test set and $n(1 - k) = n(1 - 1/v)$ as the training set. For example, in MCCV with $v = 10$ each of 50 iterations allot 10% of the data to the test set and 90% to the training set. The error rate is assessed for each of the 50 test sets and subsequently averaged over the 50 iterations. As the number of iterations increases, the computational burden of MCCV is quite large. However, unless the iterations of random splits approach infinity, the chance that each observation is included in a training set and a test set (over all iterations) is small, introducing a similar bias to that of the split-sample approach (i.e., when each observation is either in the training set or test set).

Monte Carlo CV was performed on the lymphoma dataset resulting in error rates similar to the other methods. The k -values are similar to the other methods except for split-sample and two-fold CV, which had smaller values for k .

10.7 BOOTSTRAP METHODS

10.7.1 Classical Bootstrapping Technique

Several variations of the bootstrap have been introduced to estimate prediction error. The fundamental bootstrap approach is based on randomly drawing a sample *with replacement* from the observed sample of size n (Efron, 1983; Efron and Tibshirani, 1993). The random sample, due to the replacement, is also of size n . The random sampling is repeated B times, resulting in B bootstrap samples. For any given draw, approximately one-third of the observations are not selected (i.e., $0.368n$) and serve as the test set, also known as the out-of-bag (OOB) test set. As such, the training set has approximately $0.632n$ unique observations and $(1 - 0.632)n$ repeated observations. Subsequently, the rule-building algorithm can be applied to each of the B bootstrap datasets (in effect B training sets) and the prediction error assessed using the corresponding B test sets, that is, those observations not included in the matching training set due to sampling with replacement. This measure of prediction error is referred to as the *leave-one-out bootstrap estimate*.

The major advantage of the leave-one-out bootstrap estimate of prediction error is the large training set size, n . When confronted with small sample sizes, as frequently encountered in genomics, there are obvious limitations to how well one can approximate the true error, and, as such, the leave-one-out bootstrap is very appealing. However, due to the fact that the number of distinct observations in each training set is approximately equal to $0.632n$, this estimate suffers a training set size bias. This bias, similar to two-fold CV, results in an overestimation of the prediction error.

10.7.2 0.632+ Bootstrap

Two estimators have been suggested to correct the bias observed in the leave-one-out bootstrap: the 0.632 bootstrap and the 0.632+ estimator (Efron and Tibshirani, 1997). Both corrections adjust the overestimated leave-one-out bootstrap error, denoted err_{LOOB} , by adding the underestimated resubstitution error, denoted err_{RSE} . This

TABLE 10.5 Lymphoma Data Results Using Bootstrap Approaches for Model Selection and Performance Assessment

Bootstrap	Prediction					
	error	Sensitivity	Specificity	PPV	NPV	<i>k</i> -Value
LOOB	0.185 (0.01)	0.801 (0.01)	0.829 (0.01)	0.811 (0.01)	0.820 (0.01)	1 (0)
0.632+	0.146 (0.01)	0.801 (0.01)	0.829 (0.01)	0.811 (0.01)	0.820 (0.01)	1.1 (0.08)

Note: Numbers in parentheses are standard deviations.

combination is achieved by weighting the two error estimates and summing:

$$\omega \times \text{err}_{\text{RSE}} + (1 - \omega) \times \text{err}_{\text{LOOB}}$$

where ω is a weighting scheme.

The difference between the 0.632 and 0.632+ bootstrap is the assigned weight ω . For the 0.632 bootstrap the weight ω is constant ($\omega = 0.632$) and relates to the aforementioned limitation that approximately $0.632n$ of the training set is in fact unique. The 0.632 estimate works well when there are few features and corresponding parameters. In a more intense model-building scenario, where overfitting could occur, the 0.632+ estimate is the better alternative. In the 0.632+ bootstrap estimate, ω is determined based on the “no-information error rate,” defined as the prediction error estimate if the independent variables, or features z , are independent of the outcome, y (Efron, 2004). A relative overfitting rate can be computed by comparing the difference between the resubstitution estimate and the leave-one-out bootstrap estimate to that of the no-information error rate and leave-one-out bootstrap estimate. This rate as well as the noted number of unique observations is combined to compute the weight, ω . As such, the 0.632+ prediction error estimate balances the overestimation of the leave-one-out bootstrap with the underestimation of the resubstitution estimate by accounting for the amount of overfitting.

We utilized both the leave-one-out bootstrap as well as the 0.632+ alternative to estimate the error rate of k -NN in the lymphoma dataset. In Table 10.5, LOOB’s overestimation of the error rate is quite obvious compared to the previous methods. The 0.632+ estimator correction is also apparent. Both have a significantly lower value for k compared with other methods. A k -value of 1 indicates that the best method of predicting the lymphoma subtype for a given observation is to look at the subtype of the one closest observation in terms of the genetic expression values.

10.8 SAMPLE SIZE ISSUES

Thus far, as seen in the lymphoma dataset example, the difference in prediction error estimation between methods is trivial. In part, this is due to the large observed sample size of $n = 240$. The effect of sample size on estimation and model selection becomes evident when we restrict the sample to a smaller number of patients. As before, for the

purposes of this analysis, the outcome variable y is binary and represents the lymphoma subtype: $y = 0$ for activated B-cell of which there are 115 and $y = 1$ for germinal-center B-cell of which there are 125. For $R = 1000$ repetitions, a random sample of 20 patients stratified by subtype status is selected from the 240 patients. The stratification allows for equal representation of both subtypes.

Each random sample plays two roles. First, it serves as a sample from which to calculate each resampling method's estimate of prediction error, as though we only observed 20 patients total. Second, it serves as a training set while the remaining 220 observations serve as the test set for an approximation of the true error. For feature selection, we used univariate t -statistics to identify the 10 most differentially expressed genes and k -NN to build the rule based on these 10 genes.

Given the 20 patients in the observed data set and the 220 in the independent test set, we can calculate the resampling error estimate (err). This process is repeated for $R = 1000$ repetitions, and the different resampling error estimates are compared to the true error and thus to each other using the mean-squared error (MSE) and bias, calculated as

$$\text{MSE} = \frac{1}{R} \sum_{r=1}^R (\text{err}_r - \text{Err}_r)^2 \quad \text{and} \quad \text{Bias} = \frac{1}{R} \sum_{r=1}^R (\text{err}_r - \text{Err}_r)$$

The resampling techniques under consideration are split sample with $p = 1/2$ and $p = 1/3$, that is, with one-half and one-third of the observations in the test set; v -fold CV with $v = 2, 5, 10$; LOOCV; repeated v -fold CV with $v = 10$; MCCV with 20, 50, and 100 iterations where the test set contains 10% of the observations and the training set contains 90%; and, lastly, the 0.632+ bootstrap. Table 10.6 presents the average estimated error (over $R = 100$ repetitions) for each resampling method, the standard deviation of this estimated error, the bias and MSE, as well as the average number of neighbors, k , selected by the learning/evaluation level for model selection.

It is clear from Table 10.6 that, although a split sample with $p = \frac{1}{3}$ is preferable to one with $p = \frac{1}{2}$, in small samples those two, as well as two-fold CV, have the largest

TABLE 10.6 Comparison of Resampling Methods in Small-Sample Scenario

Method	Estimate	Standard deviation	Bias	MSE	k
Split $p = \frac{1}{3}$	0.245	0.174	0.027	0.032	2.79
Split $p = \frac{1}{2}$	0.283	0.155	0.065	0.031	1.44
Two-fold CV	0.284	0.122	0.067	0.022	1.66
Five-fold CV	0.234	0.122	0.017	0.016	3.22
Ten-fold CV	0.225	0.126	0.008	0.017	3.15
MCCV $V = 20$	0.220	0.128	0.004	0.017	3.60
LOOCV	0.239	0.138	0.022	0.019	2.97
Repeated 5V	0.246	0.101	0.029	0.012	3.47
0.632+	0.236	0.087	0.020	0.010	3.59

standard deviation, MSE, and bias of all methods. The small k chosen for two-fold CV and split sample with $p = \frac{1}{2}$, is due to the reduced training set size. For v -fold CV, a significant decrease in prediction error, bias, and MSE is seen as v increases from 2 to 10. Tenfold CV has a slightly decreased error estimate compared to LOOCV as well as a smaller standard deviation, bias, and MSE; however, the LOOCV k is smaller than that of 10-fold CV. Repeated 5-fold CV decreases the standard deviation and MSE over 5-fold CV; however, values for the bias and k are slightly larger. In comparison to 10-fold CV, the 0.632+ bootstrap has a smaller standard deviation and MSE with a larger prediction error, bias, and k .

10.9 LOSS FUNCTIONS

As mentioned previously, the choice of a loss function is dependent on the scale of y . Several options, detailed in the following, are listed in Table 10.7. For continuous outcomes, a frequently used loss function is the squared error loss, $L(y_i, r_x(z_i)) = [y_i - r_x(z_i)]^2$. The error rate of the given prediction rule using the squared error loss function is the average over all observations, $n^{-1} \sum_{i=1}^n [y_i - r_x(z_i)]^2$. With a categorical outcome, a popular choice is the indicator loss function, $I(y_i \neq r_x(z_i))$, with corresponding error rate: $n^{-1} \sum_{i=1}^n I(y_i \neq r_x(z_i))$. This error rate is commonly referred to as the *misclassification* of a prediction rule.

If there are more than two levels of a categorical outcome, for example, if the outcome is the stage of disease and there are four levels indicating progression, it may not be realistic to assume the same loss for misclassifying a stage I observation into the closely related stage II versus the more dire stage IV. As such, one could use a loss function which incorporates differential misclassification costs (Breiman et al., 1984). Then the loss, or cost, of misclassifying the observation is written as $C(r_x(z_i)|y_i) \geq 0$ when $y_i \neq r_x(z_i)$, while the loss equals zero if the classification is correct, that is, $C(r_x(z_i)|y_i) = 0$ if $y_i = r_x(z_i)$. The user can define the repercussion for varying levels of misclassification. In the example of four stages, one scheme would be to assign a loss of 1 if the rule predicts an observation to be within one stage of that which was observed and 2 if it is more than one stage. Thus, the

TABLE 10.7 Loss Function Examples

Outcome	Name	$L(y_i, r_x(z_i))$
Continuous	Squared error or L_2	$[y_i - r_x(z_i)]^2$
	Absolute error or L_1	$ y_i - r_x(z_i) $
Categorical	Misclassification or indicator	$I(y_i \neq r_x(z_i))$
	Differential misclassification	$C(r_x(z_i) y_i) \geq 0, y_i \neq r_x(z_i)$ $C(r_x(z_i) y_i) = 0, y_i = r_x(z_i)$
	Log-likelihood or cross-entropy or deviance	$-2 \log p_y(x)$
Censored survival times	Inverse proportional censoring weighted	$[t_i - r_x(z_i)]^2 \frac{I(\delta_i = 1)}{P(\delta_i = 1)}$

rule which misclassifies a stage I observation as stage II receives a loss of 1 while a rule which misclassifies a stage I observation as a stage IV receives a loss of 2. The corresponding error rate for differential misclassification is $n^{-1} \sum_{i=1}^n C(r_x(z_i)|y_i)$.

Another option when y is categorical is to base the loss function on the probability that y takes a specific response value. If the outcome is categorical or binary, it assumes one of K values. The rule, or predictor, partitions the explanatory variable space into K disjoint and exhaustive groups G_k , where $k = 1, \dots, K$, such that $\hat{y} = k$ if the observed features fall into G_k . We can model the probabilities $p_k(x) = \Pr(\hat{y} = k|x)$. The corresponding loss function is $-2 \sum_{k=1}^K I(\hat{y} = k) \log p_k(x) = -2 \log p_{\hat{y}}(x)$, which is called the log-likelihood and frequently referred to as cross-entropy or deviance. The associated error rate is $-2n^{-1} \sum_{i=1}^n \log \hat{p}_{y_i}(x_i)$.

Frequently in medical studies the outcome y is a time to a particular event, for example, time to initial occurrence of disease, recurrence of disease, or death from disease. If researchers were able to observe all patients until the outcome of interest, for example, recurrence of prostate cancer, all survival times would be observed and on a continuous scale. However, this is rarely the case. Instead, by the end of a study, some patients may have dropped out, been lost to follow-up, not had the particular event of interest, or died from a competing cause. In this situation, the last date of follow-up or date at end of study is recorded and referred to as a *censored* time to event.

To differentiate between an always observed, continuous outcome y and a censored, continuous time to event outcome, we will refer to the latter as t_i and introduce an indicator of event, δ_i . Here $\delta_i = 1$ if observation i 's outcome time, t_i , is the time of the event of interest, for example, if observation i had a prostate cancer recurrence at time t_i ; and $\delta_i = 0$ if observation i 's outcome time, t_i , is censored, for example, if observation i is lost to follow-up. Robins and Rotnitzky (1992) introduced the *inverse probability of censoring weighted* (IPCW) loss function for this scenario, which we can write as $[t_i - r_x(z_i)]^2 [I(\delta_i = 1)/P(\delta_i = 1)]$. Thus, we can weight the squared error loss for the continuous outcome by a ratio with numerator equal to whether or not observation i had the event of interest, $I(\delta_i = 1)$, and denominator equal to the probability that observation i had an event of interest, $P(\delta_i = 1)$. The corresponding error rate for censored outcomes is the IPCW loss function averaged over all n observations.

10.10 BOOTSTRAP RESAMPLING FOR QUANTIFYING UNCERTAINTY

In addition to selecting models and calculating the corresponding prediction error, we are frequently interested in evaluating the uncertainty inherent in the estimation of a rule's parameters as well as that due to choosing a model based on data-adaptive methods such as CV. In high-dimensional data settings, such as genomics, it is typically difficult if not impossible to analytically assess this variability. Therefore, we must rely on methods of resampling the observed data to quantify the uncertainty of our model and its parameters.

Bootstrap resampling provides two approaches: nonparametric and parametric bootstrap samples. In both we begin with a rule or procedure fitted to n independent

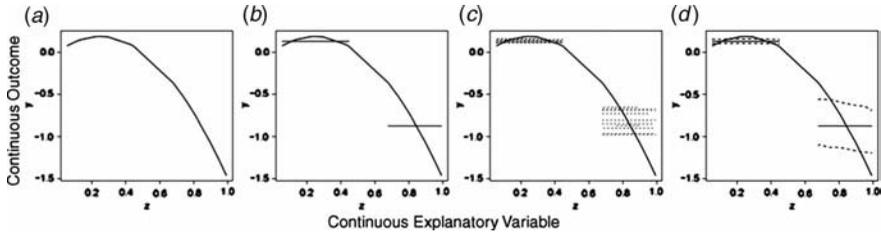


Figure 10.8 (a) Plot of observed data. (b) Simple linear regression fit to data. (c) Linear regression fit to 10 bootstrap samples. (d) Linear regression fit in solid lines with 95% confidence band based on 400 bootstrap samples in dashed lines.

observations, corresponding to the n independent subjects in a study. Accordingly, we observe a random sample $X = \{x_1, \dots, x_n\}$, where $x_i = (y_i, z_i)$ contains an outcome y_i and a collection of p measured explanatory variables, or features, $z_i = (z_{i1}, \dots, z_{ip})'$, for the i th subject. For the sake of explanation, we will focus on the observed data shown in the left-most plot of Figure 10.8. As can be seen in the plot, the data contains a single, continuous feature, that is $z_i = (z_{i1})$, and a continuous outcome, y_i .

One approach to exploring the relationship between the explanatory variable, z , and outcome, y , is to fit a simple linear regression model using an indicator function. This model can be written as $E(Y|Z = z) = \alpha + \beta I(z > 0.5)$, where $E(Y|z)$ is the expected value (i.e., mean) of the outcome y conditional on the feature value of z , $I(z > 0.5)$ is the indicator function which equals 1 when $z > 0.5$ and 0 otherwise, α is the intercept, and β is the slope. The fit of this model to the data is shown in solid lines in the second plot in Figure 10.8. In this example, the intercept α is the line on the left, which is the expected value of y when $z < 0.5$, and the line on the right, the expected value of y when $z > 0.5$, which can be written as $\alpha + \beta$.

10.10.1 Nonparametric Bootstrap

In the first bootstrap method, the actual observed data are sampled as opposed to being derived from a specified parametric model. Due to the model-free nature of this resampling, it is referred to as the nonparametric bootstrap. As such, B samples of size n are drawn *with replacement* from the original data set, also of size n , where the unit drawn is the observed $x_i = (y_i, z_i)$. For each of the B bootstrap samples, in our example, we fit the simple linear regression model using the indicator function, $I(z > 0.5)$, resulting in a slightly different estimate of the slope and intercept for each sample. Ten such estimates derived from $B = 10$ bootstrap samples are shown in the third plot in Figure 10.8.

As B increases, the bootstrap samples capture the variability of the estimates for α and $\alpha + \beta$. For example, if we set $B = 400$, we can extract a 95% confidence interval for the estimates by selecting the 0.025 and 0.975 quantiles of the bootstrap estimates for each z -value. The 95% confidence band estimated from $B = 400$ bootstrap samples is illustrated by dashed lines in the fourth plot of Figure 10.8, where the solid lines represent the simple linear regression fit. The difference in variability for estimation of α compared to $\alpha + \beta$ is clearly visible from the plot. The 95%

confidence band around α is narrow and consistent, signifying accuracy in estimation of α . On the other hand, the 95% confidence band around $\alpha + \beta$ is relatively wide, indicating less precision in estimation, and, additionally, by slanting downward, reveals the inverse correlation between y and z for values of $z > 0.5$.

10.10.2 Parametric Bootstrap

In the second bootstrap method, instead of sampling with replacement from the original data, a rule is built based on the explanatory variables with the aim of predicting the outcome, \hat{y} . Subsequently, a parametric model is assumed for the purpose of adding noise to the predicted values of y , resulting in y^* . This creates a new sample which includes the original, unaltered explanatory variable(s), z , and the new outcome y^* such that $x_i^* = (y_i^*, z_i)$. The entire procedure is repeated B times, resulting in B bootstrap samples. Similar to the nonparametric bootstrap, for each bootstrap sample, the algorithm is performed and then confidence intervals ascertained.

For example, we can build $B = 400$ parametric bootstrap samples, each by adding noise to the model for predicting \hat{y} from the last example. We assume the noise, denoted ε_i , has a Gaussian distribution with mean zero and variance $\hat{\sigma}^2 = \sum_{i=1}^n (y_i - \hat{y}_i)^2 / n$, written as $\varepsilon_i \sim N(0, \hat{\sigma})$. Then we can write $y_i^* = \hat{\alpha} + \hat{\beta}I(z_i > 0.5) + \varepsilon_i$, where $\hat{\alpha}$ and $\hat{\beta}$ are the maximum-likelihood estimates of the intercept and slope, respectively. As B increases, the parametric bootstrap samples capture the variability of the estimates for α and $\alpha + \beta$. For example, as with the nonparametric bootstrap, with $B = 400$, we can extract a 95% confidence interval for the estimates by selecting the 0.025 and 0.975 quantiles of the bootstrap estimates for each z -value.

10.11 MARKOV CHAIN MONTE CARLO METHODS

Another approach to quantifying uncertainty in inference is to use Bayesian methods. When analytic estimation is infeasible or extremely inefficient, as frequently encountered with high-dimensional data, Bayesian methods have proven to be extremely effective. In general, the goal is to make use of probability models both for the observed data and for those unknown parameters for which we would like to make inference.

Given an observed random sample $X = \{x_1, \dots, x_n\}$, where $x_i = (y_i, z_i)$ contains an outcome y_i and a collection of p measured explanatory variables, or features, $z_i = (z_{i1}, \dots, z_{ip})'$, for the i th subject, there are three fundamental steps to Bayesian data analysis. First, we specify a sampling probability model and a prior distribution for the unknown parameters. The sampling probability model is the conditional probability of the observed data, X , given the unknown parameters, Θ , written as $\Pr(X | \Theta)$. The prior distribution for the parameters, $\Pr(\Theta)$, provides what information, or knowledge, we have about the parameters before we actually observe the data. As these probability models are defined, it is important that they are consistent with both the underlying biological information as well as the data collection process.

Second, we calculate the posterior distribution, $\Pr(\Theta | X)$. This distribution reflects what we learn about the unknown parameters once we have seen the data. Additionally, it will be used to predict the future *unobserved* outcome y_0 based on the corresponding measured features z_0 . A substantial advantage to using the posterior distribution to predict future values as opposed to using the maximum-likelihood values is that the former accounts for uncertainty in estimating the unknown parameters. In the third step, we evaluate how well the model fits the data, how the assumptions affect the results, and how reasonable the conclusions are.

In order to evaluate the uncertainty in the model, we focus on understanding the posterior distribution. This is done by calculating summary measures (e.g., mean, median) as well as sampling from it. In simple Bayesian models, sampling from the posterior distribution is fairly effortless. However, as the complexity of the model increases, so does the posterior distribution. *Markov chain Monte Carlo* (MCMC) methods, including the Gibbs sampler and Metropolis–Hastings algorithms, provide a general process for iteratively sampling values of the parameters, Θ , from approximate posterior distributions and then continually updating the sampling to better approximate the true posterior distribution, $\Pr(\Theta | X)$ (Gelfand and Smith, 1990).

The key to these methods is that the values are sampled sequentially, such that the distribution only depends on the last value sampled. Therefore, a Markov chain is formed. As defined in probability, a Markov chain is a sequence of random variables $\Theta_1, \Theta_2, \dots, \Theta_K$, for which the distribution of any given Θ_k , given all previous Θ 's, depends only on the most recent value Θ_{k-1} . Therefore, for each step in a simulation, the approximate distributions improve, eventually converging on the true distribution.

Importantly, after any of the methods are implemented and the samples drawn, the convergence of the simulated sequences must be verified. This process may be computationally expensive and, therefore, entails altering the algorithms to be more efficient (Gelman et al., 2003). In the discussion, we provide relevant examples; here, we describe the two most popular MCMC methods.

10.11.1 Gibbs Sampler

The Gibbs sampler is a Markov chain algorithm which is exceedingly useful in multi-dimensional problems (Geman and Geman, 1984; Casella and George, 1992). As it is frequently difficult to sample from the joint distribution of the parameters, the Gibbs sampler employs a conditional sampling of each parameter (or subsets thereof) given the rest. Importantly, it is not necessary to explicitly state the form of the conditional densities only to be able to sample from them. The Markov chain produced by the Gibbs sampler is a stationary distribution which is, in fact, the true joint distribution of the parameters.

Given a set of parameters, $\Theta = \Theta_1, \dots, \Theta_K$, we begin by initiating each parameter with a starting value. Then for each iteration $t = 1, 2, \dots$, the Gibbs sampler will alternately sample each parameter conditional on the value of all the others. Thus, due to the K parameters, there are K steps in each iteration, t . We can write this as $\Pr(\Theta_k | \Theta_1^t, \dots, \Theta_{k-1}^t, \Theta_{k+1}^{t-1}, \dots, \Theta_K^{t-1}, X)$, where the superscript denotes which parameters have already been updated in iteration t (e.g., Θ_1^t) and those that have not (e.g., Θ_{k+1}^{t-1}). This process of alternately sampling continues until the joint distribution of the

parameters does not change. The alternate sampling can be sequential, in random order, or in groups rather than one by one. Subsequently, by applying a density estimate to the observed sample values, we can approximate the marginal density of any parameter.

The simplest MCMC algorithm, the Gibbs sampler, is particularly practical when the conditional distribution can be computed effortlessly or when the parameters take on values from a small set. Of additional importance, it is trivial to verify that the Gibbs sampler leads to a stable distribution.

10.11.2 Metropolis and Metropolis–Hastings Algorithms

The Metropolis and, more general, Metropolis–Hastings algorithms provide a family of Markov chain simulation methods for sampling from the posterior distribution (Metropolis et al., 1953; Hastings, 1970; Chib and Greenberg, 1995). Both algorithms generate random perturbations of the current parameter values and then employ an acceptance/rejection rule to converge to the true joint distribution. In fact, they are adaptations of a random walk.

Given a set of parameters, Θ , we begin by initiating either algorithm with a starting point Θ^0 , for which $\Pr(\Theta^0 | X) > 0$, from a starting distribution $\Pr(\Theta^0)$. This starting distribution may be based on a crude estimate or approximation. Then for each iteration $t = 1, 2, \dots$, the algorithm will first sample a proposal Θ^* from a jumping distribution, $J_t(\Theta^* | \Theta^{t-1})$. In the Metropolis algorithm the jumping distribution must be symmetric; however, in the Metropolis–Hastings algorithm this restriction is relaxed. The second step of each iteration is the calculation of a ratio of densities. In the Metropolis algorithm, the ratio is $r = \Pr(\Theta^* | X) / \Pr(\Theta^{t-1} | X)$. In the Metropolis–Hastings algorithm, the ratio corrects for the asymmetric jumping rule by rewriting r as a ratio of ratios:

$$r = \frac{\Pr(\Theta^* | X) / J_t(\Theta^* | \Theta^{t-1})}{\Pr(\Theta^{t-1} | X) / J_t(\Theta^{t-1} | \Theta^*)}$$

In the third step, the updated value(s), Θ^t , are assigned as Θ^* with probability equal to the minimum of r and 1 and, Θ^{t-1} otherwise. Thus, the algorithm always accepts steps that increase the density and occasionally those that decrease it. The advantage to the asymmetric jumping rules in the Metropolis–Hastings algorithm is an increase in speed.

10.12 CONCLUSIONS

10.12.1 Classical Resampling Methods for Prediction Error Assessment and Model Selection

The immediate goal in prediction is to assess a rule constructed with the information collected in the random sample X for the subsequent prediction of a future *unobserved* outcome y_0 based on its corresponding measured features z_0 . Frequently, we would like to compare the ability of rules constructed from different algorithms to correctly predict the outcome of these unknown observations. In the process of constructing

each rule, unknown parameter values must be estimated or tuning parameters selected; for example, in the lymphoma data study, we must choose k , the optimal number of nearest neighbors, for the k -NN algorithm. In an ideal world, we would have a training set to build a rule and a large, independent test set with which to validate. Unfortunately, however, this large independent dataset is oftentimes not available or feasible and we are left to perform model selection and prediction error estimation based on a single, possibly small, observed dataset.

Validating a rule with the same data used to build it underestimates the true error rate. Therefore, we must rely on resampling methods, which repeatedly partition the given dataset into training and test sets for the purpose of more accurately estimating the true error rate. As depicted in Figure 10.4, an additional layer of resampling may also be implemented for model selection. Estimation of prediction error when confronted with a multitude of covariates and small sample sizes is a relatively new problem. Inherent in this estimation are the difficulties which arise due to feature selection and unknown signal-to-noise ratios. To ascertain the “best” approach given this setting, it is important to compare various resampling methods over several samples sizes, algorithms, and data sets. The various resampling methods described within this chapter are listed in Table 10.8. For the purposes of comparing the different

TABLE 10.8 Pros and Cons of Different Resampling Methods for Prediction Error Assessment and Model Selection

Method	Pros	Cons
Split sample	<ul style="list-style-type: none"> • Computationally simple • Informal development of rule 	<ul style="list-style-type: none"> • Observed only in one set • FS and PE based on small training set leads to overestimate of TE
v -Fold CV	<ul style="list-style-type: none"> • Computationally inexpensive • Ten fold has small MSE • Ten fold best when FS is included 	<ul style="list-style-type: none"> • Two-fold CV reduces accuracy
Repeated v -fold	<ul style="list-style-type: none"> • Low standard deviation • Low MSE and bias 	<ul style="list-style-type: none"> • Computationally expensive
LOOCV	<ul style="list-style-type: none"> • If weak signal, low bias • Small MSE • Best when FS is included 	<ul style="list-style-type: none"> • Computationally expensive • If strong signal, high upward bias
MCCV	<ul style="list-style-type: none"> • Allows random selection of training and test sets. 	<ul style="list-style-type: none"> • Computationally expensive • Not enough gain over 10-fold CV to warrant use • Only have inclusion in training and test sets if iterations $\rightarrow \infty$
Bootstrap	<ul style="list-style-type: none"> • Training set sample size same as that for observed data 	<ul style="list-style-type: none"> • If low signal, underestimate TE
0.632+ bootstrap	<ul style="list-style-type: none"> • If weak signal, low bias • Small MSE 	<ul style="list-style-type: none"> • If strong signal, overestimate TE • Small standard deviation

FS, Feature Selection; PE, Prediction Error; TE, True Error.

methods, we only used the k -NN algorithm; however, many other algorithms such as diagonal linear discriminant analysis (DLDA), linear discriminant analysis (LDA), support vector machines (SVM), and CART have been studied in the literature.

Considerations when choosing a resampling method include sample size, the proportion p of observations for the test set, and the averaging of the estimate. In selecting p there is an inherent bias–variance trade-off. Whereas, the choice of a larger p results in a higher bias and lower variance, a small proportion p will typically result in a low bias but a high variance for the prediction error estimate. A third consideration is the number of times the estimate is calculated. There may be an advantage of smoothing due to averaging multiple estimates.

The following conclusions are drawn from the simulations and the literature:

- The split-sample and two-fold CV resampling methods are appealing due to their simple nature and low computational complexity. However, in small sample sizes, accuracy is sacrificed. It is clear that although a split sample with $p = \frac{1}{3}$ is preferable to one with $p = \frac{1}{2}$, in small samples those two, as well as two-fold CV, have the largest standard deviation, MSE, and bias of all methods (see Table 10.6). The bias of the split-sample and two-fold methods is large for small samples because the reduced size of the training set severely impairs the ability to effectively select features and fit a model. The large bias causes large MSE. This general trend of improved performance with a smaller proportion of observations reserved for the test set has been repeatedly observed elsewhere (e.g., see Martin and Yu, 2006). The poor performance of two-fold CV is also consistent when studied using additional classification algorithms such as DLDA and CART (Burman, 1989; Molinaro et al., 2005).
- With regard to k -NN in small samples, there is no one resampling method which is superior to all others based on the measures of standard deviation and bias. If we focus solely on MSE (which incorporates both bias and standard deviation), repeated CV performs the best. However, when we examine the bias and standard deviation, repeated CV has a slightly higher bias (in absolute value) compared to other methods although its standard deviation is the lowest. This inability to identify a universally superior method due to a bias–variance trade-off was also found in a study involving both 1-NN and DLDA (Jiang and Simon, 2007).
- In our simulation, the 0.632+ bootstrap performance was hindered by its bias. Although its standard deviation is comparable to other methods, its bias is significantly higher than other methods. We would expect bootstrap to suffer from problems with bias given that its error rate formula contains the resubstitution error estimate, which is known to have high bias because of the overlap between the training and test set. Bootstrap was also found to have a significant bias when using DLDA, although, in this case, the error rate underestimated the true error rate, whereas in our k -NN study, there was an overestimate of the true error rate (Jiang and Simon, 2007). Bootstrap only seems to perform comparably well to CV upon reaching a sample size of 500, and therefore, our conclusions regarding the 0.632+ bootstrap are not as favorable as those found by other studies (Efron, 1983; Efron and Tibshirani, 1997).

- As the sample size increases, the differences between the resampling methods decrease. As such, with small samples, it is important to be vigilant in the selection of a resampling method. This same phenomenon has been found in the literature (Molinaro et al., 2005).
- There are large differences regarding computational time between the resampling methods, frequently an important consideration. If computational time is an issue, 5- or 10-fold CV is optimal in that it runs relatively quickly and has comparable MSE, standard deviation, and bias to more computationally intensive methods. Especially for large sample sizes, the performance of 5-fold CV is similar to LOOCV (Breiman and Spector, 1992; Molinaro et al., 2005).
- In consideration of signal-to-noise ratios in the data, several interesting results have been shown in the literature. As the signal decreases from a strong signal to more noise, the discrepancies between the methods diminish (Molinaro et al., 2005). Nonetheless, when there is low or weak signal, either the LOOCV or 0.632+ estimate is suggested due to the relatively low bias. For strong signal, LOOCV has a high upward bias, and although a reduction in bias is seen in 0.632+, it still tends to overestimate the true error. For a thorough discussion, see Jiang and Simon (2007).
- The nested CV procedure reduces the bias considerably and gives an estimate of the error that is very close to that obtained on the independent testing set for both shrunken centroids and SVM classifiers (Varma and Simon, 2006).

10.12.2 Bootstrap Resampling for Quantifying Uncertainty

Typically, for many algorithms, the number of parameters and specific values for parameters, including indicator function cutoffs, may be selected via CV. Other algorithms may require the specification of tuning parameters similarly chosen via data-adaptive methods. Subsequently, the variability associated with the parameters (e.g., in the plot in Figure 10.8*b* the model's parameters are α and β) depends on both the observed data and the data-adaptive method for choosing the cutoff or tuning parameters. Frequently this is analytically intractable. Fortunately, there are two bootstrap resampling approaches which allow us to quantify this variability: nonparametric and the parametric bootstrap.

10.12.3 Markov Chain Monte Carlo Methods

In high-dimensional data, analytic estimation is frequently infeasible or extremely inefficient; as such, Bayesian methods provide an extremely effective alternative to maximum-likelihood estimation. The Bayesian approach to modeling and inference entails initially specifying a sampling probability model and a prior distribution for the unknown parameters. The prior distribution for the parameters reflects our knowledge about the parameters before the data are observed. Next, the posterior distribution is calculated. This distribution includes what we learn about the unknown parameters once we have seen the data. Additionally, it is employed to predict future observations.

By studying the posterior distribution, we are able to evaluate the uncertainty in the model. In simple Bayesian models, sampling from the posterior distribution is fairly straightforward. However, as the complexity of the model increases so does the posterior distribution. The MCMC methods, including the Gibbs sampler and Metropolis–Hastings algorithms, provide a framework for iteratively sampling values of the parameters from approximate posterior distributions to better approximate the true posterior distribution.

In the bioinformatics literature, there are numerous relevant examples. For example, Husmeier and McGuire (2002) implement MCMC methods to detect recombination by modeling the sequence of phylogenetic tree topologies along the multiple-sequence alignment of recombinant breakpoints. Additionally, Liu et al. (1995) employ the Gibbs sampler for multiple local sequence alignment. For clustering microarray expression data, McLachlan et al. (2002) apply a mixture model approach, while Sheng et al. (2003) use the Gibbs sampler. Additionally, Segal et al. (2003) employ MCMC methods for modeling gene expression.

10.12.4 Pseudocode for Prediction Error Estimation via Cross-Validation

The following pseudocode demonstrates an algorithm for estimating the prediction error from the CV resampling method performed on a training set. This estimated prediction error is compared to a “true” prediction error on a test set via bias and MSE measures. To make these comparisons, we calculate the estimated prediction error and true prediction error many times, each on a different training set/test set split of the dataset. Although this algorithm describes the CV resampling method, similar algorithms can be constructed for the other resampling methods described in this chapter. As explained in Section 10.2.3, it is vital for the training set and test set to have the same proportion of cases and controls, and this condition is accounted for below. While this algorithm assumes a binary outcome, modifications can be made in order to keep equal proportions of more than two categorical outcome classes in the training and test set and in each CV fold. For a continuous outcome, this balance is not an issue.

The code presented below portrays a general scenario; however, in our analysis, we chose the rule-building algorithm as k -NN and the feature selection method as a t -test between cases and controls for each variable. The inner “for” loop performs CV on the learning set in order to select model parameters, for example, the best number of neighbors, k , in the k -NN algorithm.

```
//v-fold cross-validation on dataset with binary outcome
SET  $n$  to number of observations in dataset
SET  $p$  to number of variables in dataset
SET train.size to the desired training set size
SET num.vars to the desired number of variables to pre-select before rule-
    building algorithm
SET  $q$  to proportion of controls (outcome=0) in dataset
```

```

SET numReps to number of desired iterations on resampling method
READ IN dataset
SET data.x to dataset' s nxp predictor matrix
SET data.y to dataset' s binary outcome y of length n.

INITIALIZE resamp.error to an empty vector of length numReps
INITIALIZE test.error to an empty vector of length numReps

FOR i from 1 to numReps

//Set up the training and test set such that each contains q proportion of controls
SET train.controls to random sample of train.size* q indices sampled
    from the control observation indices (i.e. data.y equal to 0)
SET train.cases to random sample of train.size* (1-q) indices sampled
    from the case observation indices (i.e. data.y equal to 1)

//training.set consists of components training.set.x and training.set.y
//training.set.x is a matrix with train.size rows and p columns
// training.set.y is a binary outcome vector y of length train.size.

SET training.set.x to observations in data.x with indices in train.controls
    or train.cases
SET training.set.y to observations in data.y with indices in train.controls
    or train.cases

//test.set consists of components test.set.x and test.set.y
//test.set.x is a matrix with (n-train.size) rows and p columns
// test.set.y is a binary outcome vector of length n-train.size.

SET test.set.x to observations in data.x with indices not in train.controls
    or train.cases
SET test.set.y to observations in data.y with indices not in train.controls
    or train.cases

//Set up v-fold cross-validation
SET cross.val to { REPEAT integers 1 to v UNTIL length train.size is
    reached}

// Shuffle cross.val vector (of length train.size) to randomly assign each observation to
// one of v-folds. Shuffling is done such that each fold consists of equal numbers of cases
// and controls.
SHUFFLE the first q* train.size elements of cross.val (corresponding to
    controls)
SHUFFLE the last (1-q)* train.size elements of the cross.val
    (corresponding to cases) .
INITIALIZE error.rate to empty vector of length v

```

```

FOR j from 1 to v
  SET learning.set.x to the training.set.x observations with cross.val
    not equal to j
  SET learning.set.y to the training.set.y observations with cross.val
    not equal to j

  SET evaluation.set.x to the training.set.x observations with
    cross.val equal to j
  SET evaluation.set.y to the training.set.y observations with
    cross.val equal to j

  SET parameters/model size selection to those which minimize
    average error over inner v-fold cross-validation FOR loop
    of the learning set (details not shown here)
  SET sig.vars to those num.vars variables out of the p variables in
    the learning.set.x which are most significant in feature
    selection tests to distinguish cases and controls in
    learning.set.y
  SET learning.set.x to learning.set.x with variables limited to
    sig.vars
  SET evaluation.set.x to evaluation.set.x with variables limited to
    sig.vars

  //Rule building algorithm uses learning set's x and y component to build model and then
  //uses the evaluation set's x component to make predictions.
  SET predicted.cv.y to the output from rule-building algorithm
    taking input learning.set.x, learning.set.y, and
    evaluation.set.x
  SET jth element of error.rate vector to the output from the chosen
    loss function comparing predicted.cv.y to evaluation.set.y
END FOR

//Take mean of error.rate which contains v error rates from cross-validation
SET ith element of resamp.error to average of error.rate

//Now use entire training set and test set to obtain "true" error rate
SET parameters/model size selection to those which minimize average
  error over inner v-fold cross-validation FOR loop of training set
  (details not shown here)
SET sig.vars to those num.vars variables out of the p variables in
  training.set.x which are most significant in feature selection tests
  to distinguish cases and controls in training.set.y
SET training.set.x to training.set.x with variables limited to sig.vars
SET test.set.x to test.set.x with variables limited to sig.vars
SET predicted.y to output from rule-building algorithm taking input

```

```

training.set.x, training.set.y, and test.set.x
SET  $i^{\text{th}}$  element of test.error to the output from the chosen loss function
    comparing predicted.y to evaluation.set.y
END FOR

SET estimate to average of resamp.error over the numReps iterations
SET standard.deviation to standard deviation of resamp.error over the numReps
    iterations.
SET bias to sum of (resamp.error - test.error) divided by numReps
SET mean.squared.error to sum of (resamp.error - test.error)2 divided by
    numReps

```

REFERENCES

- Baggerly, K. A., et al. (2005). Signal in noise: Evaluating reported reproducibility of serum proteomic tests for ovarian cancer. *J. Nat. Cancer Inst.*, **97**(4): 307–309.
- Breiman, L. (2001). Random forests. *Machine Learning*, **45**: 5–32.
- Breiman, L., Friedman, J. H., Olshen, R. A., and Stone, C. J. (1984). *Classification and Regression Trees*. Monterey, CA: Wadsworth and Brooks/Cole.
- Breiman, L., and Spector, P. (1992). Submodel selection and evaluation in regression. The X-random case. *Int. Stat. Rev.*, **60**: 291–319.
- Burman, P. (1989). A comparative study of ordinary cross-validation, v-fold cross-validation and the repeated learning-testing methods. *Biometrika*, **76**: 503–514.
- Casella, G., and George, E. I. (1991). Explaining the Gibbs sampler. *Am. Stat.*, **26**: 16–174.
- Chib, S., and Greenberg, E. (1995). Understanding the Metropolis–Hastings Algorithm. *Am. Stat.*, **49**(4): 327–335.
- Davison, A. C., and Hinkley, D. V. (1997). *Bootstrap Methods and Their Application*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge: Cambridge University Press.
- Dudoit, S., and van der Laan, M. J. (2005). Asymptotics of cross-validated risk estimation in model selection and performance assessment. *Stat. Methodol.*, **2**(2): 131–154.
- Efron, B. (1983). Estimating the error rate of a prediction rule: Improvement on cross-validation. *J. Am. Stat. Assoc.*, **78**: 316–331.
- Efron, B. (2004). The estimation of prediction error: Covariance penalties and cross-validation. *J. Am. Stat. Assoc.*, **99**: 619–642.
- Efron, B., and Tibshirani, R. J. (1993). *An Introduction to the Bootstrap*. Monographs on Statistics and Applied Probability, Vol. 57. New York: Chapman & Hall.
- Efron, B., and Tibshirani, R. J. (1997). Improvements on cross-validation: The .632+ bootstrap method. *J. Am. Stat. Assoc.*, **92**: 548–560.
- Geisser, S. (1975). The predictive sample reuse method with applications. *J. Am. Stat. Assoc.*, **70**: 320–328.
- Gelfand, A., and Smith, A. (1990). Sampling based approaches to calculating marginal densities. *J. Am. Stat. Assoc.*, **85**: 398–409.
- Gelman, A., et al. (2003). *Bayesian Data Analysis*. Boca Raton, FL: Chapman & Hall.
- Geman, S., and Geman, D. (1984). Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Trans. Pattern Anal. Machine Intell.*, **6**: 721–741.
- Hastie, T., Tibshirani, R., and Friedman, J. (2003). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer Series in Statistics, 1st ed. New York: Springer.
- Hastings, W. K. (1970). Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, **57**(1): 97–109.
- Husmeier, D., and McGuire, G. (2002). Detecting recombination with MCMC. *Bioinformatics*, **18**: S345–S353.

- Jiang, W., and Simon, R. (2007). A comparison of bootstrap methods and an adjusted bootstrap approach for estimating the prediction error in microarray classification. *Stat. Med.*, **26**: 5320–5334.
- Kolata, G. (2002). Breast cancer: Genes are tied to death rates. *New York Times*, December 19.
- Lachenbruch, P. A., and Mickey, M. R. (1968). Estimation of error rates in discriminant analysis. *Technometrics*, **10**: 1–11.
- Liu, L. S., Neuwald, A. F., and Lawrence, C. E. (1995). Bayesian models for multiple local sequence alignment and Gibbs sampling strategies. *JASA*, **90**: 1156–1170.
- Martin, R., and Yu, K. (2006). Assessing performance of prediction rules in machine learning. 5 Jun 2006. 7 Oct 2009 <<http://www.futuremedicine.com/doi/abs/10.2217/14622416.7.4.543>>.
- McLachlan, G. J. (1992). *Discriminant Analysis and Statistical Pattern Recognition*. New York: Wiley.
- McLachlan, G. J., Bean, R. W., and Peel, D. (2002). A mixture model-based approach to the clustering of microarray expression data. *Bioinformatics*, **18**(3): 413–422.
- Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., and Teller, E. (1953). Equation of state calculation by fast computing machines. *J. Chem. Phys.*, **21**(6): 1087–1092.
- Molinaro, A. M., Dudoit, S., and van der Laan, M. J. (2004). Tree-based multivariate regression and density estimation with right-censored Data. *J. Multivariate Anal.*, **90**: 154–177.
- Molinaro, A. M., Simon, R., and Pfeiffer, R. M. (2005). Prediction error estimation: A comparison of resampling methods. *Bioinformatics*, **21**: 309–313.
- Petricoin, E. F., et al. (2002). Use of proteomic patterns in serum to identify ovarian cancer. *Lancet*, **359**: 572–577.
- Pollack, A. (2004). New cancer test stirs hope and concern. *New York Times*, February 3.
- Quackenbush, J. (2004). Meeting the challenges of functional genomics: From the laboratory to the clinic. *Preclinica*, **2**: 313–316.
- Ransohoff, D. F. (2004). Rules of evidence for cancer molecular marker discovery and validation. *Nat. Rev. Cancer*, **4**: 309–313.
- Ransohoff, D. F. (2005). Lessons from controversy: Ovarian cancer screening and serum proteomics. *JNCI*, **97**: 315–319.
- Ripley, B. D. (1996). *Pattern Recognition and Neural Networks*. Cambridge: Cambridge University Press.
- Robins, J. M., and Rotnitzky, A. (1992). Recovery of information and adjustment for dependent censoring using surrogate markers. *Aids Epidemiology, Methodological Issues*: 297–331.
- Rosenwald, A., et al. (2002). The use of molecular profiling to predict survival after chemotherapy for diffuse large-B-cell lymphoma. *N. Engl. J. Med.*, **346**: 1937–1946.
- Segal, E., Taskar, B., Gasch, A., Friedman, N., and Koller, D. (2001). Rich probabilistic models for gene expression. *Bioinformatics*, **17**: S243–S252.
- Segal, E., Wang, H., and Koller, D. (2003). *Discovering Molecular Pathways from Protein Interaction and Gene Expression Data*. Oxford: Oxford University Press.
- Sheng, Q., Moreau, Y., and De Moor, B. (2003). Biclustering microarray data by Gibbs sampling. *Bioinformatics*, **19**(Suppl. 2): II196–II205.
- Simon, R., Radmacher, M. D., Dobbin, K., and McShane, L. M. (2003). Pitfalls in the use of DNA microarray data for diagnostic and prognostic classification. *J. Nat. Cancer Inst.*, **95**: 14–18.
- Stone, M. (1974). Cross-validated choice and assessment of statistical predictions. *J. R. Stat. Soc. Ser. B*, **36**: 111–147.
- Stone, M. (1977). Asymptotics for and against cross-validation. *Biometrika*, **64**: 29–35.
- Therneau, T., and Atkinson, E. (1997). An introduction to recursive partitioning using the RPART routine. Technical Report 61. Section of Biostatistics, Mayo Clinic, Rochester, NY.
- Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *J. R. Stat. Soc. B*, **58**(1): 267–288.
- van de Vijver, M. J., et al. (2002). A gene expression signature as a predictor of survival in breast cancer. *N. Engl. J. Med.*, **347**(25): 1999–2009.
- Varma, S., and Simon, R. (2006). Bias in error estimation when using cross-validation for model selection. *BMC Bioinformatics*, **7**: 91.
- Venables, W. N., and Ripley, B. D. (1994). *Modern Applied Statistics with S-PLUS*. New York: Springer-Verlag.
- Wright, G., Tan, B., Rosenwald, A., Hurt, E. H., Wiestner, A., and Staudt, L. M. (2003). A gene expression-based method to diagnose clinically distinct subgroups of diffuse large B cell lymphoma. *Proc. Natl. Acad. Sci. U.S.A.*, **100**: 9991–9996.

STATISTICAL NETWORK ANALYSIS FOR BIOLOGICAL SYSTEMS AND PATHWAYS

Youngchul Kim and Jae K. Lee

*Division of Biostatistics and Epidemiology, University of Virginia,
Charlottesville, Virginia, USA*

Haseong Kim

*Intelligent Systems and Networks Group, Department of Electrical and Electronic
Engineering, Imperial College, London, UK*

Annamalai Muthiah and Ginger Davis

*Department of Systems and Information Engineering, University of Virginia,
Charlottesville, Virginia, USA*

11.1 INTRODUCTION

Most frequent network modeling applications in biological systems include (i) gene signal transduction network modeling, (ii) gene regulation network modeling for transcriptional and functional gene regulatory pathways, and (iii) metabolic network modeling. In particular, recent studies in the modeling of these biological systems have often been performed using high-throughput biological data and information-rich literature database. These approaches can be largely divided into three categories—qualitative, quantitative, and integrative network modeling—based on the different types of data and information used in each study.

- 1. Qualitative Gene Network Modeling** Gene signal transduction network modeling has been traditionally carried out by using functional and annotation information in the literature and relevant databases. For example, computational prediction of genomewide transcription units (TUs) based on pathway-genome databases (PGDBs) and other organizational (i.e., protein complexes) annotation has been performed to improve TU organization information in *Escherichia coli* and *Bacillus subtilis* (Romero and Karp, 2004). A classification of transcription factors is proposed to organize pathways that connect

extracellular signaling to the regulation of transcription and constitutively activate nuclear factors in eukaryotic cells (Brivanlou and Darnell, 2002).

2. **Quantitative Network Modeling** Gene regulation networks have often been explored based on quantitative expression data. For example, Bayesian network modeling was used for capturing regulatory interactions between genes based on genomewide expression measurements on yeast (Friedman et al., 2000). Probabilistic models for context-specific regulatory relationships were also proposed to capture complex expression patterns of many genes in various biological conditions, accounting for known variable factors such as experimental settings, putative binding sites, or functional information in yeast stress data and compendium data (Segal et al., 2001). These quantitative pathway models have been found to effectively characterize both relationships and the magnitude of relevant genes' expression patterns and have been extensively used in recent pathway modeling in various studies (Friedman et al., 2000; Segal et al., 2001, 2004). Metabolic network modeling is also based on explicit mathematical models on quantitative chemical reaction data and kinetics.
3. **Integrative Network Modeling** Utilizing mathematical machinery in quantitative network modeling, integration of qualitative and quantitative gene network information has been attempted in recent pathway modeling studies. For example, a comprehensive genomic module map was constructed by combining gene expression and known functional and transcriptional information, where each module represents a distinct set of directly regulated and associated genes acting in concert to carry out a specific function (Segal et al., 2004). An integrated regression modeling on transcription motifs is proposed for discovering candidate genes' upstream sequences which undergo expression changes in various biological conditions in order to combine the known motif structural information and gene expression patterns (Conlon et al., 2003).

It is difficult to cover the wide range of so many different network modeling approaches, but we will attempt to briefly introduce three widely used network modeling techniques in this chapter: Boolean network (BN), Bayesian belief network, and metabolic network modeling methods.

11.2 BOOLEAN NETWORK MODELING

Boolean network modeling is one of the most widely used techniques to capture gene interaction networks based on temporal gene activation and/or expression data. This technique has been found to be useful for studying gene networks because binarization of biological expression data not only reduces noise of the data (Akutsu et al., 1998; Kauffman, 1969; Liang et al., 1998) but also efficiently captures the dynamic behavior in complex systems of many highly noisy biological variables (Bornholdt, 2005; Huang, 1999; Martin et al., 2007; Shmulevich et al., 2002). In discrete models of Boolean genetic networks, genes are modeled as switchlike dynamic elements that are either on or off and can represent large genetic networks without

parameter tuning (Bornholdt, 2005). There are several examples showing that the Boolean formalism is biologically meaningful (Li et al., 2004; Shmulevich and Zhang, 2002). We will introduce the Boolean networks and its inference algorithms in Sections 11.2.1 and 11.2.2 and the probabilistic Boolean network model as monitoring tools of dynamic behavior of organisms is also introduced in Section 11.2.3.

11.2.1 Formulation of Boolean Networks

The BN models were first introduced by (Kauffman, 1969). In a BN, a gene expression is quantized to two levels: on and off. A BN, labeled $G(V, F)$, is defined by a set of nodes $V = \{x_1, \dots, x_n\}$ and a set of Boolean functions $F = \{f_1, \dots, f_n\}$, where n is the number of genes. A Boolean function $f_i(x_1, \dots, x_k)$, where $i = \{1, \dots, n\}$ with k specified input nodes (in-degree) is assigned to node x_i . Each node is connected with three Boolean operators AND (\wedge), OR (\vee), and NOT (\sim). The regulation of nodes is defined by F . More specifically, given the values of the nodes in the set V at time $t - 1$, the Boolean functions, F , are used to update these values at time t . The Boolean function of a gene in a BN can describe the behavior of the target gene according to simultaneous changes in the expression levels of the other genes.

Example 11.1 Consider a BN consisting of three genes $V = \{x_1, x_2, x_3\}$ with corresponding Boolean functions $F = \{f_1 = x_2 \wedge x_3, f_2 = x_1, f_3 = x_2\}$ (Fig. 11.1). In this network, there are three genes with $2^3 = 8$ all possible states. An example of gene interaction can be drawn in a wiring diagram (Fig. 11.1a). The corresponding states and Boolean functions can then be defined in the truth table (Fig. 11.1b). Figure 11.1c summarizes the transitions of the eight states defined by this example. Each state is represented by a circle and the arrows show the transitions of the network states. The direction of the state transition is determined with all possible states in BN.

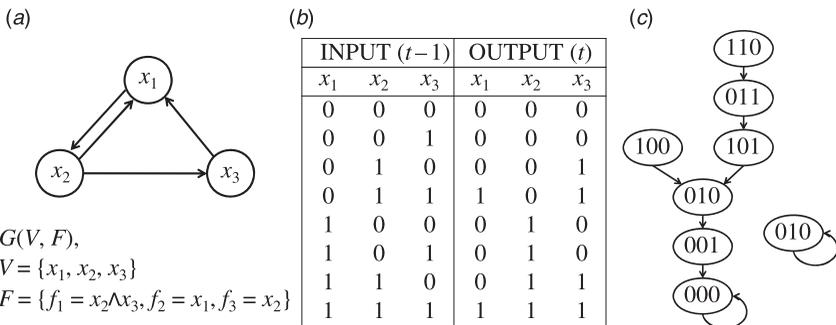


Figure 11.1 (a) Wiring diagram of BN with three nodes. (b) Truth table of the BN. (c) State transition of the BN.

11.2.2 Inference Algorithms for Boolean Networks

Many algorithms have been proposed for the inference of BNs including two widely used approaches: (1) the consistency problem (Akutsu et al., 1998) to determine the existence of a BN that is consistent with the observed data and (2) the best-fit extension problem (Boros et al., 1998).

11.2.2.1 Consistency Problem The main goal of the consistency problem is to find a perfect Boolean function for the given binary data, the so-called *identification problem*. Let $\text{pdBf}(T, F)$ be a partially defined Boolean function, that is, transitions between certain states not yet defined. Let T and F denote the set of true and false examples. For a Boolean function f , the set of true and false vectors are denoted as $T(f) = \{x \in \{0,1\}^n: f(x) = 1\}$ and $F(f) = \{x \in \{0,1\}^n: f(x) = 0\}$. The function f is then said to be a consistent extension of $\text{pdBf}(T, F)$ if $T \in T(f)$ and $F \in F(f)$. The consistency problem entails deciding whether or not there exists a consistent extension $f \in C$ for the given set of true and false examples T and F among a class of Boolean functions, denoted as C .

Example 11.2 Consider a partially defined BN of five observed binary data with three nodes, $\{(0,1,1), (1,0,1), (0,1,0), (1,0,0), (1,1,1)\}$ (Fig. 11.2a). Let $f_1 = x_2 \wedge \sim x_3$ (product of x_2 and not x_3). Then $T = \{(1,1), (1,0)\}$ and $F = \{(0,1), (0,0)\}$. The truth table of f_1 can be derived as in Figure 11.2b, that is, $T(f_1) = \{(1,0)\}$ and $F(f_1) = \{(0,0), (0,1), (1,1)\}$. In this case, f_1 is not a consistent extension with this example because $T \in T(f_1)$ is not satisfied. Figures 11.2c, d show $\text{pdBf}(T, F)$ and $\text{pdBf}[T(f_1), F(f_1)]$, respectively.

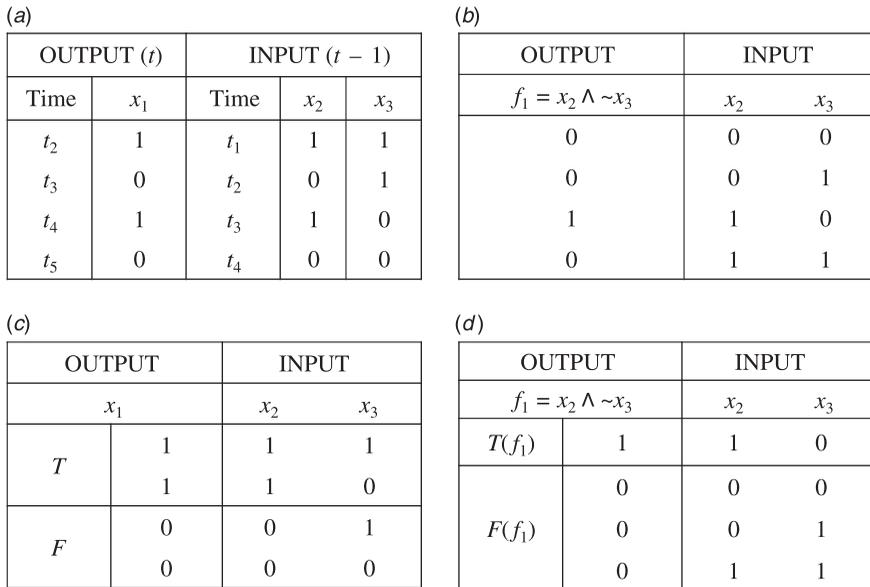


Figure 11.2 (a) Examples with three nodes. (b) Truth table of f_1 . (c) $\text{pdBf}(T, F)$. (d) $\text{pdBf}[T(f_1), F(f_1)]$.

11.2.2.2 Best-Fit Extension Problem The best way to find the perfect Boolean function is by using the consistency problem, but in realistic situations, gene expression measurements are noisy and lead to inconsistent observations. A BN learning method was thus proposed based on the so-called best-fit extension problem (Shmulevich et al., 2001). This approach is especially useful when gene expression measurements are noisy and inconsistent. As in the case of the consistency problem, let $\text{pdBf}(T, F)$ be the partially defined Boolean function. We also define a weight of any subset $S \in T \cup F$ to be $w(S) = \sum_{x \in S} w(x)$, where $w(x)$ is a positive weight for all $x \in T \cup F$.

Then the error size of the Boolean function f is defined as

$$\varepsilon(f) = w[T \cap F(f)] + w[F \cap T(f)] \quad (11.1)$$

In the best-fit extension problem, we are looking for Boolean functions from a certain class of functions, denoted as C , which make as few “misclassifications” as possible. If $w(x) = 1$ for all $x \in T \cup F$, then the error size is just the number of misclassifications. The goal is to find subsets T^* and F^* such that $T^* \cap F^* = 0$ and $T^* \cup F^* = T \cup F$ for which $\text{pdBf}(T^*, F^*)$ has an extension where $\varepsilon(f)$ is the minimum in the previously chosen class of functions, C . Consequently, any extension $f \in C$ of $\text{pdBf}(T^*, F^*)$ has minimum error size. The consistency problem can be viewed as a special case of the best-fit extension problem when $\varepsilon(f) = 0$. We may have measured identical true and/or false vectors several times, each of them possibly associated with a different positive weight.

Example 11.3 In Example 11.2, if we assume all the weights of vectors x are 1, error size of f_1 is 4 because $T \cap F(f_1) = 1$ and $F \cap T(f_1) = 0$.

11.2.3 Computation Times for Large-Scale Boolean Networks (Advanced Readers)

Simplistic Boolean formalism can represent realistic complex biological phenomena such as cellular state dynamics that exhibit switchlike behavior, stability, and hysteresis (Huang, 1999; Li et al., 2004; Shmulevich and Zhang, 2002). It is also possible to handle thousands of genes with this simple Boolean representation. However, one of the major drawbacks of Boolean networks is that they require extremely high computation times to find reliable Boolean functions. The consistency problem (Akutsu et al., 1998) and the best-fit extension problem (Shmulevich et al., 2001) work in time complexity $O[2^{2^k} \cdot {}_n C_k \cdot m \cdot n \cdot \text{poly}(k)]$, where m is the number of observed time points, n is the total number of genes, and $\text{poly}(k)$ is the time required to compare a pair of examples for a fixed in-degree k . Improved computation times of these algorithms work in $O[{}_n C_k \cdot m \cdot n \cdot \text{poly}(k)]$ (Lähdesmäki et al., 2003), but the exponential increase in the computing time for the parameters n and k is the main bottleneck for inferring the large-scale (large number of n) gene networks. In order to reduce this computational burden, the search space of possible solutions is often reduced. For example, a chi-square test-based BN (CBN) inference was used as a variable

filtering step (Kim et al., 2007). In this approach, the dichotomization of the continuous gene expression values allows us to efficiently perform the independence test for a contingency table on each pairwise network mechanism. That is, with multinomial sampling probabilities $\{\pi_{pq}\}$ in the contingency table of the i th gene at time $t - 1$ and j th gene at time t , the null hypothesis of independence of the i th and j th genes, $H_0: \pi_{pq} = \pi_{p+}\pi_{q+}$ ($p, q = 0, 1$), can be tested before the computational search for all possible solutions using the observed frequency and the expected frequency under H_0 . The conditional independence test can be performed with three (or more) genes with a contingency table in this setting.

11.2.4 Probabilistic Boolean Networks

Despite the simplicity of BN, little is known about the dynamical properties of the BN. The state of the BN can be presented by a binary string describing the states of the nodes at each moment. The binary string enables us to study the dynamic behavior of systems by monitoring the changes of the binary string as time passes (Huang, 1999; Martin et al., 2007; Shmulevich and Zhang, 2002). In BNs, a target gene is predicted by several genes known as input genes via a Boolean function. Once the input genes and the Boolean functions are known, the BN model is constructed deterministically (Figure 11.1c). However, genetic regulation exhibits uncertainty in the biological level and microarray data for inferring the model may have errors due to experimental noise in the complex measurement processes. A deterministic model is not favorable to search real situations and the development of a model incorporating the uncertainty is needed, which results in the development of probabilistic Boolean networks. Probabilistic Boolean networks (PBNs) have thus been proposed to model genetic regulatory interactions (Shmulevich et al., 2002). The steady-state probability distribution of a PBN gives important information about the captured genetic networks. In PBNs, every node (gene) can have a chance of acquiring different Boolean functions. The probabilistic selection of these Boolean functions adds flexibility in the determination of the steady state of BNs and monitoring of the dynamical network behavior for gene perturbation or intervention (Brun et al., 2005; Pal et al., 2005; Shmulevich et al., 2003).

More formally, a PBN $G(V, F)$ is defined by a set of binary-valued nodes (genes) $V = \{X_1, \dots, X_n\}$ and a list of function sets $F = (F_1, \dots, F_n)$, where each function set F_i consists of $l(i)$ Boolean functions, that is, $F_i = \{f_{j_1}^{(i)}, \dots, f_{j_{l(i)}}^{(i)}\}$. The value of each node X_i is updated by a Boolean function taken from the corresponding set F_i . A realization of the PBN at a given time instant is defined by a vector of Boolean functions. If there are N possible realizations for the PBN, then there are N vector functions $\mathbf{f}_1, \dots, \mathbf{f}_N$, where each $\mathbf{f}_i = \{f_{j_1}^{(i)}, \dots, f_{j_n}^{(i)}\}$, $1 \leq j \leq N$, $1 \leq j_i \leq l(i)$, and each $f_{j_i}^{(i)} \in F_i$. Here, \mathbf{f}_i can take all possible realizations of PBN assuming independence of the random variables in \mathbf{f}_i , $N = \prod_{i=1}^n l(i)$ is the number of possible PBN realizations. Each realization of the PBN updates the values of the nodes into their new values, that is, \mathbf{f}_j where $j \in \{1, \dots, N\}$. Let $c_j^{(i)}$ denote the probability that a certain predictor function $f_j^{(i)}$ is used to update the value of the node X_i , that is, $c_i^{(i)} = \Pr\{f^{(i)} = f_j^{(i)}\}$,

where $\sum_{j=1}^{l(i)} c_j^{(i)} = 1$. The fit of each predictor function can be evaluated by using the *coefficient of determination* (COD) (Dougherty et al., 2000). Let $f_k^{(i)}, k = 1, 2, \dots, l(i)$ denote a function of a target gene, X_i . For each k , the coefficient of determination, θ_k^i , of X_i relative to the conditioning gene sets, $X_k^{(i)}$, is defined by

$$\theta_k^i = \frac{\varepsilon_i - \varepsilon[X_i, f_k^{(i)}(X_k^{(i)})]}{\varepsilon_i} \tag{11.2}$$

where ε_i is the error of the best estimate of X_i in the absence of any conditional variables. Once appropriate predictor functions are selected for a given gene using the COD measure, the probability $c_j^{(i)}$ of the predictor function $f_j^{(i)}$ is given by:

$$c_k^{(i)} = \frac{\theta_k^i}{\sum_{j=1}^{l(i)} \theta_j^i} \tag{11.3}$$

where θ_j^i is the COD of the predictor function $f_j^{(i)}$.

The dynamics of the PBN are essentially the same as for Boolean networks but, at any given point in time, the value of each node is determined by one of the possible predictors according to its corresponding probability. The state space of the PBN consists of 2^n states. Let A be the state transition matrix. The network may transition from one state to a number of other possible states. We then have that $\sum_{j=1}^{2^n} A_{ij} = 1$ for any $i = 1, \dots, 2^n$. The detailed explanation can be found in Shmulevich et al. (2002). Thus matrix A is also a Markov matrix and the PBN is a homogeneous Markov process. An example with three nodes is shown below.

Example 11.4 Suppose we have three genes $V = (x_1, x_2, x_3)$ and the function sets $F = (F_1, F_2, F_3)$, where $F_1 = \{f_1^{(1)}, f_2^{(1)}\}$, $F_2 = \{f_1^{(2)}\}$, and $F_3 = \{f_1^{(3)}, f_2^{(3)}\}$. The truth table of these functions is as follows:

$x_1x_2x_3$	$f_1^{(1)}$	$f_2^{(1)}$	$f_1^{(2)}$	$f_1^{(3)}$	$f_2^{(3)}$
000	0	0	0	0	0
001	1	1	1	0	0
010	1	1	1	0	0
011	1	0	0	1	0
100	0	0	1	0	0
101	1	1	1	1	0
110	1	1	0	1	0
111	1	1	1	1	1
$c_j^{(i)}$	0.6	0.4	1	0.5	0.5

There are 2, 1, and 2 PBN functions for nodes 1, 2, and 3, respectively. So there are a total $N = 4$ possible network output configurations. Let K be a matrix containing lexicographically ordered rows, each row corresponding to a possible network configuration. Then,

$$K = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 2 \\ 2 & 1 & 1 \\ 2 & 1 & 2 \end{bmatrix}$$

For example, the second row of K containing (1,1,2) means that the predictors $(f_1^{(1)}, f_1^{(2)}, f_2^{(3)})$ will be used. Finally, the state transition matrix A is given by

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ P_4 & P_3 & 0 & 0 & P_2 & P_1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & P_2 + P_4 & P_1 + P_3 \\ 0 & 0 & 0 & 0 & P_2 + P_4 & P_1 + P_3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Let us consider one of the entries in A to clarify its construction. Suppose we wish to compute the transition probability $\Pr\{(1,1,0) \rightarrow (1,0,0)\}$ which corresponds to the entry $A_{7,5}$ (the indexing starts with 1). To do this, we need to use the row corresponding to $(x_1, x_2, x_3) = (1,1,0)$ in the network truth table given above. We then look for

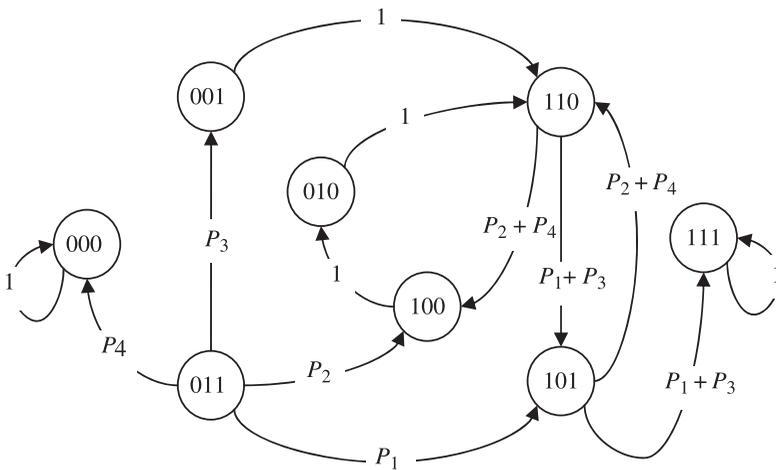


Figure 11.3 State transition diagram from Example 4.

possible combinations of the predictors for each of the three genes that will use the values (1,0,0). By direct inspection, we can see that either $(f_1^{(1)}, f_1^{(2)}, f_2^{(3)})$ or $(f_2^{(1)}, f_1^{(2)}, f_2^{(3)})$ results in (1,0,0). The two possible combinations correspond to the second and fourth rows of K . That is why this transition probability is equal to $P_2 + P_4$. All other entries in A are similarly computed. The state transition diagram corresponding to this matrix is shown in Figure 11.3.

11.2.5 Dynamics of Probabilistic Boolean Networks (Advanced Readers)

We can define the temporal iterative system $D^{t+1} = \Psi(D^t)$ by using A as

$$D^{t+1} = D^t \cdot A = D^0 \cdot A^{t+1} \quad (11.4)$$

where $D^0 = D(x)$ is the starting joint distribution. In the previous example, the probabilities of the four networks, P_1, P_2, P_3 , and P_4 in K can be obtained by using the CODs $P_1 = c_1^{(1)} c_1^{(2)} c_1^{(3)} = 0.6 \times 1 \times 0.5 = 0.3$, $P_2 = 0.3$, $P_3 = 0.2$, and $P_4 = 0.2$. Let D^t and D^{t+1} be represented by 1×2^n vectors containing the joint probabilities. Substituting those values into A and iterating Eq. (11.4), assuming that the starting joint distribution of all the genes is uniform, that is, $D^0 = [1/8, 1/8, \dots, 1/8]$, the limiting probabilities become $p = [0.15, 0, 0, 0, 0, 0, 0, 0.85]$. It means that in the long run all three genes will either be off (000), with probability 0.15, or all three genes will be on (111), with probability 0.85. These two states are called the absorbing state. This can be easily seen by looking at the state transition diagram corresponding to A in Figure 11.3.

It is important to consider whether or not a steady-state distribution exists in a PBN, of which, treatment of these concepts will not be discussed here. These can be found in many standard textbooks on stochastic processes. If the chain has a steady-state distribution, we can investigate the following question: In the long run, what is the probability that the chain is in state i regardless of the initial state? That is, with currently observed data of a patient such as gene expression microarray, we can predict the probability of the patient's risk of a disease in the future.

11.2.5.1 Computation of Steady-State Probability Distribution of PBN

The computation of the steady-state probability distribution usually includes construction of the transition probability matrix and computation of the steady-state probability distribution. The size of the transition probability matrix is $2^n \times 2^n$ where n is the number of genes in the genetic network. Therefore, the computational costs are very expensive and an efficient approximation method is required. Ching et al. (2007) introduced the approximation method for computing the steady-state probability distribution of PBN. It is based on neglecting some Boolean networks with very small probabilities during the construction of the transition probability matrix. Let us briefly introduce the random gene perturbation (Brun et al., 2005; Pal et al., 2005; Shmulevich et al., 2003). Random gene perturbation is the description

of the random inputs from the outside due to external stimuli. In BNs, the effect of the random gene perturbation is to make the genes flip from state 1 to state 0 or vice versa. When random gene perturbation is included, the transition probability matrix \tilde{A} is

$$\tilde{A}(i, j) = (1 - p)^n A(i, j) + p^{h[v(i), v(j)]} (1 - p)^{n - h[v(i), v(j)]} I_{v(i) \neq v(j)} \quad (11.5)$$

where $h[v(i), v(j)]$ is Hamming distance between the two vectors $v(i)$ and $v(j)$, p is the perturbation probability of each gene, and $I_{v(i) \neq v(j)}$ is the indicator function.

Besides the random gene perturbation, we can allow for the perturbation during state transition, which is called context-sensitive PBN (Pal et al., 2005). In the context-sensitive PBN, at each time step, the BN will be changed with a probability q . The transition matrix A in context-sensitive PBN is described as

$$\begin{aligned} A &= \sum_{j=1}^N \left((1 - q) P_j A_j + \sum_{k \neq j} q P_j \frac{P_k}{\sum_{l \neq j} P_l} A_k \right) \\ &= \sum_{j=1}^N P_j \left((1 - q) + q \sum_{k \neq j} \frac{P_k}{\sum_{l \neq j} P_l} \right) A_j \end{aligned} \quad (11.6)$$

Similarly, the random gene perturbations can be introduced into the context-sensitive PBN.

Now the computation of the steady-state probability method consists of two steps: (i) The transition probability matrix is constructed by summing the probabilities of the BNs that can lead state j to state i . In this step, an extremely high computing time is often spent to compute for many zero entries because the transition probability matrix is sparse. So, considering only nonzero entries can save the matrix construction time. (ii) Computing the eigenvector corresponding to the maximum eigenvalue—the eigenvector in the normalized form is the steady-state probability vector. The power method for eigenvectors is applied to solve the dominant eigenvalue and the corresponding eigenvector (Zhang et al., 2007). The power method has been successfully applied to compute the steady-state probability distribution based on an efficient construction of the transition probability matrix of a PBN without random perturbation.

11.2.6 Conclusion

The BN and PBN models effectively explain the temporal dynamic behaviors of living systems. Simplistic Boolean formalism can represent realistic complex biological phenomena and large-scale gene networks. It also enables us to model nonlinear relations between genes in a complex living systems (Thomas and Kaufman, 2001). The dichotomization often improves the accuracy of classification and simplifies the obtained models by reducing noise levels in experimental data. However, biological observations and data from a relatively large number, for example, 10 or more of consecutive time points are required for a reliable BN model development. Although the

major drawback of BN and PBN, especially extremely high computation times is tackled by applying efficient computation algorithms, further study is required for easily applicable BN and PBN methods. BN and PBN models are useful to investigate gene interaction of complex biological systems.

11.3 BAYESIAN BELIEF NETWORK

11.3.1 Introduction

A Bayesian (belief) network is a graphical model for probabilistic relationships among a set of variables whose joint probability distributions are compactly represented in relation to future data. The Bayesian network has several advantages in modeling biological network systems:

- It allows learning about causal relationships between variables in biological systems.
- Its probabilistic framework captures the stochastic nature of biological networks.
- Its probabilistic formalization provides mechanisms for describing uncertainty in biological systems.
- Its hierarchical structure is flexible to model a large number of parameters and variables and to perform efficient inference on them based on conditional probabilistic arguments.
- Its graphical representation provides an easy way to understand and display the network structure.

Consequently, Bayesian network modeling has become a popular tool for analyzing various gene networks to infer their causal relationship by examining statistical properties and dependency of relevant variables (Friedman et al., 2000). For example, it has been found to be very useful when each target gene is regulated by direct cascade relationships with a relatively small number of genes. The Bayesian network has also been successfully used in many applications of gene interaction discoveries such as transcription binding sites and protein–protein interaction (Imoto et al., 2003; Werhli and Husmeier, 2008).

11.3.2 Framework of Bayesian Network

11.3.2.1 Graphical Network Structure The Bayesian network can be represented as a graphical model with the joint probability distribution of all network variables and their conditional probability distributions. For example, let a set of n variables $X = \{X_1, \dots, X_n\}$ consist of a graphical network structure S and their conditional probability distributions P which describe the association among these variables. These two components can then completely specify the joint probability distribution of the Bayesian belief network for X . In this graphical representation of the Bayesian network, variables are represented by nodes connected with edges representing relationships between variables. Specifically, the graph structure of a Bayesian

network is defined by the so-called *directed acyclic graph (DAG)*, which implies that there is no cyclic path in a graph or that a directed graph is acyclic without any directed path $X_1 \rightarrow X_2 \rightarrow \dots \rightarrow X_n$ such that $X_1 = X_n$. This structural constraint of Bayesian networks is needed to utilize the probabilistic inference techniques based on stochastic and conditional independence properties of DAGs. This graph structure also indicates how different variables interact in the network where “interaction” implies a direct causal relationship between connected variables. Directed arrows (or directed edges) represent causal relationships between nodes.

Example 11.5 A simple five-gene regulatory network is shown as a Bayesian network in Figure 11.4. Here, each node represents the gene and the arrow direction indicates the causal relationship among genes. Both genes $G1$ and $G2$ regulate $G3$ and then gene $G2$ regulates both $G3$ and $G4$ simultaneously. Finally, gene $G3$ regulates $G5$. Both $G3$ and $G4$ are related because they share $G2$ as a common parent node.

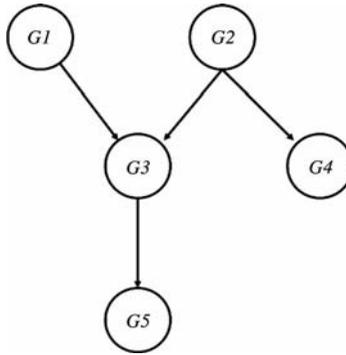


Figure 11.4 Five-gene regulatory Bayesian network.

Example 11.6 A Bayesian network can be constructed with both observed and unobserved biological factors as in Figure 11.5. The squares represent unobserved nodes whereas circles represent observed nodes. If a hidden biological variable exists, that is, there is no direct way to measure the activities of the biological variable in contrast to missing value, it is represented as a shaded square or circle.

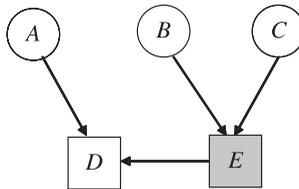


Figure 11.5 Bayesian network for observed, unobserved, and hidden variables.

11.3.2.2 Conditional and Joint Probability Distributions In addition to its graphical structure, a Bayesian network needs to be specified by the *conditional probability* distribution of each node given its parents. Let A and D be variables of interest with a direct causal (parental) relationship in Example 11.6. This relationship can be represented by a conditional probability distribution $P(D|A)$ which represents the probabilistic distribution of child node D given the information of parent node A . When both child and parent nodes are discrete variables, a contingency table can summarize the conditional probabilities for all possible states given each of its parent node states. For continuous variables, a conditional probability density function needs to be defined. For the combination of continuous and discrete nodes, a mixture distribution, for example, mixture normal distribution, will be required (Imoto et al., 2002).

The joint probability distribution of a Bayesian network has all the information about the relationships between variables in the network. However, as the number of nodes increases, the complexity of the joint probability distribution also increases rapidly. Fortunately, there is a simpler way to represent the joint probability distribution of a Bayesian network with fewer parameters. The so-called Markov conditional independence properties of the Bayesian network can be used to decompose the whole DAG S into subcomponents of directly related pairs of parent and child nodes. That is, each variable (or node) is independent of all other variables except its own parental nodes given the information of the parent nodes. Let $p(x_i | x_1, \dots, x_{i-1}) = p(x_i | \pi_i)$ be the (full) conditional distribution of x_i given all the other variables.

The joint probability distribution can then be expressed by the product of these full conditional distributions, applying the chain rule of conditional probabilities (Heckerman, 1995):

$$p(x) = \prod_i p(x_i | x_1, \dots, x_{i-1}) = \prod_i p(x_i | \pi_i)$$

For example, the full joint probability distribution can be decomposed into the full conditional distributions of paired parent–child nodes for the network structure S in Figure 11.4:

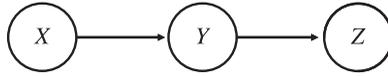
$$\begin{aligned} P(G_1, G_2, G_3, G_4, G_5) \\ &= P(G_5 | G_1, G_2, G_3, G_4)P(G_4 | G_3, G_2, G_1)P(G_3 | G_2, G_1)P(G_2 | G_1)P(G_1) \\ &= P(G_5 | G_3)P(G_4 | G_2)P(G_3 | G_2, G_1)P(G_2)P(G_1) \end{aligned}$$

11.3.2.3 Conditional Independence on Bayesian Network Two variables X and Y are defined to be *conditionally independent* given Z if

$$P(X, Y | Z) = P(X | Z)P(Y | Z)$$

These conditional independence relationships are used with three kinds of forms in the structure of the Bayesian network (Needham et al., 2008).

Example 11.7 Gene regulation as a serial connection in the Bayesian network (Fig. 11.6). In this example, gene X regulates gene Y and then gene Y regulates gene Z . If the activity status of gene Y is known, the activity of Z is only dependent on Y but independent of X (that is, X and Z are conditionally independent given Y). If the status of Y is unknown, the activity of Z still depends on X through the conditional probability distribution of Y given X .

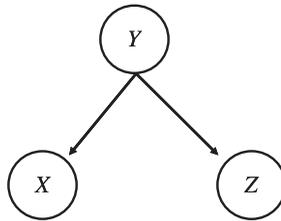


$$P(X, Y, Z) = P(X)P(Y | X)P(Z | Y)$$

$$P(Z | X) = \sum_y P(X, Y, Z) / P(X) = \sum_y P(Z | Y)P(Y | X)$$

Figure 11.6 Serial connection of Bayesian network.

Example 11.8 Transcription regulation as a diverging connection in Bayesian network (Fig. 11.7). In this example, transcription factor Y activates both genes X and Z . Information about Y is transmitted to both X and Z . For example, if gene expression of Y increases, then this will increase gene expression levels of both X and Z . Once the information about Y is known, X and Z are independent conditional on that information. If the status of Y is unknown, the expression of Z depends on the expression level of X through the conditional probability distributions of Y given X and of Z given Y .



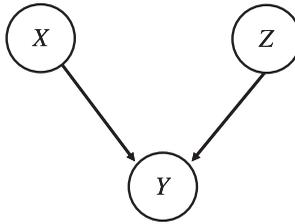
$$P(X, Y, Z) = P(Y)P(X | Y)P(Z | Y)$$

$$P(Z | X) = \sum_y P(Y)P(X | Y)P(Z | Y) / P(X)$$

Figure 11.7 Diverging connection of Bayesian network.

Example 11.9 Coregulation as a converging connection in the Bayesian network (Fig. 11.8). In this example, two genes X and Z coregulate gene Y . If the information of Y is known, then the information about gene X can be inferred without the information of Z , that is, X and Z are marginally independent. However, if information Y is unknown,

then the information of X needs to be inferred related to that of Z , that is, they are not independent given the information of Y .



$$P(X, Y, Z) = P(X)P(Z)P(Y | X, Z)$$

$$P(Z | X) = \sum_y P(Y | X, Z)P(X)P(Z) / P(X) = P(Z)$$

Figure 11.8 Converging connection.

Therefore, the structure encoded in the Bayesian network model leads to different dependencies and causal relationships. These relationships help us to understand how the changes in one or more genes may cause changes in other genes.

11.3.2.4 Inference on Bayesian Network Once we have constructed a Bayesian network, we can infer the information of each variable from those of other variables based on the network structure S and corresponding conditional probability distributions P . Then, the Bayes theorem is needed for our subsequent derivations in Example 11.5.

Using the Bayes theorem, the probability of G_3 given other sets of variables can be computed as

$$P(G_3 | G_1, G_2, G_4, G_5) = \frac{P(G_1, G_2, G_3, G_4, G_5)}{\sum_{g_3} P(G_1, G_2, G_3, G_4, G_5)}$$

where the summation is taken over all possible states g_3 of G_3 . The conditional independence in the Bayesian network makes this computation more efficient:

$$P(G_3 | G_1, G_2, G_4, G_5) = \frac{P(G_5 | G_3)P(G_4 | G_2)P(G_3 | G_2, G_1)P(G_2)P(G_1)}{\sum_{g_3} P(G_5 | G_3)P(G_4 | G_2)P(G_3 | G_2, G_1)P(G_2)P(G_1)}$$

However, the exact probabilistic inference on a Bayesian network based on the Bayes theorem often becomes computationally impractical, especially when the network size and the numbers of different states of variables are big. This kind of exact inference is known to be an NP (non-polynomial)–hard problem for discrete variables in computation. Difficulty also arises when there are a good number of undirected edges in a Bayesian network structure. These issues and commonly used

algorithms are more carefully discussed by Lauritzen and Spiegelhalter (1988), Jensen et al. (1990), Dawid (1992), and Heckerman (1995).

11.3.3 Learning Bayesian Networks (Advanced Readers)

One of the key questions in Bayesian network inference is how to learn its network structure and parameters from observed data. These problems can be summarized as follows:

Network structure	Observation	Method
Known	Full	Maximum likelihood, maximum a posteriori
	Partial	Expectation-maximization (EM) algorithm
Unknown	Full	Search through all model space
	Partial	Structural EM

Partial observation of a Bayesian network implies that there are some unknown variables such as hidden variables or variables with missing values. If the network structure is unknown, the topology of the graph is also partially or completely unknown. When the network structure also partially known through prior information such as literature and experimental information, the unknown parameters of the network can be estimated based on the conditional probability inference aforementioned. However, if there is no prior knowledge about the network structure, the inference on this kind of Bayesian network becomes difficult and often computationally challenging to estimate both the optimal structure and the parameters of the network (Murphy and Mian, 1999).

11.3.3.1 Learning Parameters of Bayesian Networks The parameter estimation for a network can be performed by maximizing its joint probability distribution, the so-called maximum-likelihood estimation (MLE) method:

$$\hat{\theta}_{\text{ML}} = \arg \max_{\theta} L(\theta) = \arg \max_{\theta} \prod_{i=1}^m \prod_{j=1}^n P(x_{ij} | \theta_i)$$

where $L(\theta)$ is the likelihood of observed data given a network structure and parameters. However, for full Bayesian inference, prior distributions of model parameters need to be considered. Then, maximum a posteriori (MAP) estimators can be obtained by maximizing its posterior probability:

$$\hat{\theta}_{\text{MAP}} = \arg \max_{\theta} P(\theta | x)$$

This problem is often solved by maximizing the full joint distribution, the product of the joint probability distribution and prior distribution $\pi(\theta)$ through Bayes's theorem:

$$P(\theta | x) = \frac{P(x | \theta)\pi(\theta)}{\int_{\Theta} P(x | \theta)\pi(\theta) d\theta}$$

Often, maximum-likelihood and maximum-posterior estimates are quite consistent and provide good predictive models for many practical applications (Needham et al., 2008). The confidence intervals for each parameter can also be obtained based either on maximum-likelihood or maximum-posterior estimators. Note again that the parameters for Bayesian networks may be learned even when the data information is incomplete using statistical techniques such as the EM algorithm and the Markov chain Monte Carlo (MCMC) technique (Murphy and Mian, 1999).

Let us illustrate how the conditional probability inference is performed on an actual estimating problem. We can represent the joint probability distribution in the form

$$f(x) = \prod_{i=1}^n g_i(x_i | \pi_i)$$

where g_i is the corresponding probability function and π_i represents the set of all other variables and/or prior parameters. We can then learn about the parameters from data using various statistical approaches such as ML or EM for maximum-likelihood estimation and MAP or MCMC for maximum-posterior estimation.

Example 11.10 Bayesian network parameter estimation. Let X_{i1}, \dots, X_{in_i} be continuous variables for individual genes' microarray expression levels and have identically independent normal distributions with mean μ_i and variance σ_i^2 . The probability density function for the normal distribution is

$$g_i(x_i | \theta) = \frac{1}{2\pi\sigma_i^2} \exp\left[-\frac{(x_i - \mu_i)^2}{2\sigma_i^2}\right]$$

And the likelihood function for parameter $\theta = (\mu_i, \sigma_i^2)$ is

$$L(\theta) = \prod_{j=1}^{n_i} g_i(x_{ij} | \theta)$$

Finally, the following ML estimators of parameters are obtained by maximizing $L(\theta)$:

$$\hat{\mu}_{i,\text{ML}} = \bar{X}_i = \frac{1}{n_i} \sum_{j=1}^{n_i} X_{ij} \quad \text{and} \quad \hat{\sigma}_{i,\text{ML}}^2 = \frac{1}{n_i} \sum_{j=1}^{n_i} (X_{ij} - \bar{X}_i)^2$$

Note that it is often convenient to maximize the log-likelihood function, a monotone function of the likelihood whose maximum will correspond to the maximum of the original likelihood function. If our variables are discrete, we instead use the multinomial distribution whose probability mass function is as follows:

$$g_i(x_i | \theta) = \binom{N}{N_{i1} \dots N_{iK}} \prod_{n=1}^K \theta_{ij}^{N_{ij}}$$

where $\theta = (\theta_{ij})_{j=1}^K$ are the parameters with $\sum_{j=1}^K \theta_{ij} = 1$ and N_{ij} is the observed frequency of states j and $N = \sum_{j=1}^K N_{ij}$. Assume the prior distribution for parameter θ is the Dirichlet distribution $\text{Dir}(\theta | \alpha_{i1}, \dots, \alpha_{iK})$, which is the conjugate prior for multinomial distribution in Bayesian inference. The corresponding probability density function is

$$p(\theta | \alpha_{i1}, \dots, \alpha_{iK}) = \frac{\Gamma(\sum_{j=1}^K \alpha_{ij})}{\prod_{j=1}^K \Gamma(\alpha_{ij})} \prod_{j=1}^K \theta_{ij}^{\alpha_{ij}-1} \quad \text{with } \alpha_{i1}, \dots, \alpha_{iK} > 0$$

We then obtain the Dirichlet posterior distribution as follows:

$$p(\theta | x_i, \alpha_{i1}, \dots, \alpha_{iK}) = c \prod_{j=1}^K \theta_{ij}^{N_{ij} + \alpha_{ij} - 1}$$

where c is a normalization constraint. Then the MAP estimator for parameter θ_{ik} is $\hat{\theta}_{ik, \text{MAP}} = (N_{ik} + \alpha_{ik}) / (N + A)$ where $A = \sum_{j=1}^K \alpha_{ij}$.

In certain applications, continuous variables are converted into discrete variables in order to reduce high noise levels in continuous measurements. Note that making a continuous variable into a discrete one may lose certain information whose benefits need to be evaluated according to study goals, for example, gene classification performance between responders and non responders to a therapy. When child and parent nodes are discrete variables, we can evaluate the conditional probability by a contingency table which specifies the transition probabilities among different states of the child node for each combination of parent node states. When such calculations are computationally intractable, there are alternative methods such as Gaussian distribution-based linear inference, semiparametric density modeling and nonparametric regression models to derive approximated probabilities among various state values in the Bayesian network (Friedman et al., 2000; Imoto et al., 2002).

For a continuous child node both with discrete and continuous parent nodes (hybrid nodes), we can use, for example, a mixture (conditional) Gaussian distribution with the population mean of a child node as the linear combination of the continuous parent conditional distributions across different states of the discrete parent node. Conversely, for a discrete child node with hybrid nodes, we can apply, for example, logistic regression or probit models for inferring conditional distributions of the discrete child node variable. For example, in a logistic regression framework, we use

$$p(x_i | \pi_i, \theta) = \frac{\exp(\pi_i^t \theta_i + \theta_{i0})}{\sum_j \exp(\pi_j^t \theta_j + \theta_{j0})}$$

where θ is the logistic regression parameter vector and π_i is the state of the parent node.

11.3.3.2 Learning Structure of Bayesian Network If the complete structure of a network is unknown or partially known, we must learn its structure from the

observed data. Let S^h represent the (topological) structure parameter for the possible network structure as in Heckerman (1995). Given data D , the posterior distribution $p(S^h | D)$ and the posterior distributions $p(\theta | D, S^h)$ can be computed as described earlier. The posterior distribution $p(S^h | D)$ is then obtained by the Bayes theorem as

$$p(S^h | D) = p(S^h)P(D | S^h)/p(D)$$

where $p(D)$ is a normalizing constant. The log marginal likelihood $\log p(D | S^h)$ needs to be efficiently calculated for identifying the optimal structure across the entire structure parameter space S^h . Under certain conditions such as exponential family, mutual independence, conjugate priors, and no missing data, this posterior distribution can be easily computed in an analytic manner. Otherwise, we may need to rely on stochastic and approximation techniques (Buntine, 1994; Heckerman et al., 1996). Specifically, to compute the log marginal likelihood for incomplete data, the Monte Carlo sampling method can be applied using Bayes theorem:

$$p(D | S^h) = \frac{p(\theta_S | S^h)p(D | \theta_S, S^h)}{p(\theta_S | D, S^h)}$$

Such a Monte Carlo method can provide a more accurate estimate if a large number of iterations or sampling can be implemented. However, this kind of computational strategy often becomes extremely inefficient for large data. A Gaussian approximation which is based on Laplace's method may be a more efficient alternative for inferring the structure parameter space on such a large dataset (Chib, 1995; Raftery, 1996).

Note that it is well known that an exhaustive computational search for the best network over all possible network structures is an NP-hard problem—a computing problem that cannot be solved by any finite computing algorithm. Thus, heuristic search algorithms such as greedy search, Monte Carlo methods, and other statistical algorithms are often used in the search for the best Bayesian network structure. For example, a greedy search algorithm compares structures which differ only by a single arc added, removed, or reversed. Then, there are various optimization criteria to select the relevant network structure. The log relative posterior probability is frequently used:

$$\log p(D, S^h) = \log p(S^h) + \log p(D | S^h)$$

For large data, this can be approximated as the so-called Bayesian information criterion (BIC):

$$\log p(D | S^h) \approx \log p(D | \hat{\theta}_S, S^h) - \frac{k}{2} \log N$$

or $\text{BIC} = -2 \ln L(\hat{\theta}_k) + k \ln N$, where $\hat{\theta}_S$ is the ML configuration of θ_S and k is the dimension of parameter θ_S . Note that this BIC approximation does not depend on prior specifications. Other frequently used model selection criteria are the Akaike information criterion (AIC) and the information-theoretic measure of model

complexity (ICOMP) as follows:

$$\begin{aligned} \text{AIC} &= -2 \ln L(\hat{\theta}_k) + 2k \\ \text{ICOMP} &= -2 \ln L(\hat{\theta}_k) + s \ln \text{tr}(I^{-1}(\hat{\theta}_k/s) - \ln |I^{-1}(\hat{\theta}_k)|) \end{aligned}$$

where k is the number of parameters, n is the sample size, $\hat{\theta}_k$ are the parameter estimates, $I^{-1}(\hat{\theta}_k) = \text{var}(\hat{\theta}_k)$ (the inverse of Fisher information matrix), and $s = \text{rank of } I^{-1}(\hat{\theta}_k)$. Intuitively, these criteria reflect how well the parameterized structure predicts the data whereas the best network structure can be selected by minimizing one of these criteria. Note that BIC gives a penalty for the complexity of the structure ($k/2 \log N$) and that ICOMP accounts for the covariance structure of a model and thus colinearity between the factors and dependence among the parameter estimates.

More advanced search algorithms include a greedy search algorithm with many random initial start structures and a simulated annealing method to efficiently search the structure parameter space with multimodality (Heckerman, 1995) and a linear space best-first search which explores the linear space of all network structures (Korf, 1993). Chickering (2002) showed a greedy search with random starts produces better models than either simulated annealing or best-first search alone. Allen and Greiner (2000) provided an empirical comparison among standard model selection criteria such as the AIC, BIC, and cross-validation for learning belief networks, suggesting that both AIC and cross-validation are good criteria for avoiding overfitting problem but the BIC is not. Peña et al. (2005) studied cross-validation as a scoring criterion for learning the dynamics of the Bayesian network which showed that cross-validation led to models that performed better for a wide range of sample sizes by carrying out an experimental comparison of cross-validation and the Bayesian scoring criterion.

11.3.3.3 Prior Specifications To compute the posterior probability of a network structure-given data, we need to consider the structure prior $p(S^h)$ and the parameter prior specifications $p(\theta_S | S^h)$. However, there were too many possible network structures even for data with a small number of variables so that it is extremely difficult to specify priors with all possible network structures. Fortunately, it has been shown that some assumptions such as distribution independence equivalence and independence modularity provide an easy way to derive the structure and parameter priors with a manageable number of network structures for direct assessment (Heckerman, 1995). For example, the simplest approach for assigning priors to network structure is to assume that every structure is equally likely with a uniform prior, which is convenient but generally quite unrealistic. It is often more realistic to give the order constraint of variables in each structure through a knowledge of time precedence, which significantly reduces the space of network structures (Buntine, 1991). Madigan et al. (1995) used a posterior probability distribution constructed by human experts but suggested using more automated and intelligent knowledge-generating strategies in obtaining and defining such prior structure information.

11.3.4 Recent Applications of Bayesian Networks

11.3.4.1 Dynamic Bayesian Networks A dynamic Bayesian network (DBN) is an extension of the standard Bayesian network to a model with cyclic causal relationships among temporal time-series biological variables. While classic Bayesian networks are based on static data, DBN can use time sequential data for constructing causal relationships. Since there are many feedback processes in biological systems, it is important to consider such feedback loops in DBN. The joint probability distribution of DBN can be represented as

$$P(X_1, \dots, X_n) = P(X_1)P(X_2|X_1) \times \dots \times P(X_n|X_{n-1})$$

where $X_i = (x_{i1}, \dots, x_{ip})'$ is a p -dimensional random vector. We assume that gene i has parents π_i and the network structure is assumed to be stable through all time points. In addition, forward edges, edges from time $i - 1$ to i , are allowed in the network structure according to the time dependence. The conditional probability $P(X_i | X_{i-1})$ can then be decomposed into the product of conditional probabilities of each gene given its parents as below:

$$P(X_i | X_{i-1}) = P(X_{i1} | P_{i-1,1}) \times \dots \times P(X_{ip} | P_{i-1,p})$$

where $P_{i-1,j} = (P_{i-1,1}^{(j)}, \dots, P_{i-1,p}^{(j)})$ is a random variable vector of parent genes of the j th gene at time $i - 1$. The DBN can be trained by a similar learning process as a standard Bayesian network (Murphy and Mian, 1999; Kim et al., 2003). Although DBN is a useful approach for predicting gene regulatory networks with time course observations, it has two disadvantages: the relatively low accuracy of prediction and the excessively long computational time. Since these problems are mainly caused by a large number of variables defined in the network structure, one may overcome these limitations of DBN by reducing the network structure to the most essential core structure with a relatively small number of variables in the network (Zou and Conzen, 2005). For example, potential regulators for each gene can be preselected by identifying differentially expressed genes by comparing time-lagged expression data for each target gene and its potential regulators.

11.3.4.2 Integration of Prior Knowledge of Gene Regulatory Networks in Bayesian Network

Based on a large number of recent gene expression profiling studies, inference on gene regulatory networks has become one of the most active fields in bioinformatics. However, the information contained in a limited number of “snapshot” gene expression datasets is limited to infer many dynamic cascade relationships of gene networks. Therefore, many researchers are interested in integrating various types of data from heterogeneous sources of information such as microarray data, known biological pathways, transcription binding sites, and functional information in the literature and large biological information databases.

For example, Imoto et al. (2003) introduced a statistical method for estimating a gene network based on both microarray gene expression data and various biological knowledge databases such as protein–protein and protein–DNA interaction,

transcription binding sites, and literature information. These authors included biological knowledge to estimate gene network structures under a Bayesian framework, attempting to control the trade-off between microarray information and biological knowledge based on the prior knowledge as an energy function. The hyperparameters in this modeling are represented by the weights associated with the various sources of prior knowledge relative to the data. In their subsequent study, Imoto et al. (2006) proposed an error-tolerant model for incorporating biological knowledge with expression data in estimating gene regulation networks, considering the fact that some known biological pathway information may be incorrect due to experimental artifacts or human error.

Husmeier et al. (2008) recently attempted to improve the reconstruction of regulatory networks by a systematic integration of prior knowledge. They defined the biological prior knowledge matrix B in which the entries $B_{ij} \in [0, 1]$ represent the strength of prior knowledge about interactions between nodes. The agreement between a given network structure and prior knowledge matrix were then assessed with an energy function. This study partially demonstrated the Bayesian integration scheme systematically improved the network reconstruction compared with the conventional approaches that rely either on protein concentrations or prior knowledge from literature alone.

11.3.5 Hierarchical Bayesian Models

Hierarchical Bayesian models are commonly used in many biological applications because they incorporate both physical and statistical models with uncertainty. These models are used in pathway analysis because of their ability to manage multiple data sources, uncertain, physical models with stochastic parameters, and expert opinion. A summary of these models is given in Wikle (2004). In brief, all unknowns are treated as if they are random and evaluated probabilistically as follows:

$$[\text{Process} \mid \text{Data}] \propto [\text{Data} \mid \text{Process}][\text{Process}]$$

where $[A]$ denotes the distribution of A .

In hierarchical Bayesian models, the way this equation is decomposed is based on what we know about the process, and what assumptions we are willing and able to make for simplification. We do this because it is often easier to express conditional models than full joint models. For example, one decomposition would be

$$[A, B, C] = [A \mid B, C][B \mid C][C]$$

If A and C are conditionally independent, our decomposition becomes

$$[A, B, C] = [A \mid B][B \mid C][C]$$

The hierarchical Bayesian modeling methodology begins with separating the unknowns into two groups: (1) process variables (actual physical quantities of interest) and (2) model parameters (quantities introduced in model development). Three distributions are specified: (1) $[\text{data} \mid \text{process, parameters}]$, (2) $[\text{process} \mid \text{parameters}]$, and (3) $[\text{parameters}]$ to give us the posterior $[\text{process, parameters} \mid \text{data}]$ which is proportional to the product of these three distributions. One simple example is given as follows:

Data model:

$$X_i | \theta, \sigma^2 \sim N(\theta, \sigma^2)$$

Process model:

$$\theta | \theta_0, \mathbf{z}, \boldsymbol{\beta}, \tau^2 \sim N(\theta_0 + \mathbf{z}\boldsymbol{\beta}, \tau^2)$$

Parameter model:

$$\boldsymbol{\beta} | \boldsymbol{\beta}_0, \boldsymbol{\Sigma} \sim N(\boldsymbol{\beta}_0, \boldsymbol{\Sigma})$$

The posterior distribution $\theta, \boldsymbol{\beta} | X_1, \dots, X_n$ is proportional to the product of these distributions.

Due to the complexity of most of these models, an analytical solution is intractable. Therefore, one often relies on numerical integration (for low dimensions) or MCMC methods such as the Metropolis–Hastings (Hastings, 1970; Metropolis et al., 1953) or Gibbs sampler algorithms (Gelfand and Smith, 1990).

One implementation of Bayesian hierarchical modeling can be found in Wang (2008) where differential expression patterns are compared in the pathway and the rest of the genes. The model is one for the gene expression values:

$$\text{Model 1: } y_{gklm} = \mu_{jk} + \text{Array}_l + \text{Pathway}_{m(g)} + \varepsilon_{gklm}$$

where y represents log-transformed gene expression values, $j = 1$ if gene g is from the pathway m and $j = 0$ otherwise, $k = 1$ for case values and $k = 0$ for control values. The parameters μ_{jk} model systematic effects affecting gene expression values. The terms Array_l and $\text{Pathway}_{m(g)}$ are random variables denoting the effect of the l th array and the m th pathway. Finally, ε_{gklm} represents variations due to measurement error, which are assumed to have a normal distribution with mean zero and constant variance. Differential expression of candidate genes between cases and controls are tested by computing the difference between $\mu_{11} - \mu_{10}$ and $\mu_{01} - \mu_{00}$. Relevant distributions can be assumed as follows:

$$\text{Array}_l \sim N(0, \sigma_{\text{array}}^2)$$

$$\text{Pathway}_{m(g)} \sim N(0, \sigma_{\text{pathway}}^2)$$

This model allowed the analysis of microarray data with more refined modeling of covariance structure between genes through specification of random effects, and the ability to account for complicated experimental designs through inclusion of design factors and covariate effects.

In summary, the flexible framework of Bayesian hierarchical modeling will likely be exploited more often in future pathway analysis. The method is computationally expensive, but its benefits (allowing the quantification of all aspects of uncertainty in the problem and incorporating physical models and expert opinions) far outweigh this cost.

11.3.6 Software

Many publicly available software algorithms have been developed for learning Bayesian networks. Scheines et al. (1994) developed TETRAD II software for learning the Bayesian network and Spiegelhalter et al. (1995) developed flexible software called BUGS (Bayesian inference using Gibbs sampling) for the Bayesian analysis of complex statistical models using MCMC.

In particular, the statistical analysis software R, the so-called GNU Splus, is a freely available language and environment for statistical computing and graphics which can provide an extensive range of statistical and graphical techniques (see Chapter 13 for more information). Many useful R packages such as gRbase, deal, and BNArray are also available for learning the Bayesian network (Bøtcher et al., 2003, Chen et al., 2006). GRAPPA also provides a software suite of functions in R for probability propagation in discrete graphical models. Table 11.1 summarizes some of the widely used software for learning Bayesian networks.

11.3.7 Summary

Bayesian network inference is a very useful method to understand and visualize the relationship among various genetic variables of interest. In particular, its Markov probabilistic inference is quite suitable in capturing causal relationships in gene network cascades. However, there still remain many problems to avoid excessive computational time to efficiently update the network structure and to improve accuracy and complexity in network structure inference. It is thus highly recommended that these Bayesian network models be validated both with known true-positive and simulated false-positive network variables and relationships.

The main goal of Bayesian network inference is to learn the causal relationship between observed data and variables. However, the resulting Bayesian network graph does not represent directional causal relationships. That is, due to the symmetry or

TABLE 11.1 Software for Learning Bayesian Network

Name	Description
DEAL	R package for learning Bayesian networks with mixed variable
BNArray	R package for constructing gene regulatory networks from microarray data by using Bayesian network
Bnlearn	R package for learning Bayesian network structure
GRAPPA	Suite of functions in R for probability propagation in discrete graphical models
BUGS	Bayesian inference using Gibbs sampling
CoCo	Statistics package for analysis of associations between discrete variables
DIGRAM	Discrete graphical models
JAGS	Just another Gibbs sampler
MIM	Mixed interaction modeling, a Windows program for graphical modeling
TETRAD	The TETRAD project for causal models and statistical data

equivalence property of the Bayesian network, the same probabilistic scores can be obtained from two conditional network structures of gene X, Y : $X \rightarrow Y, Y \rightarrow X$, that is, $P(X, Y) = P(X|Y)P(Y) = P(Y|X)P(X)$.

Certain biological experiments and observations can be obtained without the knowledge of gene regulations, for example, from the literature, transcription factor, and protein interaction modeling. For more detailed examples, readers may refer to Werhli and Husmeier (1998). It is also important to decide whether the hidden variables are present in the network structure. When the best network structure is found without hidden variables, the edges of a Bayesian network can be more easily interpreted with causal relationships under the assumption of the Markov assumption. If a Bayesian network includes hidden causal variables, it increases the complexity and uncertainly for inferring such causal relationships (Chickering and Heckerman, 1997). Therefore, it is highly recommended to carefully consider the necessity of hidden variables and the possible experimental strategies to observe such hidden variables.

11.4 MODELING OF METABOLIC NETWORKS

The metabolic network refers to the different biochemical reactions occurring in the cell's cytoplasm catalyzed by enzymes. These reactions are responsible for generating energy by the process of catabolism (energy molecules like ATP and electron carriers like NADH, NADPH, etc., electron carriers are also sources of energy because they generate ATP by oxidative phosphorylation at the mitochondrial membrane) and synthesis of vital cellular components and essential amino acids (the 12 major precursor metabolites) (Nielsen et al., 2003). Just like gene regulatory networks and other forms of networks, metabolic networks are interconnected and modeling of biochemical networks will reveal regulatory information about the network. Metabolic network modeling has several important biotechnological implications toward improving certain biological processes. For example, one may want to maximize the yield of commodity chemicals such as lactate and pharmaceuticals, for example, HIV vaccines and insulin, by accurately understanding and controlling such metabolic kinetics.

11.4.1 Representation of Metabolic Networks

There are several ways to graphically represent metabolic reactions wherein nodes represent the objects of interest and the links represent the relation between them. The nodes and links connecting them could represent one of the four following cases (Figure 11.9):

- (a) *Substrate Graph* Nodes represent the chemical compounds or the different metabolites and the links connecting them represent the different reactions involving the metabolites.
- (b) *Reaction Graph* In this case, the reactions form the nodes and a link exists between two reactions if the product of one reaction becomes the substrate of the other or vice versa.

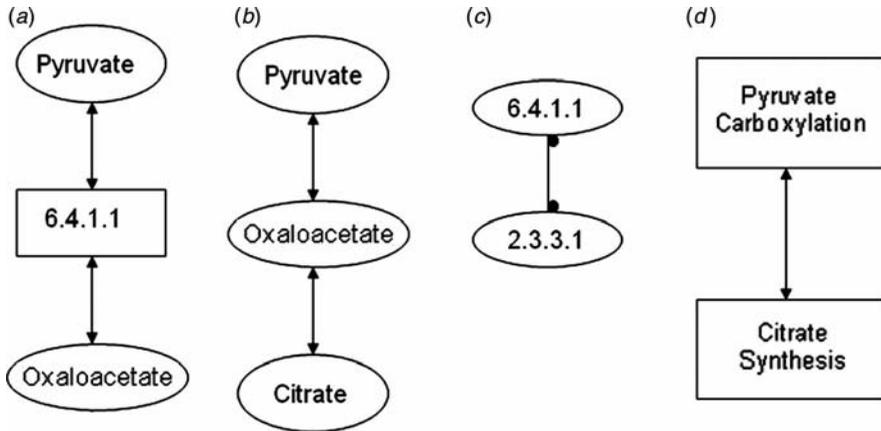


Figure 11.9 (a) Substrate–enzyme bipartite graph where one set of nodes, E.C.6.4.1.1 and E.C.2.3.3.1, represents the enzymes pyruvate carboxylase and citrate (Si)-synthase and the other set represents the substrates and products. (b) Substrate graph. (c) Enzyme-centric graph. (d) Reaction-centric graph. (Courtesy of Rahman and Schomburg, 2006.)

- (c) *Enzyme-Centric Graph* Enzymes form the nodes and the enzymes are connected if the products of the reaction represented by one enzyme become the substrate of another enzyme.
- (d) *Substrate–Enzyme Bipartite Graph* Consists of two sets of nodes and only the nodes from different sets can be connected. A typical substrate–enzyme bipartite graph could consist of two groups of nodes wherein one group represents the substrate and products of a reaction while the other groups could represent the reaction or the enzyme catalyzing the reaction; the reaction node is linked to the substrate and product node associated with it (Jing et al., 2006).

11.4.2 Basic Concepts for Metabolic Network Modeling

Metabolic network modeling provides an in-depth understanding of different biological mechanisms by breaking down all intracellular reactions to molecular details. The rates of different reactions are quantified and the model thus developed would serve as an efficient way to predict the effect of different strategies aimed at enhancing the rate of production of certain metabolites. The entire metabolic network modeling follows two major conservation principles: conservation of mass and energy. We discuss some key ideas central to metabolic network modeling.

Of all the data available about metabolic networks, the most relevant information is the flux of metabolites across different pathways. As stated earlier, the study of metabolic networks helps achieve several biotechnological objectives such as enhanced synthesis of proteins, low-molecular-weight compounds, and transforming substrates into products (Bonarius et al., 1997). The flux distribution information concerning the rate of flow of metabolites across different pathways could later be

used to perform a flux balance analysis (FBA) on the network to achieve one or more of the following goals:

- (a) The primary goal of many chemical and biotechnological industries employing microbial cultures to generate useful products would be to maximize the rate of production of these chemicals from the microbes. To that end, it is imperative that the process engineers comprehend the importance of identifying the most important pathway the microorganisms use toward generating useful chemicals and attempt to maximize productivity or efficiency of the particular pathway.
- (b) The ratio of nutrients utilized toward synthesizing biomass and products is smaller in comparison toward energy supply for cells and maintenance. In addition, several undesired products are also produced and significant energy is wasted in driving those pathways. Analysis must be carried out on the metabolic network to minimize resources and energy wastage by the cell (Bonarius et al., 1997).
- (c) FBA could have other potential applications in understanding the location of metabolic network control, predicting maximum theoretical yield and toxicological effects on cellular growth, and understanding the critical gene networks relevant in disease propagation (Bonarius et al., 1997).

To achieve the above goals, quantification of intracellular metabolite flow (or flux) is required. Presently, there are no efficient ways to quantify intracellular metabolite flux except for an isotope tracer technique using nuclear magnetic resonance (NMR) (Bonarius et al., 1997). However, this technique is expensive, laborious, and not suitable for industrial scale due to cell-to-cell heterogeneity. If a cell is disrupted in order to measure the concentration of different metabolites within cells to measure the flux, the internal environment such as the enzymatic control on metabolites is irreversibly lost and flux measurement becomes impossible. However, measurable rates such as substrate uptake rate and product secretion rates could be utilized to quantify internal metabolite flux. Besides, there are also certain conditions and constraints imposed while evaluating the flux across different pathways (Nielsen et al., 2003).

The basic equations representing the material and energy balance underlying FBA are given by Eqs. (11.7)–(11.9) (Nielsen et al., 2003). Equation (11.7) represents the rate of generation, consumption, and accumulation of substrate, product, and intracellular metabolites. Equations (11.8) and (11.9) represent the energy conservation principle inside the cell by describing the intracellular flow (generation, consumption, and accumulation) of ATP and NADH:

$$r = T^T \mathbf{v} \quad (11.7)$$

$$T_{\text{ATP}}^T \mathbf{v} = 0 \quad (11.8)$$

$$T_{\text{NADH}}^T \mathbf{v} = 0 \quad (11.9)$$

where matrix T in (11.7) provides the stoichiometry for different reactions in different rows while columns represent the different metabolites. Similarly, T in Eqs. (11.8) and (11.9) represents the stoichiometry of ATP and NADH in different reactions. The r in Eq. (11.7) represents the rate of change in concentration for different metabolites in the

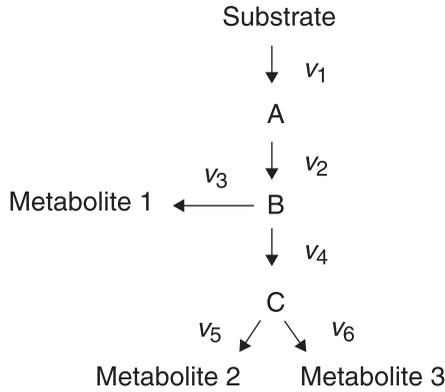


Figure 11.10 Simple network (subset of network in Fig. 11.11) whose fluxes need to be evaluated. (Courtesy of Nielsen et al., 2003.)

network while \mathbf{v} in Eqs. (11.7)–(11.9) represents the flux vector. While the rate of substrate consumed and product secreted by the cell is nonzero, the rate of change of intracellular metabolites is zero because of the assumption of steady state. The example in Figure 11.10 illustrates the concept of the stoichiometry matrix [T in Eq. (11.7)] and how it can be solved to obtain the intracellular fluxes (Fig. 11.11) for different reactions.

A steady-state assumption of metabolites is one of the major assumptions behind deriving flux balance in Figure 11.10. Framing flux balance for intracellular metabolites A , B , and C , we obtain

$$A: v_1 = v_2 \tag{11.10}$$

$$B: v_2 = v_3 + v_4 \tag{11.11}$$

$$C: v_4 = v_5 + v_6 \tag{11.12}$$

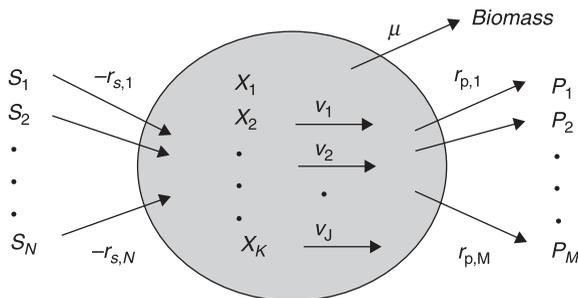


Figure 11.11 Illustration of metabolite flow (substrate uptake, product secretion, and intracellular reaction) across cell. (Courtesy of Nielsen et al., 2003.)

Transforming Eqs. (11.10)–(11.12) into the stoichiometry matrix (T), flux vector (\mathbf{v}), and rate of change of metabolite concentration (r), we obtain

$$T = \begin{matrix} & S & P_1 & P_2 & P_3 & A & B & C \\ \begin{bmatrix} -1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 1 & 0 & 0 & -1 \end{bmatrix} & & & & & & & \end{matrix} \quad (11.13)$$

$$\mathbf{v} = \begin{matrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \end{bmatrix} & & \end{matrix} \quad (11.14)$$

$$r = \begin{matrix} \begin{bmatrix} r_s \\ r_{p1} \\ r_{p2} \\ r_{p3} \\ 0 \\ 0 \\ 0 \end{bmatrix} & & \end{matrix} \quad (11.15)$$

Among the above matrices and vectors, the matrix T and vector \mathbf{r} could be determined from metabolic network databases and experimental measurements, respectively, which in turn could be used to evaluate the unknown flux parameters for intracellular metabolites in vector \mathbf{v} .

11.4.3 Identification of Control Structures and Robustness in Metabolic Networks

After evaluating the fluxes in different pathways, the next step is to identify the control points or bottlenecks in the pathway in question to maximize the productivity and yield of the desired product. This section describes the different approaches toward achieving the goal.

Metabolic control analysis (MCA) is a powerful tool to identify the bottleneck control in a given metabolic pathway (Nielsen et al., 2003). One of the important ideas of MCA is to evaluate the degree of flux control exhibited by each enzyme in the

pathway toward product formation. This is evaluated by calculating the flux control coefficient (C_i^j) defined as

$$C_i^j = \left(\frac{\partial J}{\partial k_i} \right) \left(\frac{k_i}{J} \right) \quad (11.16)$$

where J represents the flux of the pathway in question (j) and k_i represents the rate constant of the enzyme i whose control is to be evaluated. Also, by the flux control summation theorem,

$$\sum_i C_i^j = 1 \quad (11.17)$$

Equation (11.17) explains that the control of flux in a pathway is distributed only among enzymes in the given pathway. The slowest reaction in the pathway yields the highest control coefficient and hence the strategy should be to change the architecture of the pathway such that the resistance to flow through different steps is the same and also minimized. This can be achieved by increasing the activity of the identified enzyme that wields the maximum control on the pathway flux.

The activity of an enzyme is also increased by altering the concentration of the substrates involved in the pathway. Other useful parameters such as elasticity and concentration–control coefficients have been described in great detail in Nielsen et al. (2003) and are used in conjunction with the prime parameter, the control coefficient, to identify the bottleneck in a pathway and consequently maximize the product yield.

However, this MCA approach has its own limitations. The control coefficients, elasticity coefficients, and so on, are difficult to measure and practical applications of these parameters are yet to be realized. The measurement of these coefficients is laborious and expensive. Since metabolic networks have numerous nodes, determining individual control coefficients is an enormous task (Nielsen et al., 2003).

11.4.4 Topology of Metabolic Network

The main shortcoming of MCA could be eliminated by adopting a reductionist approach by focusing on the overall organization of the network aided by topology studies. The topology of a network refers to one of the four different physical representation discussed above.

The degree of a node (k) is the number of edges connected to it. Unlike a random network, the metabolic network across different life forms of varying degrees of complexity (archae to eukaryotes) have a scale-free form of organization where the probability of finding nodes of a particular degree is given by the power law: $p(k) \sim k^{-r}$ ($r = 2.2$) (Jing et al., 2006), which implies there are a higher number of nodes with a lower degree than with a higher degree. In other words, there is a hierarchical organization of nodes where at each higher level the number of nodes decreases.

The clustering coefficient around a node (v) of degree k [CC (v)] is defined as

$$CC(v) = \frac{2|N(v)|}{k(k-1)} \quad (11.18)$$

where $N(v)$ represents the number of links between neighbors of node v . The clustering coefficient, ranging from 0 to 1 represents the ability of a node and its neighbors to

form a cluster. The average clustering coefficient for all nodes with degree k , represented by $C(k)$, also follows the power law: $C(k) \sim k - 1$. This reiterates the hierarchical organization of the network. Smaller modules congregate to form a bigger module where each module is also a functional module representing a fraction of the larger metabolic network (Jing et al., 2006).

A similar organization of the network based on the idea of hierarchical organization is that of a bow-tie which groups highly interconnected nodes as a separate module. The modules are arranged in a hierarchical network and modules with a higher number of nodes are placed at a higher position. The pinnacle is represented by a module that has the maximum number of nodes and has the strongest interconnections. Based on the number of nodes and the interconnectivity, the bow-tie typically consists of four central parts: GSC (giant substrate components), substrate subset (S), product subset (P), and isolated subset (I). GSC is the most vital component of the network which comprises a bunch of highly interconnected nodes which control the flow of important intracellular metabolites. GSC represents an important piece of the network because it comprises all the important biological pathways such as the amino acid synthesis pathways for synthesizing vital cellular components, glycolytic pathways for energy generation, and nucleic acid synthesis pathways (Jing et al., 2006).

11.4.5 Future Direction of Metabolic Network Modeling

The above-mentioned network topology analysis holds the key to major network analysis in the future. It is an elegant approach to decompose the complex network into manageable functional modules whose throughput (flux) can be subsequently enhanced to obtain higher yield and productivity of the desired metabolite. One of the key ideas that could be exploited to achieve the goal is the concept of scale-free networks where the control of the entire network is concentrated in a few select nodes and where it might be possible to maximize the flux through the network by increasing the activity of those nodes.

All major network sciences seem to concur on the view that real networks are not random but are based on robust and strong organizational principles. As explained above, major metabolic network analysis techniques could be strongly influenced by the *network reductionist approach*, where the behavior of the network could be potentially predicted by its elementary constituents and their interactions alone. Therefore, the hierarchical and scale-free property of gene networks can effectively complement techniques like metabolic control analysis and become a vital tool in future gene network analysis (Almaas and Barabási, 2006).

REFERENCES

- Akutsu, T., Kuhara, S., Maruyama, O., and Miyano, S. (1998). Identification of gene regulatory networks by strategic gene disruptions and gene overexpressions. In *Proceedings of the Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, H. Karloff (Ed.). San Francisco, CA: Society for Industrial and Applied Mathematics, pp. 695–702.
- Allen, T. V., and Greiner, R. (2000). Model selection criteria for learning belief nets: An empirical comparison. In *Proc. 17th Int. Conf. Machine Learning*, pp. 1047–1054.

- Almaas, E., and Barabási, A.-L. (2006). *Dynamics of Complex Interconnected Systems: Networks and Bioprocesses*. NATO Sciences Series. Amsterdam: Springer and IOS Press, pp. 1568–2609.
- Bonarius, H. P. J., Schmid, G., and Tramper, J. (1997). Flux analysis of underdetermined metabolic networks: The quest for the missing constraints. *Trends Biotechnol.*, **15**: 308–314.
- Bornholdt, S. (2005). Systems biology: Less is more in modeling large genetic networks. *Science*, **310**: 449–451.
- Boros, E., Ibaraki, T., and Makino, K. (1998). Error-free and best-fit extensions of partially defined Boolean functions. *Inform. Comput.*, **140**: 254–283.
- Bøtcher, S. G., and Dethlefsen, C. (2003). Deal: A package for learning Bayesian networks. *J. Stat. Software*, **8**(20): 1–40.
- Brivanlou, A. H., and Darnell, J. E., Jr. (2002). Signal transduction and the control of gene expression. *Science*, **295**: 813–818.
- Brun, M., Dougherty, E. R., and Shmulevich, I. (2005). Steady-state probabilities for attractors in probabilistic Boolean networks. *Signal Process.*, **85**: 1993–2013.
- Buntine, W. (1991). Theory refinement on Bayesian networks. In *Proceedings of Seventh Conference on Uncertainty in Artificial Intelligence*. Los Angeles, CA: Morgan Kaufmann, 52–60.
- Buntine, W. L. (1994). Operations for Learning with Graphical Models. *Journal of Artificial Intelligence Research*, **2**(1994): 159–225.
- Chib, S. (1995). Marginal likelihood from the Gibbs output. *Journal of the American Statistical Association*, **90**: 1313–1321.
- Chickering, D. M. (2002). Learning equivalence classes of Bayesian-network structures. *Journal of Machine Learning Research*, **2**: 445–498.
- Chickering, D. H. and Heckerman, D. (1997) Efficient approximations for the marginal likelihood of Bayesian networks with hidden variables. *Machine Learning*, **29**: 181–212.
- Chen, X., Chen, M., and Ning, K. (2006). *BNArray*: An R package for constructing gene regulatory networks from microarray data by using Bayesian network. *Bioinformatics*, **22**(23): 2952–2954.
- Ching, W. K., Zhang, S., Ng, M. K., and Akutsu, T. (2007). An approximation method for solving the steady-state probability distribution of probabilistic Boolean networks. *Bioinformatics*, **23**: 1511.
- Conlon, E. M., Liu, X. S., Lieb, J. D., and Liu, J. S. (2003). Integrating regulatory motif discovery and genome-wide expression analysis. *Proc. Natl. Acad. Sci. U.S.A.*, **100**: 3339–3344.
- Dawid, P. (1992). Applications of a general propagation algorithm for probabilistic expert systems. *Stat. Comput.*, **2**: 25–36.
- Dougherty, E. R., Kim, S., and Chen, Y. (2000). Coefficient of determination in nonlinear signal processing. *Signal Process.*, **80**: 2219–2235.
- Friedman, N., Linal, M., Nachman, I., and Pe'er, D. (2000). Using Bayesian networks to analyze expression data. *J. Comput. Biol.*, **7**: 601–620.
- Gelfand, A. E., and Smith, A. F. M. (1990). Sampling-based approaches to calculating marginal densities. *J. Am. Stat. Assoc.*, **85**: 398–409.
- Haseong, K., Jae, L., and Taesung, P. (2007). Boolean networks using the chi-square test for inferring large-scale gene regulatory networks. *BMC Bioinform.*, **8**: 38.
- Hastings, W. K. (1970). Monte Carlo sampling methods using Markov chains and their applications, *Biometrika*, **57**(1): 97–109.
- Heckerman, D. (1995). A tutorial on learning with Bayesian networks. Technical report MSR-TR-95-06. Microsoft Research, Redmond, WA.
- Heckerman, D. and Geiger, D. (1996). Likelihoods and priors for Bayesian networks. Technical report MSR-TR-95-54. Microsoft Research, Redmond, WA.
- Huang, S. (1999). Gene expression profiling, genetic networks, and cellular states: An integrating concept for tumorigenesis and drug discovery. *J. Mol. Med.*, **77**: 469–480.
- Imoto, S., Kim, S., Goto, T., Aburatani, S., Tashiro, K., Kuhara, S., and Miyano, S. (2002). Bayesian network and nonparametric heteroscedastic regression for nonlinear modeling of genetic network. In *Proceedings of the IEEE Computer Society Bioinformatics Conference (CSB'02)*, 200.
- Imoto, S., Higuchi, T., Goto, T., Tashiro, K., Kuhara, S., and Miyano, S. (2003). Combining microarrays and biological knowledge for estimating gene networks via Bayesian networks. In *Proceedings of the IEEE Computer Society Bioinformatics Conference (CSB'03)*, pp. 104–113.

- Imoto, S., Higuchi, T., Goto, T., and Miyano, S. (2006). Error tolerant model for incorporating biological knowledge with expression data in estimating gene networks. *Stat. Methodol.*, **3**: 1–16.
- Jensen, F., Lauritzen, S., and Olesen, K. (1990). Bayesian updating in recursive graphical models by local computations. *Comput. Stat. Q.*, **4**: 269–282.
- Jing, Z., Hong, Y., Jianhua, L., Cao, Z. W., and Yixue, L. (2006). Complex networks theory for analyzing metabolic networks. *Chin. Sci. Bull.*, **51**: 1529–1537.
- Kauffman, S. A. (1969). Metabolic stability and epigenesis in randomly constructed genetic nets. *J. Theor. Biol.*, **22**: 437–467.
- Kim, S. Y., Imoto, S., and Miyano, S. (2003). Inferring gene networks from time series microarray data using dynamic Bayesian networks. *Brief. Bioinform.*, **4**(3): 228–235.
- Korf, R. (1993). Linear-space best-first search. *Artif. Intell.*, **62**: 41–78.
- Lähdesmäki, H., Shmulevich, I., and Yli-Harja, O. (2003). On Learning gene regulatory networks under the boolean network model. *Machine Learning*, **52**: 147.
- Lauritzen, S., and Spiegelhalter, D. (1988). Local computations with probabilities on graphical structures and their application to expert systems. *J. R. Stat. Soc. B*, **50**: 157–224.
- Li, F., Long, T., Lu, Y., Ouyang, Q., and Tang, C. (2004). The yeast cell-cycle network is robustly designed. *Proc. Nat. Acad. Sci.*, **101**: 4781–4786.
- Liang, S., Fuhrman, S., and Somogyi, R. (1998). REVEAL, a general reverse engineering algorithm for inference of genetic network architectures. *Pacific Symp. Biocomput.*, **3**: 22.
- Madigan, D., Garvin, J., and Raftery, A. (1995). Eliciting prior information to enhance the predictive performance of Bayesian graphical models. *Communications in Statistics: Theory and Methods*, **24**: 2271–2292.
- Martin, S., Zhang, Z., Martino, A., and Faulon, J. L. (2007). Boolean dynamics of genetic regulatory networks inferred from microarray time series data. *Bioinformatics*, **23**: 866.
- Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., and Teller, E. (1953). Equations of state calculations by fast computing machines. *J. Chem. Phys.*, **21**(6): 1087–1092.
- Murphy, K., and Mian, S. (1999). Modeling gene expression data using dynamic Bayesian networks. Technical Report. Computer Science Division, University of California, Berkeley, CA.
- Needham, C. J., Bradford, J. R., Bulpitt, A. J., and Westhead, D. R. (2008). A primer on learning in Bayesian networks for computational biology. *PLOS Computat. Biol.* **3**(8): e129.
- Nielsen, J., Villadsen, J., and Liden, G. (2003). *Bioreaction Engineering Principle*. New York: Kluwer Academic/Plenum Publishers.
- Pal, R., Datta, A., Bittner, M. L., and Dougherty, E. R. (2005). Intervention in context-sensitive probabilistic Boolean networks. *Bioinformatics*, **21**: 1211–1218.
- Peña, J. M., Björkegren, J., and Teegné, J. (2005). Learning dynamic Bayesian network models via cross-validation. *Pattern Recognition Lett.*, **26**(14): 2295–2308.
- Raftery, A. (1996). Hypothesis testing and model selection, Practical Markov Chain monte Carlo (Gilks, W. R., Richardson, S., Spiegelter, D.) Chapman and Hall.
- Rahman, S. A., and Schomburg, D. (2006). Observing local and global properties of metabolic pathways: “Load points” and “choke points” in the metabolic networks. *Bioinformatics*, **22**(14): 1767–1774.
- Romero, P. R., and Karp, P. D. (2004). Using functional and organizational information to improve genome-wide computational prediction of transcription units on pathway-genome databases. *Bioinformatics*, **20**: 709–717.
- Scheines, R., Spirtes, P., Glymour, C., and Meek, C. (1994). *TETRAD II: Tools for Discovery*. Hillsdale, NJ: Lawrence Erlbaum Associates.
- Segal, E., Friedman, N., Koller, D., and Regev, A. (2004). A module map showing conditional activity of expression modules in cancer. *Nat. Genet.*, **36**: 1090–1098.
- Segal, E., Taskar, B., Gasch, A., Friedman, N., and Koller, D. (2001). Rich probabilistic models for gene expression. *Bioinformatics*, **17**(Suppl. 1): S243–252.
- Shmulevich, I., Dougherty, E. R., Kim, S., and Zhang, W. (2002). Probabilistic Boolean networks: A rule-based uncertainty model for gene regulatory networks. *Bioinformatics*, **18**: 261–274.
- Shmulevich, I., Gluhovsky, I., Hashimoto, R. F., Dougherty, E. R., and Zhang, W. (2003). Steady-state analysis of genetic regulatory networks modelled by probabilistic Boolean networks. *Compar. Funct. Genom.*, **4**: 601–608.

- Shmulevich, I., Yli-Harja, O., and Astola, J. (2001). Inference of genetic regulatory networks under the best-fit extension paradigm. In *Proc. 2001 IEEE EURASIP Workshop on Nonlinear Signal and Image Processing*, Baltimore, MD, June, 3–6.
- Shmulevich, I., and Zhang, W. (2002). Binary analysis and optimization-based normalization of gene expression data. *Bioinformatics*, **18**: 555–565.
- Spiegelhalter, D. J., Thomas, A., Best, N. G., and Gilks, W. R. (1995). *BUGS: Bayesian Inference Using Gibbs Sampling*, Version 0.50. Cambridge: MRC Biostatistics Unit.
- Thomas, R., and Kaufman, M. (2001). Multistationarity, the basis of cell differentiation and memory. I. Structural conditions of multistationarity and other nontrivial behavior. *Chaos: Interdisciplinary J. Nonlinear Sci.*, **11**: 170.
- Werhli, A. V., and Husmeier, D. (2008). Gene regulatory network reconstruction by Bayesian integration of prior knowledge and/or different experimental conditions. *J. Bioinform. Comput. Biol.*, **6**(3): 543–572.
- Wikle, C. K. (2004). Brief introduction to (hierarchical) bayesian methods, AMS Short Course, January.
- Zhang, S. Q., Ching, W. K., Ng, M. K., and Akutsu, T. (2007). Simulation study in probabilistic boolean network models for genetic regulatory networks. *Int. J. Data Mining Bioinform.*, **1**: 217–240.
- Zou, M., and Conzen, S. D. (2005). A new dynamic Bayesian network (DBN) approach for identifying gene regulatory networks from time course microarray data. *Bioinformatics*, **21**(1): 71–79.

**TRENDS AND STATISTICAL
CHALLENGES IN GENOMEWIDE
ASSOCIATION STUDIES**

Ning Sun and Hongyu Zhao

*Department of Epidemiology and Public Health, Yale University School of
Medicine, New Haven, Connecticut, USA*

12.1 INTRODUCTION

Genomewide association studies (GWAS) conduct genetic association analysis between clinical traits and genetic variations at the genome level through the use of high-throughput platforms. Recent successes in the GWAS approach can be attributed to three major advances in the human genetics field in the past several years: (1) the completion of the HapMap project whose goal is to identify and survey genetic variations across distinct populations and to study the associations between these variations at the population level (International HapMap Consortium, 2003); (2) the development of reliable and affordable high-throughput genotyping platforms that are capable of querying genetic variations at more than one million sites across the genome; and (3) the formation of large consortia to study a number of common diseases. In this chapter, we first briefly review the key concepts, resources, and technologies that drive GWAS. We then discuss major statistical challenges presented to us from GWAS. We conclude this chapter with a brief discussion of related topics and the future challenges in this area. Because there is a vast literature on these topics, we are only able to cite a small number of relevant papers in this chapter.

**12.2 ALLELES, LINKAGE DISEQUILIBRIUM,
AND HAPLOTYPE**

One major objective in human genetic studies is to identify genetic mechanisms underlying a clinical trait of interest, for example, cancer, hypertension, and obesity. This often starts with the localization of a chromosomal region associated with the trait. The basic principle is that if a region is associated with a phenotype and this

region contains variations across different individuals impacting the phenotype, for example, coding variations, the trait distribution for all those individuals having one particular form of the variation, also called *allele*, will differ from that for those individuals having another allele. Therefore, this region may be inferred by examining statistical associations between trait values and genetic variations. Note that no matter how critical a region is in influencing the trait, if it does not have any genetic variations in the population, this association approach will not work. There are different types of genetic variations commonly studied in human genetics. Recent studies have focused on *single-nucleotide polymorphisms (SNPs)* due their abundance and reliable high-throughput genotyping platforms for them. SNPs refer to specific DNA nucleotide variations at a given site in the genome. A SNP usually has two alleles; the more common one is called the major allele whereas the other one is called a minor allele. It is estimated that about 10 million SNPs exist in humans with a minor allele frequency of 1% or higher (International HapMap Consortium, 2003). Other types of genetic variations include microsatellites and structural variations that involve larger segments of DNA. In the most ideal world, a human geneticist would like to have data consisting of all the genetic variation information in the genome and all the phenotypic information from all the people on this planet. Then for a given variable site in the genome, we would examine whether there is an association between a specific variation and a trait of interest. However, it is neither feasible nor necessary to collect all the variation information if our primary goal is to identify common variations, and therefore regions covering these variations, underlying a trait. This is because it will still be prohibitively expensive in the foreseeable future to gather such data plus there is enough redundancy among common variable sites in the genome that a subset of these markers will provide adequate information for association analysis.

The dependency among markers is referred to as *linkage disequilibrium (LD)* by population geneticists. In a homogeneous population, LD exists only for markers in close proximity. Consider two nearby SNPs on the same chromosome, denote the two alleles for the first SNP by A and a and those for the second SNP by B and b . Let the allele frequencies of A and a be p_A and p_a , and those for B and b be p_B and p_b . A *haplotype* refers to the set of alleles on the same chromosome. In the case of two SNPs, a haplotype can have four possible allele combinations: AB , Ab , aB , and ab . If the two markers are independent, the proportion of the chromosomes in the population carrying haplotype AB , Ab , aB , or ab should be $p_A p_B$, $p_A p_b$, $p_a p_B$, or $p_a p_b$, respectively. However, it is commonly observed that if the two markers are very close to each other, for example, 10 kb (10,000 base pairs) apart, there is often a departure from these probabilities. In the most extreme case, only two haplotypes (e.g., AB and ab) are observed. This case is called *perfect LD*. In this case, the marker allele at one marker has complete prediction power of the allele at the other marker. It is apparent that in this case studying one marker is equivalent to studying the other marker, so there is no need to type both markers. A less extreme scenario is that only three of the four possible haplotypes are observed, for example, ab is absent in the population. This case is called *complete LD*. Although one marker cannot totally replace the other marker, based on the observed genotype at one marker, we can still gather valuable information about the other marker. There are

different ways to measure the degree of dependence or the degree of such gained information. Two commonly used measures are r^2 and D' , although many other measures have been introduced in the literature. For a concrete example, if the haplotype frequencies are 0.3, 0.2, 0.2, and 0.3 for AB , Ab , aB , and ab , respectively, it is easy to see that the expected haplotype frequencies when there is no LD are 0.25 for all four haplotypes, because each allele at the two markers has an allele frequency of 0.5. The difference between the observed and expected haplotype frequency is $D = 0.05$ and r^2 is $D^2/p_A p_a p_B p_b = 0.05^2/0.5^4 = 0.04$ and $D' = 0.05/0.5^2 = 0.2$. See Devlin and Risch (1995) for a comprehensive review on various measures of LD. In GWAS, r^2 is more commonly used as it has direct relevance to the statistical power of a study. For example, let us assume that the first marker is truly associated with a trait. If this marker is studied, assume that a total sample size of N individuals is needed to achieve a desired statistical power at a given statistical significance level. However, if the second marker is studied instead of the first one, and the degree of dependence between the two markers is measured by r^2 , then it is apparent that a larger sample size is needed to achieve the same power because we are now studying a surrogate marker. It can be shown that the sample size needed is N/r^2 , making r^2 a useful measure to quantify the loss of information. For the same set of haplotypes, the values of different LD measures can be very different as they are designed to capture different aspects of the dependency between the two markers. For example, if two markers have unequal allele frequencies, even when one haplotype is totally absent, the value of D' can be 1 yet the value of r^2 can be anywhere from 0 to 1, depending on the allele frequency differences between the two markers.

12.3 INTERNATIONAL HapMap PROJECT

Because of the LD among the markers, a subset may be informative enough to allow an accurate inference of the genotypes of the markers not included in the subset. However, the dependency patterns are both population and chromosomal region specific. In general, there is higher dependency in European and Asian populations than that in African populations. At the chromosomal level, the degree of LD is closely related to the local recombination rates, but many other factors also play a role in affecting the level of LD in a region. As a result, it is not possible to predict local LD patterns in a given population from theoretical modeling. The *HapMap* project was launched for the precise purpose of collecting empirical data on common genetic variations in diverse human populations to study the LD patterns in the whole genome. In the Phases I and II of the HapMap project, more than three million markers were typed on four HapMap samples, including 30 family trios (both parents and one offspring) in a European American population, 30 trios from a Yoruban population, 45 unrelated individuals from Tokyo, and 45 unrelated individuals from Beijing (International HapMap Consortium, 2005, 2007). The empirical data revealed the complex and distinct LD structures in different populations. For common variants, it is estimated that 500,000 SNPs are sufficient to serve this purpose for the

European population, but a larger set of markers is needed for the African population. The markers in this subset are called *tag SNPs*. A number of methods have been proposed to identify tag SNPs (e.g., an early review by Stram, 2005; Howie et al., 2006; Eyheramendy et al., 2007). These methods differ on the criterion used to define an optimal set of markers, either based on diversity, information, or predictive power of the tag SNPs on untagged SNPs. There are generally a number of parameters that need to be specified, for example, the r^2 cutoff between the tag SNPs and untagged SNPs for tag SNP selections. In general, there exist many more or less equivalent sets of tag SNPs that capture the same amount of the information for a given population due to the dependency of the markers. It has been found that the tag SNPs identified from a given population are generally transferable to other populations in the same broad geographical (e.g., continental) area (e.g., De Bakker et al., 2006; Xing et al., 2008).

12.4 GENOTYPING PLATFORMS

GWAS would not have taken off if the cost of conducting such a study is prohibitively expensive or the data quality is not sufficiently high. Currently there are two major commercial high-throughput genotyping platforms, *Affymetrix* (Matsuzaki et al., 2004) and *Illumina* (Steemers et al., 2006). Their earlier products had between 10,000 and 100,000 SNPs and low genome coverage. The latest products can genotype more than one million SNPs and *copy number variations* (CNVs). See Table 12.1 for a brief summary of the most commonly used current products by these two companies. The markers in the Illumina platform are selected through the tag SNP approach, whereas the Affymetrix markers represent roughly random markers in the genome. Therefore, for the same number of markers, Illumina offers better coverage than Affymetrix. The cost for genotyping the whole genome using either platform is less than \$1000 for their latest chips, making it possible to gather genotype information for thousands of individuals at reasonable cost. Both platforms have very high accuracy. This is critical as even a very small proportion of markers with errors

TABLE 12.1 Commercial Array Platforms

Platform	Product name	Marker information
Illumina	Infinium HD Human1M-Duo (two samples/chip)	~1.2 million markers Median spacing: 1.5 kb CNV regions: 4274
	Human610-Quad (four samples/ chip)	~621,000 markers Median spacing: 2.7 kb CNV regions: 3938
Affymetrix	SNP Array 6.0	~907,000 SNPs and ~946,000 probes for CNVs

may lead to too many false-positive results when more than one million markers are investigated and such false-positive results may easily dominate true signals.

12.5 OVERVIEW OF CURRENT GWAS RESULTS

Another driving force for recent GWAS success stories is the realization that thousands or tens of thousands of subjects may be needed to identify genetic variants having small to moderate effects on common diseases. In order to achieve such a sample size, investigators from different studies have joined forces to form large consortia for different disease areas, for example, Barrett et al. (2008) for Crohn's disease and Zeggini et al. (2008) for type II diabetes. These developments in public resources, technologies, and collaborations have led to many disease–gene association discoveries in recent years. A highly successful example is the Wellcome Trust Case Control Consortium (2007) where seven common diseases were studied through a total of 14,000 cases and 3000 shared controls. Currently, there is almost no single week going by without one or more replicated positive findings from GWAS. An updated list of SNPs/genes identified associated with various diseases is maintained by Hindorf and colleagues (2008) at www.genome.gov/26525384. In Table 12.2, we list several representative successful GWAS studies and their sample characteristics. As can be seen, these studies all have large discovery and replication samples, enabling the investigators to identify regions carrying weak to moderate genetic risks.

A GWAS can analyze data collected through different epidemiological designs. Table 12.3 summarizes three commonly used designs (including their advantages and limitations) that have all been used to date to identify genes associated with a trait of interest. The case–control design is by far the most commonly used one due to its relatively low cost and ease of sample collection. A typical GWAS analysis pipeline usually consists of genotype calling from all the markers typed on a given platform,

TABLE 12.2 Several Successful Disease Studies Based on GWAS Approach

Phenotype	Initial samples	Replication samples	Number of loci identified	Significance threshold	Reference
Crohn's disease	3230 cases, 4829 controls	2325 cases, 1809 controls, 1339 affected trios	30 regions	5×10^{-8}	Barrett et al., 2008
Bipolar disorder	1098 cases, 1267 controls	4387 cases, 6209 controls	<i>ANK3</i> and <i>CACNA1C</i>	7×10^{-8}	Ferreira et al., 2008
Height	30,968 individuals	8541 individuals	27 regions	1.6×10^{-7}	Gudbjartsson et al., 2008
Type 2 diabetes	4549 cases, 5579 controls	24,194 cases, 55,598 controls	6 regions	5.0×10^{-8}	Zeggini et al., 2008

TABLE 12.3 Different Study Designs for GWAS

	Case–Control	Cohort	Family
Design	Genotype/allele frequencies are compared between cases and controls ascertained based on their disease status. Genetic variants showing statistically significant differences between the cases and controls are followed up with replication and molecular studies.	Genotype/allele effects assessed on a set of individuals followed over time.	In addition to comparing genotype/allele frequencies between cases and controls, this design allows the study of transmission from heterozygous parents to offspring. Nonrandom transmission implies disease association.
Advantages	Relatively easy sample collection. Efficient for uncommon diseases.	Extensive longitudinal information is available for genetic studies. Allows genetic risk estimate. Less recall and other biases.	Transmission-based tests are robust to population stratification, a major source of bias in genetic association studies.
Limitations	Confounding factors may bias study results, with genetic background heterogeneity as the most studied potential confounder. Limited information available to study traits other than the primary disease status.	Expensive, especially for rare/uncommon phenotypes. Sample heterogeneity in genetic background.	Most expensive in sample collection.

removal of low-quality samples and markers, imputations of untyped markers, and association analysis based on single markers and haplotypes. Figure 12.1 presents a typical output from a GWAS analysis. Figures 12.1*b*, *c* present single marker and haplotype association results, and the LD structure in this region is shown in Figure 12.1*d*. Although most published papers present results in clean forms, GWAS data do present many statistical and computational challenges, and some have been well addressed while others need rigorous development. In the following, we will review the main statistical problems that arise in GWAS. We will focus on the nature of these problems instead of the detailed statistical techniques that have been developed to address these problems. We hope that the readers will find these problems of sufficient scientific and statistical interests and follow up on more statistical oriented papers through the listed references.

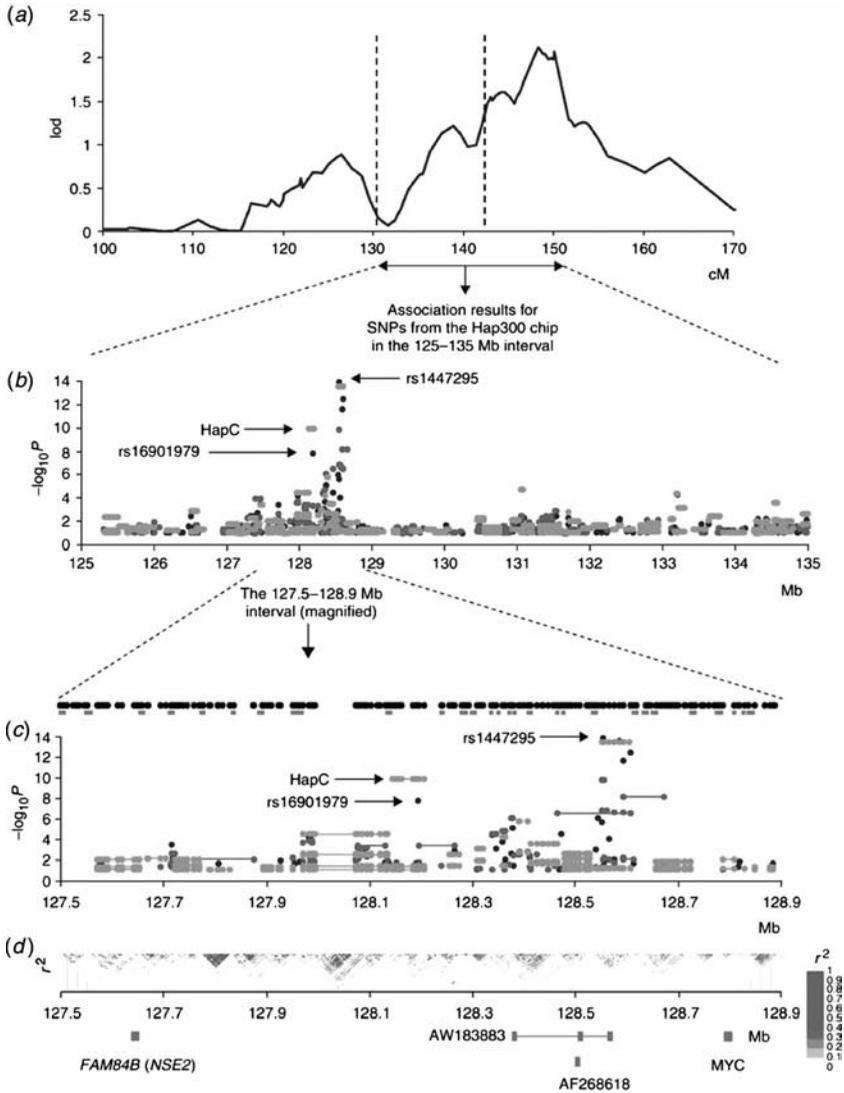


Figure 12.1 GWAS results from Gudmundsson et al. (2007). (a) Linkage scan results for chromosome 8q from 871 Icelandic individuals with prostate cancer in 323 extended families. (b) Single-marker (black circles), two-marker (gray circles), and LD-block haplotype (gray circles) association results for all Icelandic individuals with prostate cancer ($n = 1453$) using 1660 SNPs from the HumanHap300 chip along with marker rs16901979, distributed over a 10-Mb region. Shown are P -values < 0.1 , corrected for relatedness. (c) Association results from (b), shown in greater detail, for a 1.4-Mb interval on 8q24.21. Filled black circles represent all 225 SNPs used in the association analysis of the 1.4-Mb interval, and the gray boxes denote the recombination hot spots. (d) Pairwise correlation coefficient (r^2) from the CEU HapMap population for the 1.4-Mb region in (c); the boxes at the bottom indicate the location of the *FAM84B* (*NSE2*), *AF268618* (*POU5FLC20*), and *MYC* (*c-MYC*) genes and the AW183883 EST previously described. A scale for r^2 is shown at right. (Reproduced with permission from Nature Publishing Group.)

12.6 STATISTICAL ISSUES IN GWAS

12.6.1 Data Preprocessing and Quality Control

As shown in Table 12.1, Illumina and Affymetrix provide the two most commonly used genotyping platforms for GWAS. Each platform can study both SNPs and CNVs. For SNP, the presence or absence of a specific allele is queried through specifically designed probes. CNVs are studied through similar means. The raw data are intensity levels that are used to make genotype calls for each individual. Similar to results from microarray gene expression experiments, the measured intensity levels may vary from chip to chip and normalization is necessary to make meaningful comparisons across chips. A number of methods have been developed for both platforms (e.g., Nicolae et al., 2006; Rabbee and Speed, 2006; Teo et al., 2007; Xiao et al., 2007). One common theme of these approaches is the use of clustering methods to define genotypes. Figure 12.2 shows that different SNPs may have different qualities where the clusters may be very clear for some markers but difficult to separate out for other markers. This is because the intensity levels from people with different genotype tend to form distinct clusters to allow the inference of individual genotypes. For SNP markers, when there are no clear clusters of the three marker genotype groups, it is probably more appropriate to model the association between the intensity data and phenotype data.

Although technologies keep advancing and data quality keeps improving, data are never perfect. A lack of attention to data quality may lead to false-positive results that can be avoided through better quality control. It is necessary to remove samples that have poor quality and markers that have poor calling rates. If a specific biological sample has an unusually high missing genotyping rate, it is better to remove this sample from further analysis. Similarly, for a specific marker, if an unusually large number of individuals have poorly called genotypes, it should be removed from further analysis. Different samples may have different qualities and this needs to be taken into account to avoid bias due to nonrandom missing genotypes. Statistical methods have been proposed to address this (e.g., Plagnol et al., 2007).

After the above two steps to remove questionable individual samples and markers, one common procedure used to detect potential errors is to test whether there is a violation of *Hardy–Weinberg equilibrium* (HWE) (e.g., Cox and Kraft, 2006). In a population satisfying HWE, the two alleles on the two chromosomes carried by one individual are independent of each other. As a result, for a SNP with allele frequencies p and q , the three genotypes should have frequencies p^2 , $2pq$, and q^2 , respectively. A significant departure from HWE at a given marker can be due to a number of reasons, including the presence of sample heterogeneity in genetics background, sample ascertainment, genotyping errors, or purely statistical chance. In human genetics studies, *sample heterogeneity* is a great concern, especially for studies collecting samples from multiple sites, for example, several metropolitan areas in the United States. As discussed below, a number of statistical strategies have been proposed to address sampling heterogeneity. If sampling heterogeneity is extensive, this will be reflected through departures from HWE for a large number of markers. Sample ascertainment can also lead to departure from HWE. For example, in a case–control study, if a genetic marker has a significant impact on disease phenotype, then it is

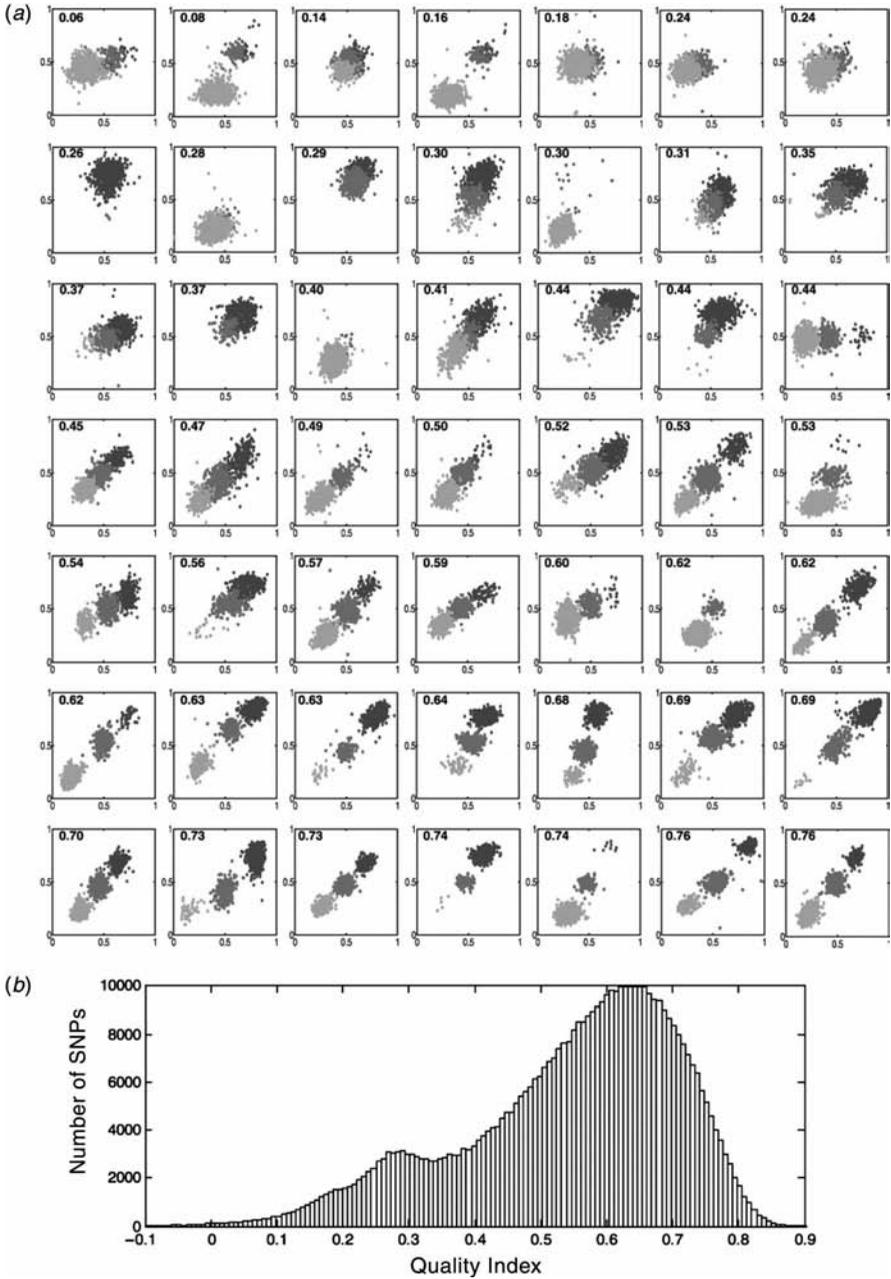


Figure 12.2 Genotype calls on 49 SNPs of various quality indices and the distribution of quality index shown in Hua et al. (2007). (a) Each scatterplot corresponds to one SNP. The SNPs are ordered according to the quality index. Three genotypes are denoted as black, dark gray, and light gray points. (b) The Histogram of quality index on all SNPs.

likely that HWE will fail in the case population and possibly also in the control population. In fact, Hardy–Weinberg disequilibrium (HWD) is a tool for mapping genes associated with disease (e.g., Jiang et al., 2001). In GWAS, the most likely reason for departure from HWE is genotyping errors and markers showing statistical significant departure are generally excluded from further analysis. In addition to the above three reasons, sampling variation may also lead to departure from HWE. For example, even when all the markers satisfy HWE, if 500,000 independent markers are screened at the statistical significance level of 10^{-4} , 50 markers will show significant departures and therefore are excluded from further analysis. Among the possible causes of departure from HWE, we would like to avoid removing markers truly associated with the disease phenotype, for example, true signals. One way to distinguish this from genotyping errors is to examine whether there is HWD across a set of closely spaced markers as a true signal is likely reflected through departure from a set of markers that are in linkage disequilibrium with each other, whereas genotyping errors likely lead to isolated departures. In addition, the allele frequencies in the sampled individuals can be compared to those from a closely related reference population where such information is available; for example, compare a European American sample versus the CEU sample from the HapMap project. If there are great discrepancies between the two populations, genotyping errors likely have occurred.

12.6.2 CNV Analysis

In addition to SNPs, CNVs may also play a role and also serve as markers for common diseases (e.g., Sebat et al., 2007; Walsh et al., 2008). There is an increasing number of studies surveying the distribution and features of CNVs in human populations (e.g., Tuzun et al., 2005; Komura et al., 2006; Redon et al., 2006). Although most SNP markers can now be accurately called, CNV data are much more difficult to deal with. Similar to CNV data, the raw data for CNV are summarized as intensity levels, with regions having higher CNV counts likely presenting higher intensity levels. Therefore, if we move along the chromosome and there is a very high density coverage of the genome, we may be able to identify CNVs through the change of intensity levels. However, the exact state of a specific CNV can be many, making it difficult to infer the exact state. Current methods for CNV calling can be broadly defined into two approaches, the change point approach and the hidden Markov model (HMM) approach. For the change point approach, the average intensity in a putative CNV region is compared to its neighboring regions and this is often done in a sequential order (Olshen et al., 2004). The HMM approach models the underlying CNV state as a Markov chain with the states representing the CNV counts (e.g., Colella et al., 2007; Wang et al., 2007; H. Wang et al., 2008). The observed data are the observed intensities. Other approaches are also available, for example, the fused lasso approach by Tibshirani and Wang (2008).

For both SNP and CNV data, data quality can be improved if we have data available from multiple individuals in the same family through checking the consistency of called genotypes across all family members. Mendelian inheritance can be checked for each marker to identify markers not consistent with Mendelian inheritance. For example, if the genotypes of the parents and their child are 11, 12, and 22, then

either there is a genotyping error or there was a mutation during one of the two parental meioses. This approach is less useful for CNVs. For example, if the CNV counts for the parents are 3 and 3, then the offspring can have his or her own CNV anywhere between 0 and 6, depending on the distributions of the CNVs on the two chromosomes in the parents.

Although intensity data can be used for association analysis, from this point on, we assume that each individual's genotype has been called and the association analysis is done between the trait of interest and the called genotypes, either SNPs or CNVs.

12.6.3 Multiple-Comparison Adjustments

In the simplest case of GWAS data analysis of examining associations between a trait and single markers, we will be conducting hundreds of thousands of tests. If we apply the usual statistical significance cutoff, for example, 0.01, to declare all SNPs with a p -value less than this cutoff to be significant, we will end up with thousands of positive results purely by chance even when there is no association between the trait and any of the markers. Therefore, a more stringent criterion is adopted in GWAS to reduce the rate of false positives. The importance of appropriately interpreting statistical significance was thoroughly discussed by Wacholder et al. (2004). Suppose that there are 500,000 markers studied. If *Bonferroni correction* is used, a per-marker threshold of $0.05/500,000 = 10^{-7}$ is needed to control the overall familywise error rate at the 0.05 level. Because Bonferroni can be a conservative approach when some of the markers are dependent, *permutations* are often used as a more effective empirical approach for assessing the false-positive results. In a standard permutation procedure, the trait values among all the study subjects are permuted and associations between the permuted traits and markers are reexamined. Each permutation will result in a set of permuted p -values. After a large number of permutations, we will end up with a matrix of permuted p -values where each row represents a specific permutation whereas each column corresponds to a marker. Then the question is how to compare the observed p -values to the permuted p -values to establish an appropriate threshold for declaring statistical significance. One possibility is to select the smallest p -value from each set of permuted p -values (i.e., from each column) and form a vector of the smallest p -values as the empirical distribution for the smallest p -value. Then the observed p -values can be compared against the fifth percentile (or any other percentile) of the empirical distribution, and all the markers having a p -value smaller than that value will be deemed statistically significant at that statistical level. This approach is in line with controlling the *familywise error rate*. An alternative approach is to select a statistical significance level and examine how many of the markers have p -values less than that level in the observe data. For each permuted dataset, the same threshold can be applied to count how many markers have p -values less than the threshold. If 10 of the observed p -values are less than the threshold, and on average only 5 of the permuted p -values are less than the threshold, this indicates that it is likely that half of the 10 may be chance findings. This 50% may be interpreted as a *false discovery proportion*. A similar issue also arises in functional genomics studies where the expression levels of tens of thousands of genes are measured at the same time, and statistical inference needs to be made as to which genes have different expression levels across a set of

conditions. The concept of *false discovery rate* control (Hochberg and Benjamini, 1990) was indeed popularized in gene expression studies. Both familywise error control and false discovery rate control are used with some preference for the false discovery control as it is commonly believed that familywise error control tends to be too conservative. Although permutation is very intuitive, there are some practical limitations. First is the sample heterogeneity issue that will be discussed in the next section. One implicit assumption underlying permutation is that the samples are exchangeable. However, when a sample consists individuals from diverse genetic background, this assumption likely fails, making it necessary to have alternative permutation procedures (Kimmel et al., 2007). Second, permutation cannot be readily applied to more complex sample structures, for example, samples with related individuals. Third, permutation is computationally extensive. Computationally less expensive alternatives may work well (e.g., Sabatti et al., 2003; Dudbridge and Koeleman, 2004; Dudbridge and Gusnanto, 2008). As most GWAS need to be followed up by replication studies, it seems that the exact cutoff point may not be as important as to put the markers/genes in the right order in terms of biological significance.

12.6.4 Sample Heterogeneity

One major concern for the validity of any GWAS is whether there is sufficient genetic heterogeneity in the sampled individuals so that the observed association between the phenotype and a marker is a mere reflection of such heterogeneity present in the samples. A classical example is the genetic association between type 2 diabetes and immunoglobulin gene *Gm* (Knowler et al., 1988). Among residents of the Gila River Indian Community in Arizona, diabetes was associated with the haplotype *Gm*. Although there was a significant association between disease status and genotype at this gene, it was found that the genotype reflects the European ancestry of the sampled individuals. This association no longer existed among ethnically homogeneous subjects.

In another study to document the potential impact of genetic heterogeneity, Campbell and colleagues (2005) found a statistically highly significant association between height and lactase gene from a European American sample. In this case, the allele frequency of this gene has a clear north–south cline in Europe, consistent with the height cline in Europe. This population stratification effect is well known in human genetics and of great concern for any population-based association studies. This concern was the main force for the very popular *family-based association* study design (Spielman et al., 1993) where the nontransmitted allele from a parent serves as the control for the transmitted allele, which is considered as cases.

Devlin and Roeder (1999) were the first to point out that this confounding problem may be addressed if there are many markers available on each individual and the overall genetic heterogeneity roughly has the same effects on all the markers. They proposed the *genomic control* method to realize this idea. The basic approach is as follows. First, a standard association test is conducted between the trait and each individual marker. Under the null hypothesis of no association, the distribution of all the test statistics should follow a specific distribution, for example, a chi-square distribution for a test based on a contingency table. With hundreds of thousands of markers, the asymptotic distribution should provide a very good

approximate distribution for the test statistics across all the markers because it is likely the majority of the markers are likely not associated with the trait; that is, the null hypothesis is true. However, in the case of sample heterogeneity, the test statistics will be inflated and lead to a departure from the expected distribution. The main idea of the genomic control approach is to rescale each test statistic by the same constant factor so that the mean or median of the distribution of the rescaled statistics is the same as that of a reference distribution. This approach changes the values of all the test statistics by the same factor but does not change the order of the markers. The other approach popularized by Pritchard and colleagues (2000a, b) is to use genotype data to infer the underlying structure in the sample, commonly called *structure analysis*. The most basic set-up is to assume that there are a total of K subpopulations in the sample. Each population has its own allele frequencies and satisfies HWE at each marker. Each individual comes from one of the K populations. The inferential goal is to use all the genotype data to group individuals from the same subpopulation together and then conduct association analysis taking such inferred subpopulation information into account. A more refined version of this approach allows each individual having different chromosomal segments from different subpopulations and markers can be in linkage disequilibrium with each other. This *structure approach* is very powerful when the samples consist of individuals from distinct genetic backgrounds. However, it has difficulty in dealing with samples consisting of individuals from closely related populations, for example, different populations within Western Europe. More recently, *principal-component analysis (PCA)* has been adopted as a computationally efficient and practically effective approach to dealing with sample heterogeneity (Zhu et al., 2002; Price et al., 2006; Novembre and Stephens, 2008). Although this method has a long history in population genetics, it is popularized with the EIGENSTRAT package (Price et al., 2006). These different approaches are complementary to each other. For example, in a given study, the structure-based approach can first identify apparent structure in the samples and the PCA-based analysis can be performed to remove more subtle effects in the sample. Lastly, the genomic control method can be used to both examine whether there is a significant departure from the expected distribution for the test statistics and rescale the test statistics if necessary. Other methods are also available to stratify the samples to form more homogeneous groups for association testing (e.g., Epstein et al., 2007).

12.6.5 Imputation Methods

Despite being high density, the SNPs on most genotyping platforms represent only a subset of the known SNPs, and there is no a priori reason to believe that the SNPs not genotyped are less relevant biologically. Therefore, it would be highly desirable if an association analysis can also be conducted between disease phenotype and untyped SNPs. This would not be possible in typical statistical analysis when there is no association between the typed and untyped SNPs and, even when there is association, no detailed knowledge on the exact relationships. However, much has been learned about the relationships between the typed and untyped SNPs from the HapMap samples and other samples, and such knowledge can be used to sometimes accurately

predict the genotypes of the untyped SNPs, for example, if the following haplotypes are observed in the population: 111, 122, 211, and 222. Then knowing the genotypes at two of the markers completely determines the genotype of the other marker. If only single-marker association is of interest here, then in addition to single-marker tests at the two typed markers, we can also study associations between disease and the third, untyped, marker. This is the basic idea underlying a number of *imputation methods* for GWAS data (e.g., Nicolae, 2006; Marchini et al., 2007). These methods differ in how they call the untyped markers and how the uncertainties in inference of untyped markers are used in association analysis.

12.7 HAPLOTYPE ANALYSIS

Haplotypes refer to the combination of alleles on the same chromosome, and haplotype analysis is the key component in the HapMap project. We will discuss first haplotype analysis in a set of random samples and then haplotype analysis in the GWAS setting. A major objective in haplotype analysis of a randomly sampled individual is the inference of the haplotypes this individual carries on his or her two chromosomes. This is necessary because most genotyping technologies can only gather genotype data at individual markers separately, so any information about haplotypes has to be inferred from the observed data from individual markers. Let us consider a very simple scenario where a person has genotype 12 at the first marker and genotype 12 at the second marker. Then there are two possible haplotype scenarios that are consistent with the observed genotypes: this person can either carry 11 and 22 on two chromosomes or 12 and 21 on the two chromosomes. If we only have data from this person, it would be impossible to infer the true haplotypes. However, this ambiguity can be resolved with information from related people or from a set of unrelated people. For example, if we have information from this person's parents, whose genotypes are (11)(11) and (12)(12), then one parent can only transmit haplotype 11 to this person so the haplotypes have to be 11 and 22. On the other hand, if in a sample of unrelated individuals there are only three different genotypes across the two SNPs—(11)(11), (22)(22), and (12)(12)—then there is no evidence for the presence of 12 or 21 in this population. This is because otherwise we would observe genotypes other than these three possibilities. Based on this information, we would infer that it is very likely that this person has haplotypes 11 and 22. More rigorously, we can estimate the probabilities of the two haplotype pair probabilities based on the following statistical model. Let p_{11} , p_{12} , p_{21} , and p_{22} denote the haplotype proportions for haplotypes 11, 12, 21, and 22, respectively, in the general population. Assuming HWE, we can calculate the probability of each possible haplotype pair a person carries. For example, the probability that a person has 11 and 22 as haplotypes is $2p_{11}p_{22}$, and the probability that a person carries 12 and 21 as haplotypes is $2p_{12}p_{21}$. If only individual markers are observed, these two possibilities cannot be distinguished, so we have the probability that we observe (12)(12) as $2p_{11}p_{22} + 2p_{12}p_{21}$. For all other genotypes, it is possible to resolve the haplotypes. Therefore, for a sample of individuals, we can obtain the likelihood for the observed genotypes at the two SNPs. Then we can use the maximum-likelihood estimates to infer the

haplotype probabilities. These estimates can be achieved via the expectation—maximization algorithm where the haplotype pair a person carries are treated as the complete data. This general principle can be extended to deal with more than two markers. This approach can allow the inference of haplotypes for a small number of markers. However, the computational burden can be substantial when there are a large number of markers as the possible number of haplotypes is an exponential function of the number of markers. In addition, the simple application of the likelihood principle treats each haplotype as a separate parameter and does not consider the similarities of different haplotypes. Under a *coalescent population genetics model*, haplotypes tend to be more similar to each other. PHASE, developed by Stephens et al. (2001), employed a Bayesian framework to incorporate haplotype similarity and allow sequential updating to deal with a large number of markers. However, PHASE may be slow for an extremely large number of markers and alternative methods [e.g., fastPHASE by Scheet and Stephens (2006) and Mach by Li and Abecasis (2006)] can be used for faster inference.

Now consider a case–control study where we are interested in inferring whether there is an association between the disease status and haplotypes. There are two potential benefits of haplotype analysis versus single-marker analysis. First, haplotypes can capture untyped markers that are more directly related to the trait. Second, even when all the markers are typed, haplotypes can more effectively model interactions among different sites. As for haplotype association analysis for a case–control study, one intuitive approach would be to compare the haplotype frequencies between the cases and controls to see whether there is a significant difference between the two groups. This can be done through a likelihood ratio test. Under the null hypothesis, the haplotype frequencies are assumed to be the same in the case and control groups. Under the alternative hypothesis, the two groups can have different haplotype frequencies. If an HWE assumption is made for both hypotheses, then we can calculate the likelihood for each hypothesis and the log-likelihood ratio should follow a chi-square distribution under the null hypothesis where the number of the degrees of freedom is equal to the number of haplotypes in the region. One issue with this simple approach is that if there is an association between the disease status and haplotypes, the HWE assumption generally does not hold in the case group nor the control group, so the model may only allow the exploration of a very limited space for the alternative hypothesis. A more rigorous way to set up the likelihood is to make an explicit assumption about the association between the disease risk and haplotypes. For example, we can assume a logistic model relating disease risk to an additive model for haplotype effects. Based on this model, it is possible to derive the likelihood under a prospective design, where the samples are followed over time and their disease status is recorded. However, in a case–control study, the sampling is usually based on the disease status of the sampled individuals, so the prospective likelihood is not correct. Various approaches have been developed to appropriately model this relationship (e.g., Epstein and Satten, 2003; Lin and Zeng, 2006).

Similar to the difference between the likelihood-based haplotype inference methods and coalescent model-based methods, the above approach does not take similarity among haplotypes into account. It seems reasonable to assume that if a gene is associated with disease, similar haplotypes may have similar effects on disease

outcome. Very early, this idea was incorporated in association analysis where haplotypes are partitioned into haplotype groups according to their similarities, and association is only examined at the group level so that fewer degrees of freedom are used (Templeton et al., 1987). A number of more refined statistics have been developed along this path (e.g., Tzeng et al., 2006).

For any haplotype method, the computational burden may be high if it is adapted to the genome setting. Furthermore, there is always the issue of how many and which markers are included in a haplotype analysis. The methods proposed by Browning and Browning (2007) and Huang et al. (2007) are promising as they seem to be scalable to GWAS data and were developed under sound statistical principles.

12.8 HOMOZYGOSITY AND ADMIXTURE MAPPING

Inbreeding refers to mating between related individuals, for example, marriages between first cousins, and is a common practice in certain populations. For offspring resulting from such marriages, segments of a chromosomal region can be totally homozygous as they are copies of the same genetic material from the same ancestral chromosome. With genomewide data, such regions can be identified relatively easily. These populations offer a great opportunity to identify regions associated with disease risk. Even different genetic variants from the same region are associated with disease risk if the disease mode is recessive, that is, a person will develop disease if he or she inherits two copies of risk alleles. Homozygosity mapping looks for regions in the genome with an excess level of homozygosity and the putative regions are regarded as good candidates harboring disease alleles (e.g., Miyazawa et al., 2007).

Admixture mapping is a method that is applicable to admixed populations (e.g., African Americans). It is estimated that up to 20% of DNA in an African population has European ancestry. If two ancestral populations have differences in disease risks (e.g., high risk in Europeans and low risk in Africans), then a person in the admixed population with more European ancestry tends to have higher risk than a person with less European ancestry. With GWAS data, it is often possible to assign specific regions to a particular population ancestry (e.g., Tang et al., 2006). For a given region, if the diseased individuals have more European ancestry than their overall genomewide average, then this region may be implicated to be associated with disease risk.

12.9 GENE \times GENE AND GENE \times ENVIRONMENT INTERACTIONS

Haplotype analysis explicitly considers more than one marker at a time with the idea that haplotypes may more effectively capture association signals between the trait of interest and a region, leading to better statistical power for association detection. However, haplotype analysis still focuses on a small physical region in the genome. It is conceivable that more than one gene is involved in disease etiology and a person's disease risk is affected by the joint interactions of many genes as well as environmental

factors. Therefore, it is likely that joint analysis of genetic and environmental risk factors may lead to better disease model and higher statistical power. The main challenge to study $G \times G$ and $G \times E$ interactions is the sheer number of models that need to be considered. For example, with 500,000 markers, a simple exhaustive two-way modeling involving all possible marker pairs will consider more than 10^{11} models, a possible but extremely computationally intensive endeavor. An alternative approach is to conduct single-marker analysis first and then only consider interactions among markers with p -values less than a given threshold. These joint modeling approaches increase the number of models evaluated orders of magnitudes higher, so the gain in power has to be substantial to justify a naive two-dimensional exhaustive search. This issue has been explored in the literature either through simulations or through empirical studies, but the conclusions are not equivocal (e.g., Marchini et al., 2005; Storey et al., 2005). In one extreme case, it is theoretically possible that two markers may only have interaction effects with no observable marginal effects. In this case, exhaustive search is the only way to find the two markers to be associated with the trait. Therefore, which approach is better depends on the unknown model space. Some published studies suggest that the interactions, even if they exist, may be weak or moderate (e.g., Barrett et al., 2008).

12.10 GENE AND PATHWAY-BASED ANALYSIS

In the sections above, we have discussed statistical methods that are used to examine associations between trait and markers (either one marker at a time or haplotypes consisting of multiple markers or joint marker analysis). One critical piece missing is the knowledge that has been accumulated on our understanding of the human genome. The above analyses simply treat each marker as a physical entity, a predictor, of the genome. However, enormous amounts of efforts have been put to annotate the human genome through experimental and bioinformatics studies to identify where the genes and regulatory elements are and how genes function (e.g., ENCODE Project Consortium, 2007). Such knowledge is available in the public domain databases in different formats. For example, the Human Genome Browser (<http://genome.ucsc.edu/cgi-bin/hgGateway>) is one of the most commonly used resources for gene annotation. Many biological pathway databases are available to illustrate how a set of genes/proteins work together to perform a specific biological function. A large number of microarray databases have also been developed so that the expression profiles of different samples (e.g., representing different disease states and responses from different treatments) can be easily queried and analyzed. Protein databases include interacting proteins and protein complexes. There are also databases available directly annotating the functional relevance of genetic polymorphisms. Given such rich information in the literature, it is important to incorporate it in the analysis of GWAS data. For example, instead of considering one marker at a time, we may use the gene as the analysis unit. To do this, we need to associate SNPs with a specific gene. This can be accomplished through the Human Genome Browser where each gene is annotated. To include potential regulatory regions and some inaccurate gene annotation, it is common practice to extend the gene region both upstream and

downstream in order not to lose potentially informative markers. After SNPs are mapped to a given gene, the problem becomes how to assess association between the trait and all the SNPs included. Different statistical methods can be used here, with the simplest one assigning each with a p -value obtained from the SNP that has the smallest p -value. We can also use regression models to assess association between the trait and multiple SNPs where all the SNPs are treated as predictors. Because many SNPs may have high dependency and there can be up to hundreds of SNPs for certain genes, *regularized statistical methods* can be applied to improve the performance of statistical methods (e.g., Malo et al., 2008). Some more complicated methods first reduce the potential high dimensionality to a lower dimensional space through standard statistical tools—for example, PCA (Gauderman et al., 2007) or Fourier transform (Wang and Elston, 2007); then regression analysis can be performed between the trait and a set of new predictors on the lower dimensional space.

The next level of analysis is to examine association at the pathway level, where a set of genes are considered simultaneously. Gene set analysis was first pioneered in functional genomics studies (Mootha et al., 2003), and it is likely that such analysis may also lead to insights on GWAS data. If we can summarize gene-based analysis results with a single value (e.g., p -value), then the null hypothesis to be tested is that the p -values in the gene set of interest show no different patterns from the p -values in all of the genes. This null hypothesis can be tested in different ways. For example, we can combine the p -values through their log transformations. From this analysis, we will be able to obtain a set of pathway-based summary statistics. The question is then how to identify pathways that are significantly different from what we expect if there is no pathway-level association. In this case, we can adopt a false discovery-based approach through permutation analysis. By focusing on the complement pathway, Dinu and colleagues (2007) were able to identify associations between age-related macular degeneration and three genes in this pathway, with the effects of two of these genes being moderate and impossible to identify through a genomewide approach. Interesting pathways were also revealed through pathway analysis of GWAS data on a Parkinson's disease dataset (K. Wang et al., 2008).

One limitation of pathway-based analysis is that not all the genes have been mapped to pathways, and a large proportion of the SNPs cannot be associated with any gene. Other sources of information may be used to associate more SNPs with prior biological information. For example, with thousands of gene expression datasets readily available in the publicly accessible databases (e.g., Stranger et al., 2007), we can create gene expression networks where genes with similar expression profiles are linked (e.g., Ghazalpour et al., 2006). Then we may combine such network information with the observed association signals to better analyze such data. One way to achieve this is through the rationale that genes next to each other in the gene coexpression network are more likely to be both associated or not associated with a given trait. This idea may be realized with different statistical models. As for SNPs not close to any annotated genes, their relative significance can be established from previous studies, such as prior linkage analysis results, comparative genomics studies, and other annotations (e.g., Roeder et al., 2006; Ng and Henikoff, 2003).

12.11 DISEASE RISK ESTIMATES

After an SNP is identified to be associated with a trait, one immediate question is how much effect this SNP has. If the trait is a continuous one (e.g., height), the effect can be quantified as the proportion of variation explained. If the trait is binary (e.g., disease or not), the effect can be quantified as the relative risk or odds ratio. Although it is natural to use the values directly estimated from the observed data that established the relevance of the SNP, it is well known that such estimate tends to be biased upward, that is, the real effect size tends to be overstated (Zollner and Pritchard, 2007). The reason is that, in contrast to most studies where only a limited number of risk factors are studied, GWAS starts with hundreds of thousands of potential risk factors. Suppose that there are a set of SNPs having the same relative risks (e.g., 1.1). Due to sampling variations, some will be observed to have a higher risk and some with lower risk in the observed samples. Because often a stringent threshold is used, not all of these SNPs will be found to be significant. Although the average effect from the observed data will be around 1.1, those found to be significant will be higher than this level, leading to a biased estimate. A number of statistical methods have been proposed to correct for this bias, including both the conditional argument or simulation-based approach (e.g., Zollner and Pritchard, 2007; Ghosh et al., 2008). A more unbiased estimate would be to obtain the estimate in a totally independent sample that has the same general demographical characteristics.

A tightly related question is disease risk prediction using the markers that have been found to be associated with a certain disease. This can be treated as a classification problem where the outcome is disease status and the predictors are the genotypes at the SNPs that have been found to be associated with disease as well as environmental risk factors. Under this set-up, any classical classification methods can be potentially used, for example, logistic regression and regression tree-based methods. However, there is the risk of overestimating the prediction accuracy of the SNP-based approach if the same dataset is used for SNP discovery and risk prediction for the same reason discussed above on disease risk estimates. It is always the best to obtain such estimate in an independent sample of individuals. Another important difference between SNP discovery and risk prediction is that although some SNPs may have high statistically significant association results, the risk prediction utility may be quite limited. This is because when the sample size is large, small effect size can be translated into highly statistically significant results, but these sizes may not provide much added benefit for risk prediction. For example, Gail (2008) compared the prediction based on known risk nongenetic factors versus that based on including genetic markers and found that the benefit is minimal.

12.12 META-ANALYSIS

With the launch of *dbGaP* (Mailman et al., 2007), SNP-level results from numerous GWAS can be readily obtained and jointly analyzed if desired. This opens up the possibility of combined analysis of different datasets for the same disease or similar diseases. *Meta-analysis* is commonly used in genetic studies to pool results from

different studies (e.g., Dong et al., 2008). Although a combined analysis will likely yield more statistical power due to increased sample size, there are a number of complexities that need to be addressed in such analyses. These include: different disease definitions and ascertainment criteria are used across different studies; different study populations are targeted, for example, studies conducted in different countries and continents; and different genotyping platforms are used, for example, Affymetrix versus Illumina. Some of these issues can be dealt with through statistical means, for example, the use of imputation methods to allow the examination of the same set of markers (albeit that imputed marker genotypes can be inaccurate) and the use of mixed-effects models to allow heterogeneity across different studies. However, great care must be taken to examine key epidemiological factors, for example, the distributions of age, gender, and other risk factors. To date, some published studies use imputation methods to pool results from different studies (e.g., Barrett et al., 2008; Zeggini et al., 2008). Ioannidis et al. (2007) found statistical evidence for significant heterogeneity across studies. It may well be possible that different genetic factors may play a role in the same apparent disease phenotype across these factors, making a combined analysis possibly having lower power than individual analyses.

12.13 RARE VARIANTS AND SEQUENCE-BASED ANALYSIS

Current genotyping platforms have proven useful to identify common genetic variants affecting disease risks. However, it is likely that the majority of genetic variants that have effects on disease risk have low frequency in the population or they may be very rare (e.g., Gorlov et al., 2008). In fact, one argument for the use of haplotypes is that they may be capable of capturing rare variants through a combination of common variants. With major efforts to expand the capacity and throughput of sequencing, for example, the 1000 Genomes Project (<http://www.1000genomes.org>), it is conceivable that great progress will be made to develop and apply sequencing technologies so that DNA sequence data are obtained at either the candidate gene level or the whole-genome level for hundreds or thousands of subjects. For such data, the majority of the genetic variants are likely to be rare, for example, observed once or only a few times in the study population. It would be impossible to have any statistical power to establish statistically significant association between any specific variant and disease phenotype due to the very low frequency of these variants. One way to increase statistical power is to pool different variants into different groups according to certain criteria. This approach has proven useful for some disease studies (e.g., Ji et al., 2008), and it certainly will attract major research efforts in the near future.

12.14 CONCLUSIONS

In this chapter, we have covered a number of statistical and computational challenges arising from the analysis and interpretation of GWAS data. Because the literature is

very rich in this area and the field is undergoing major developments, this review only provides key concepts and a selection of references.

Although most published studies to date have analyzed unrelated individuals, real studies often collect samples of different structures. For example, a study may have unrelated individuals, nuclear families, and extended families. In fact, extended families may be enriched for genetic variants affecting disease risk compared to sporadic cases. Therefore, statistical methods are needed to combine different samples. The statistical issues involved include appropriate control of relatedness in the analysis and optimal weighting across the samples to give more weight to sampling units that more likely carry disease variants (e.g., Thornton and McPeck, 2007).

Even though the cost of genotyping has been coming down in recent years, GWAS are still expensive as thousands or more individuals are genotyped. A number of research groups have considered optimal study designs without losing statistical power. These include DNA pooling (e.g., Sham et al., 2002) and staged designs (Elston et al., 2007; Skol et al., 2007).

It is conceivable that we will be flooded with sequence data in GWAS in the near future. There are significant challenges in the processing and quality control of these data. More importantly, these data will generate an unprecedented amount of less common, rare, or unique variants. There is a lacking of statistical tools to analyze such data. But the information offered by such data is extremely rich, and it is important to conduct integrated analysis of data from different sources, such as gene annotations, to make the most use of these data. There is much to learn from many on-going activities in statistics on high-dimensional data analysis and, at the same time, there is no question that GWAS data will motivate many creative statistical approaches in the future.

ACKNOWLEDGMENTS

Supported in part by a Yale YCCI Novel Methodology in Biostatistics Pilot Award, and National Institutes of Health grants R21 GM 84008 and R01 GM59507.

REFERENCES

- Barrett, J. C., et al. (2008). Genome-wide association defines more than 30 distinct susceptibility loci for Crohn's disease. *Nat. Genet.*, **40**: 955–962.
- Browning, S. R., and Browning, B. L. (2007). Rapid and accurate haplotype phasing and missing-data inference for whole-genome association studies by use of localized haplotype clustering. *Am. J. Hum. Genet.*, **5**: 1084–1097.
- Campbell, C. D. (2005). Demonstrating stratification in a European American population. *Nat. Genet.*, **37**: 868–872.
- Colella, S., et al. (2007). QuantiSNP: An objective Bayes hidden-Markov model to detect and accurately map copy number variation using SNP genotyping data. *Nucleic Acids Res.*, **35**: 2013–2125.
- Cox, D. G., and Kraft, P. (2006). Quantification of the power of Hardy–Weinberg equilibrium testing to detect genotyping error. *Hum. Hered.*, **61**: 10–14.
- de Bakker P. I., et al. (2006). Transferability of tag SNPs in genetic association studies in multiple populations. *Nat. Genet.*, **38**: 1298–1303.

- Devlin, B., and Risch, N. (1995). A comparison of linkage disequilibrium measures for fine-scale mapping. *Genomics*, **29**: 311–322.
- Devlin, B., and Roeder, K. (1999). Genomic control for association studies. *Biometrics*, **55**: 997–1004.
- Dinu, V., Miller, P. L., and Zhao, H. (2007). Evidence for association between multiple complement pathway genes and AMD. *Genet. Epidemiol.*, **31**: 224–237.
- Dong, L. M., Potter, J. D., White, E., Ulrich, C. M., Cardon, L. R., and Peters, U. (2008). Genetic susceptibility to cancer: The role of polymorphisms in candidate genes. *JAMA*, **299**: 2423–2436.
- Dudbridge, F., and Gusnanto, A. (2008). Estimation of significance thresholds for genomewide association scans. *Genet. Epidemiol.*, **32**: 227–234.
- Dudbridge, F., and Koelman, P. C. (2004). Efficient computation of significance levels for multiple associations in large studies of correlated data, including genomewide association studies. *Am. J. Hum. Genet.*, **75**: 424–435.
- Elston, R. C., Lin, D., and Zheng, G. (2007). Multistage sampling for genetic studies. *Annu. Rev. Genom. Hum. Genet.*, **8**: 327–342.
- ENCODE Project Consortium (2007). Identification and analysis of functional elements in 1% of the human genome by the ENCODE pilot project. *Nature*, **447**: 799–816.
- Epstein, M. P., Allen, A. S., and Satten, G. A. (2007). A simple and improved correction for population stratification in case-control studies. *Am. J. Hum. Genet.*, **80**: 921–930.
- Epstein, M. P., and Satten, G. A. (2003). Inference on haplotype effects in case-control studies using unphased genotype data. *Am. J. Hum. Genet.*, **73**: 1316–1329.
- Eyheramendy, S., Marchini, J., McVean, G., Myers, S., and Donnelly, P. (2007). A model-based approach to capture genetic variation for future association studies. *Genome Res.*, **17**: 88–95.
- Ferreira, M. A., et al. (2008). Collaborative genome-wide association analysis supports a role for ANK3 and CACNA1C in bipolar disorder. *Nat. Genet.*, in press.
- Gail, M. H. (2008). Discriminatory accuracy from single-nucleotide polymorphisms in models to predict breast cancer risk. *J. Natl. Cancer Inst.*, **100**: 1037–1041.
- Gauderman, W. J., Murcray, C., Gilliland, F., and Conti, D. V. (2007). Testing association between disease and multiple SNPs in a candidate gene. *Genet. Epidemiol.*, **31**: 383–395.
- Ghazalpour, A., et al. (2006). Integrating genetic and network analysis to characterize genes related to mouse weight. *PLoS Genet.*, **2**: e130.
- Ghosh, A., Zou, F., and Wright, F. A. (2008). Estimating odds ratios in genome scans: An approximate conditional likelihood approach. *Am. J. Hum. Genet.*, **82**: 1064–1074.
- Gorlov, I. P., Gorlova, O. Y., Sunyaev, S. R., Spitz, M. R., and Amos, C. I. (2008). Shifting paradigm of association studies: Value of rare single-nucleotide polymorphisms. *Am. J. Hum. Genet.*, **82**: 100–112.
- Gudbjartsson, D. F., et al. (2008). Many sequence variants affecting diversity of adult human height. *Nat. Genet.*, **40**: 609–615.
- Gudmundsson, J., et al. (2007). Genome-wide association study identifies a second prostate cancer susceptibility variant at 8q24. *Nat. Genet.*, **39**: 631–637.
- Hindorf, L. A., Junkins, H. A., and Manolio, T. A. (2008). A catalog of published genome-wide association studies. Available: www.genome.gov/26525384.
- Hochberg, Y., and Benjamini, Y. (1990). More powerful procedures for multiple significance testing. *Stat. Med.*, **9**: 811–818.
- Howe, B. N., Carlson, C. S., Rieder, M. J., and Nickerson, D. A. (2006). Efficient selection of tagging single-nucleotide polymorphisms in multiple populations. *Hum. Genet.*, **120**: 58–68.
- Hua, J., et al. (2007). SNIper-HD: Improved genotype calling accuracy by an expectation-maximization algorithm for high-density SNP arrays. *Bioinformatics*, **23**: 57–63.
- Huang, B. E., Amos, C. I., and Lin, D. Y. (2007). Detecting haplotype effects in genomewide association studies. *Genet. Epidemiol.*, **31**: 803–812.
- Inada, T., et al. (2008). Pathway-based association analysis of genome-wide screening data suggest that genes associated with the gamma-aminobutyric acid receptor signaling pathway are involved in neuroleptic-induced, treatment-resistant tardive dyskinesia. *Pharmacogenet. Genomics*, **18**: 317–323.
- International HapMap Consortium (2003). The International HapMap Project. *Nature*, **426**: 789–794.
- International HapMap Consortium (2005). A haplotype map of the human genome. *Nature*, **437**: 1299–1320.

- International HapMap Consortium (2007). A second generation human haplotype map of over 3.1 million SNPs. *Nature*, **449**: 851–861.
- Ioannidis, J. P., Patsopoulos, N. A., and Evangelou, E. (2007). Heterogeneity in meta-analyses of genome-wide association investigations. *PLoS ONE*, **2**: e841.
- Ji, W., et al. (2008). Rare independent mutations in renal salt handling genes contribute to blood pressure variation. *Nat. Genet.*, **40**: 592–599.
- Jiang, R., Dong, J., Wang, D., and Sun, F. Z. (2001). Fine-scale mapping using Hardy-Weinberg disequilibrium. *Ann. Hum. Genet.*, **65**: 207–219.
- Kimmel, G., Jordan, M. I., Halperin, E., Shamir, R., and Karp, R. M. (2007). A randomization test for controlling population stratification in whole-genome association studies. *Am. J. Hum. Genet.*, **81**: 895–905.
- Knowler, W. C., Williams, R. C., Pettitt, D. J., and Steinberg, A. G. (1988). Gm3–5,13,14 and type-2 diabetes mellitus: An association in American-Indians with genetic admixture. *Am. J. Hum. Genet.*, **43**: 520–526.
- Komura, D., et al. (2006). Genome-wide detection of human copy number variations using high-density DNA oligonucleotide arrays. *Genome Res.*, **16**: 1575–1584.
- Kwee, L.C., Liu, D., Lin, X., Ghosh, D., and Epstein, M. P. (2008). A powerful and flexible multilocus association test for quantitative traits. *Am. J. Hum. Genet.*, **82**: 386–397.
- Li, Y., and Abecasis, G. R. (2006). Mach 1.0: Rapid haplotype reconstruction and missing genotype inference. *Am. J. Hum. Genet.*, **S79**: 2290.
- Lin, D. Y., and Zeng, D. (2006). Likelihood-based inference on haplotype effects in genetic association studies (with discussion). *J. Am. Stat. Assoc.*, **101**: 89–118.
- Mailman, M. D., et al. (2007). The NCBI dbGaP database of genotypes and phenotypes. *Nat. Genet.*, **39**: 1181–1186.
- Malo, N., Libiger, O., and Schork, N. J. (2008). Accommodating linkage disequilibrium in genetic-association analyses via ridge regression. *Am. J. Hum. Genet.*, **82**: 375–385.
- Manolio, T. A., Brooks, L. D., and Collins, F. S. (2008). A HapMap harvest of insights into the genetics of common disease. *J. Clin. Invest.*, **118**: 1590–1605.
- Marchini, J., Donnelly, P., and Cardon, L. R. (2005). Genome-wide strategies for detecting multiple loci that influence complex diseases. *Nat. Genet.*, **37**: 413–417.
- Marchini, J., Howie, B., Myers, S., McVean, G., and Donnelly, P. (2007). A new multipoint method for genome-wide association studies by imputation of genotypes. *Nat. Genet.*, **39**: 906–913.
- Matsuzaki, H., et al. (2004). Genotyping over 100,000 SNPs on a pair of oligonucleotide arrays. *Nat. Methods*, **1**: 109–111.
- Miyazawa, H., et al. (2007). Homozygosity haplotype allows a genomewide search for the autosomal segments shared among patients. *Am. J. Hum. Genet.*, **80**: 1090–1102.
- Mootha, V. K., et al. (2003). PGC-1 α -responsive genes involved in oxidative phosphorylation are coordinately downregulated in human diabetes. *Nat. Genet.*, **34**: 267–273.
- Ng, P. C., and Henikoff, S. (2003). SIFT: Predicting amino acid changes that affect protein function. *Nucleic Acids Res.*, **31**: 3812–3814.
- Nicolae, D. L. (2006). Testing untyped alleles (TUNA)-applications to genome-wide association studies. *Genet. Epidemiol.*, **30**: 718–727.
- Nicolae, D. L., Wu, X., Miyake, K., and Cox, N. J. (2006). GEL: A novel genotype calling algorithm using empirical likelihood. *Bioinformatics*, **22**: 1942–1947.
- Novembre, J., and Stephens, M. (2008). Interpreting principal component analyses of spatial population genetic variation. *Nat. Genet.*, **40**: 646–649.
- Olshen, A. B., Venkatraman, E. S., Lucito, R., and Wigler, M. (2004). Circular binary segmentation for the analysis of array-based DNA copy number data. *Biostatistics*, **5**: 557–572.
- Plagnol, V., Cooper, J. D., Todd, J. A., and Clayton, D. G. (2007). A method to address differential bias in genotyping in large-scale association studies. *PLoS Genet.*, **3**: e74.
- Price, A. L., Patterson, N. J., Plenge, R. M., Weinblatt, M. E., Shadick, N. A., and Reich, D. (2006). Principal components analysis corrects for stratification in genome-wide association studies. *Nat. Genet.*, **38**: 904–909.
- Pritchard, J. K., Stephens, M., and Donnelly, P. (2000a). Inference of population structure using multilocus genotype data. *Genetics*, **155**: 945–959.

- Pritchard, J. K., Stephens, M., Rosenberg, N. A., and Donnelly, P. (2000b). Association mapping in structured populations. *Am. J. Hum. Genet.*, **67**: 170–181.
- Rabbee, N., and Speed, T. P. (2006). A genotype calling algorithm for affymetrix SNP arrays. *Bioinformatics*, **22**: 7–12.
- Redon, R., et al. (2006). Global variation in copy number in the human genome. *Nature*, **444**: 444–454.
- Roeder, K., Bacanu, S.-A., Wasserman, L., and Devlin, B. (2006). Using linkage genome scans to improve power of association in genome scans. *Am. J. Hum. Genet.*, **78**: 243–252.
- Sabatti, C., Service, S., and Freimer, N. (2003). False discovery rate in linkage and association genome screens for complex disorders. *Genetics*, **164**: 829–833.
- Scheet, P., and Stephens, M. (2006). A fast and flexible statistical model for large-scale population genotype data: Applications to inferring missing genotypes and haplotypic phase. *Am. J. Hum. Genet.*, **78**: 629–644.
- Sebat, J., et al. (2007). Strong association of de novo copy number mutations with autism. *Science*, **316**: 445–449.
- Sham, P., Bader, J. S., Craig, I., O'Donovan, M., and Owen, M. (2002). DNA Pooling: A tool for large-scale association studies. *Nat. Rev. Genet.*, **3**: 862–871.
- Skol, A. D., Scott, L. J., Abecasis, G. R., and Boehnke, M. (2007). Optimal designs for two-stage genome-wide association studies. *Genet. Epidemiol.*, **31**: 766–788.
- Spielman, R. S., McGinnis, R. E., and Ewens, W. J. (1993). Transmission test for linkage disequilibrium: The insulin gene region and insulin-dependent diabetes mellitus (IDDM). *Am. J. Hum. Genet.*, **52**: 506–516.
- Stemers, F. J., Chang, W., Lee, G., Barker, D. L., Shen, R., and Gunderson, K. L. (2006). Whole-genome genotyping with the single-base extension assay. *Nat. Methods*, **3**: 31–33.
- Stephens, M., Smith, N. J., and Donnelly, P. (2001) A new statistical method for haplotype reconstruction from population data. *Am. J. Hum. Genet.*, **68**: 978–989.
- Storey, J. D., Akey, J. M., and Kruglyak, L. (2005). Multiple locus linkage analysis of genomewide expression in yeast. *PLoS Biol.*, **3**: e267.
- Stram, D. O. (2005). Software for tag single nucleotide polymorphism selection. *Hum. Genomics*, **2**: 144–151.
- Stranger, B. E., et al. (2007). Population genomics of human gene expression. *Nat. Genet.*, **39**: 1217–1224.
- Tang, H., Coram, M., Wang, P., Zhu, X., and Risch, N. (2006). Reconstructing genetic ancestry blocks in admixed individuals. *Am. J. Hum. Genet.*, **79**: 1–12.
- Templeton, A. R., Boerwinkle, E., and Sing, C. F. (1987). A cladistic analysis of phenotypic associations with haplotypes inferred from restriction endonuclease mapping. I. Basic theory and an analysis of alcohol dehydrogenase activity in drosophila. *Genetics*, **117**: 343–351.
- Teo, Y. Y., et al. (2007). A genotype calling algorithm for the Illumina BeadArray platform. *Bioinformatics*, **23**: 2741–2746.
- Thornton, T., and McPeck, M. S. (2007). Case-control association testing with related individuals: A more powerful quasi-likelihood score test. *Am. J. Hum. Genet.*, **81**: 321–337.
- Tibshirani, R., and Wang, P. (2008). Spatial smoothing and hot spot detection for CGH data using the fused lasso. *Biostatistics*, **9**: 18–29.
- Tuzun, E., et al. (2005). Fine-scale structural variation of the human genome. *Nat. Genet.*, **37**: 727–732.
- Tzeng, J. Y., Wang, C. H., Kao, J. T., and Hsiao, C. K. (2006). Regression-based association analysis with clustered haplotypes through use of genotypes. *Am. J. Hum. Genet.*, **78**: 231–242.
- Wacholder, S., Chanock, S., Garcia-Closas, M., El Ghormli, L., and Rothman, N. (2004). Assessing the probability that a positive report is false: An approach for molecular epidemiology studies. *J. Natl. Cancer Inst.*, **96**: 434–442.
- Walsh, T., et al. (2008). Rare structural variants disrupt multiple genes in neurodevelopmental pathways in schizophrenia. *Science*, **320**: 539–543.
- Wang, H., Veldink, J., Opooff, R., and Sabatti, C. (2008). Markov models for inferring copy number variations from genotype data on Illumina platforms. UCLA preprint. University of California, Los Angeles.
- Wang, K., Li, M., and Bucan, M. (2008). Pathway-based approaches for analysis of genomewide association studies. *Am. J. Hum. Genet.*, in press.
- Wang, K., et al. (2007). PennCNV: An integrated hidden Markov model designed for high-resolution copy number variation detection in whole-genome SNP genotyping data. *Genome Res.*, **17**: 1665–1674.

- Wang, T., and Elston, R. C. (2007). Improved power by use of a weighted score test for linkage disequilibrium mapping. *Am. J. Hum. Genet.*, **80**: 353–360.
- Wellcome Trust Case Control Consortium (2007). Genome-wide association study of 14,000 cases of seven common diseases and 3,000 shared controls. *Nature*, **447**: 661–678.
- Xiao, Y., Segal, M. R., Yang, Y. H., and Yeh, R. F. (2007). A multi-array multi-SNP genotyping algorithm for Affymetrix SNP microarrays. *Bioinformatics*, **23**: 1459–1467.
- Xing, J., Witherspoon, D. J., Watkins, W. S., Zhang, Y., Tolpinrud, W., and Jorde, L. B. (2008). HapMap tagSNP transferability in multiple populations: General guidelines. *Genomics*, **92**: 41–51.
- Zeggini, E., et al. (2008). Meta-analysis of genome-wide association data and large-scale replication identifies additional susceptibility loci for type 2 diabetes. *Nat. Genet.*, **40**: 638–45.
- Zhu, X., Zhang, S., Zhao, H., and Cooper, R. S. (2002). Association mapping, using a mixture model for complex traits. *Genet. Epidemiol.*, **23**: 181–196.
- Zollner, S., and Pritchard, J. K. (2007). Overcoming the winner's curse: Estimating penetrance parameters from case–control data. *Am. J. Hum. Genet.*, **80**: 605–615.

DEFINITIONS

Allele: Different forms of DNA sequences that vary among individuals.

Admixture mapping: An association approach based on the study of admixed samples where the objective is to identify regions that tend to have genetic material from one ancestral population among the diseased individuals.

Bonferroni correction: A commonly adopted approach to controlling false-positive results due to multiple comparisons. Under this approach, the statistical significance level is set to be the ratio between the global nominal level and the number of statistical tests conducted.

Coalescent: A retrospective model that traces the genetic materials from all the individuals in a sample to a single ancestor.

Copy number variation (CNV): Genetic polymorphism that is defined by the differing number of a specific segment of DNA carried among different people.

False discovery rate: Proportion of false-positive results among all the reported positive results. This is usually estimated through permutations in practice.

Family-based association study: A type of association study that is robust to population stratification. It is based on examining departure from nonrandom transmission of alleles from heterozygous parents to the affected offspring.

Familywise error rate: The error rate for one or more false positives among all the tests conducted. Controlling a familywise error rate leads to more stringent control than the false discovery rate control at the same nominal level.

Genomic control: A statistical approach to reducing the effect of sample heterogeneity through the use of an adjustment factor in statistical tests.

Haplotype: The combination of alleles across different markers on the same chromosome. It is used to tag untyped markers as well as to capture interactions among alleles on the same chromosome.

Hardy–Weinberg equilibrium (HWE): Under HWE, the two chromosomes from the same individual can be thought of as two random samples from the general population chromosome pool. HWE is commonly used as a tool to detect genotyping errors. The degree of departure from HWE is often used to quantify population heterogeneity.

Imputation: Statistical methods to infer the genotypes of untyped genetic markers from tag SNPs based on the dependency structure among the markers.

Inbreeding: Mating between related individuals. Inbreeding can lead to segments of homozygotes in an individual.

Linkage disequilibrium: Nonindependency among physically closely spaced genetic markers. The presence of linkage disequilibrium is the basis of selecting tag SNPs and using tag SNPs to conduct genome-wide association studies.

Permutation: A statistical method to empirically assess statistical significance of the observed association signals.

Principal component analysis: A statistical technique to summarize the observed high-dimensional data in a lower dimensional space through the linear combination of observed variables. It is commonly used to summarize genetic background information across study subjects.

Sample heterogeneity (population stratification): Study subjects come from diverse genetic backgrounds. The presence of sample heterogeneity can introduce spurious associations between disease and genetic markers.

Single-nucleotide polymorphism (SNP): Genetic polymorphisms that differ at a single nucleotide among individuals.

Structure analysis: The use of genetic markers to infer the underlying genetic structure in the sampled individuals.

Tag SNP: SNPs genotyped in a study that can be used to infer the state of the untyped SNPs and other variants.

***R AND BIOCONDUCTOR
PACKAGES IN BIOINFORMATICS:
TOWARD SYSTEMS BIOLOGY***

*Nolwenn LeMeur, Michael Lawrence, Merav Bar,
Muneesh Tewari, and Robert Gentleman*

*Fred Hutchinson Cancer Research Center, Program in Computational Biology,
Division of Public Health Sciences, Seattle, Washington*

13.1 INTRODUCTION

Recent improvement in molecular biology and associated technologies have generated a flow of valuable experimental data. To explore and analyze those data require efficient flexible data structures and statistical tools designed with the particular challenges of high-throughput data in mind. Since 2001, the Bioconductor community has worked on developing such methods and the associated infrastructure for a wide variety of technologies (from DNA sequences to flow cytometry data). This has led to interesting biological results in terms of individual entities, such as genes or proteins. However, those approaches typically consider only a single aspect of a biological system, such as gene expression, and hence are insufficient to explain the complexity of the system in its entirety. Integrative approaches will be needed to extend our knowledge and to acquire a holistic understanding of a particular biological system.

In this chapter we use a series of experiments carried out in the Tewari lab at the Fred Hutchinson Cancer Research Center (FHCRC) as a motivating example (Bar et al., 2008). We will only be able to consider a few of the simpler aspects of the data-analytic process, but these experiments show how complex experiments in molecular biology have become. Briefly, one of the main goals of the project was to understand how microRNAs (He and Hannon, 2004) regulate the differentiation of stem cells from the pluripotent state. Deep sequencing was used to determine differentially expressed microRNAs, and this was then followed up by a number of other experiments, in particular a microarray gene expression experiment to determine whether functional effects of the microRNAs could be detected. MicroRNAs are believed to regulate messenger RNA (mRNA) expression. Hence differences in microRNA abundance between two conditions may also be reflected in differences

in mRNA abundance. We will focus our attention mainly on this microarray portion of the overall experiment.

In this chapter we begin with a brief description and discussion of data structures and provide an overview of the tools that are available for preprocessing and quality assessment. We then describe a number of different graphical user interfaces (GUIs) that can be used to access these tools. We next discuss some of the many other types of data that might arise and how their analysis could be integrated with gene expression data. In molecular biology experiments there is a great deal of additional information that is often pertinent to the analysis, ranging from the scientific literature to sequence information and known functional characteristics of the genes involved, to name just a few. We discuss some of the many ways this information can be retrieved, manipulated, and visualized. Then, based on our analysis we consider a specific pathway that is shown to be differentially expressed and apply a more classical systems biology approach to it, showing how to visualize it and quantitatively simulate it. The tools come in the form of different R packages that can be downloaded and used on a local computer. Each package provides software, or data, to carry out a specific set of tasks, and the integration of the packages is an effective strategy for the rapid development of methods for analyzing biological data in the system context.

This chapter was written using the Sweave (Leisch, 2002) system, and all the code used to produce the outputs, graphics, and so on, can be retrieved from our online supplements, located at <http://www.bioconductor.org/Publications/2008/Wiley>. In addition, a few other plots and details are provided at that site for interested readers.

13.2 BRIEF OVERVIEW OF THE BIOCONDUCTOR PROJECT

The Bioconductor project essentially consists of several hundred software packages that are aimed at providing solutions to a large number of bioinformatic and computational biology problems. In addition there is an extensive set of packages that provide annotation information for different microarray experiments as well as whole organisms. A number of complete real datasets that can be used for method development and testing are also available. To aid users in navigating the large collection of packages a simple keyword hierarchy or ontology has been developed (called *biocViews*) and this can be accessed using the Packages link on the left side of the main Bioconductor page. Links are also found on every package-level web page.

Bioconductor is widely used for the analysis of gene expression data. The *arrayQualityMetrics* package provides a comprehensive overview of quality metrics for any type of array (both one and two color). Normalization is a process that is used to remove nonbiological differences between arrays prior to determining whether there are differentially expressed genes. The *affy*, *vsn*, and *limma* are the most widely used set of packages for carrying out these steps. Differential expression can be assessed by a number of packages, with *limma* being the tool of choice, largely due to its use of a Bayesian shrinkage method, which is important when there are relatively few arrays. Testing predefined sets of genes using either a hypergeometric

approach or gene set enrichment analysis is also popular and relevant packages include GOSTats, GSEABase, GSEAlm, and Category.

Many analyses rely heavily on genomic annotation. This might include known functions or properties of genes, their sequence, pathways they participate in, and so on. Bioconductor provides a comprehensive collection of platform-independent SQLite-based annotation packages that users can download and use. Many of the software packages make use of standard application programming interfaces (APIs) and have well-integrated tools for analyzing microarray or other data in conjunction with these annotation data. The *annaffy* package provides a fairly comprehensive set of tools for creating linked HTML output, based on the results of an analysis, that are suitable for sharing. There are tools in the *annotate* package that provide infrastructure for those that want to create their own linked output. Other tools in the *annotate* package can be used to access papers and abstracts in PubMed, providing a rudimentary system for literature mining.

Tools for representing, computing on, and rendering graphs and networks are provided by the *graph*, *RBGL*, and *Rgraphviz* packages. These packages can be used for representing pathways, hierarchical ontologies (such as GO), as well as models in systems biology, as we see in Section 13.5.

High-throughput genotyping via single-nucleotide polymorphism (SNP) chips is addressed at several levels. First, capture of raw SNP array data from Affymetrix arrays is handled by the *oligo* package, and data from Illumina arrays are handled by *beadarraySNP*. Second, organization and computation on megaSNP series are facilitated by the *snpMatrix* package. Third, coordinated structure and analysis of expression and genotype data are handled by the *GGtools* package, which supports full phase II HapMap genotyping (4 million SNPs) on multiple HapMap populations.

High-throughput screening for the activity of molecular compounds or RNA interference (RNAi) with cell-based assays is supported by the *cellHTS2* package. For higher content assays that employ (flow) cytometry, there is a suite of packages for data management, visualization, automated gating, and inference; relevant packages include *flowCore*, and *flowViz*. For high content screening using automated microscopy, the *EBImage* provides basic functionality for image input–output (I/O), display, and representation in R, and a growing set of image processing and analysis functions pertinent to this application domain.

Recent developments include robust, fast tools for string matching as well as alignment of biological sequences. The *ShortRead* package provides an interface to many different high-throughput short-sequence data formats while *Biostrings* has tools for matching sequences to target genomes and for performing classical alignments. Whole genomes are represented in a special format and the general infrastructure is contained in the *BSgenome* package, with many model organism genomes available for download.

13.3 EXPERIMENTAL DATA

Systems biology proposes high-level models that attempt to explain a complex series of events that occur at the biochemical level. The derivation and validation of such a

model depend on the ability to experimentally query the cell for actual biochemical states. Model participants often include many different molecular species, such as genes, proteins, and metabolites. For example, in a cell-signaling cascade, protein enzymes might conduct the signal through protein phosphorylation events, and this might result in the regulation of gene transcription. The change in transcription might then affect the metabolism of the organism. Thus, the analysis of multiple types of experimental data is central to systems biology.

In this section, we introduce the general infrastructure in Bioconductor for representing and accessing experimental datasets and specifically describe support for manipulating and analyzing gene expression, proteomics, and metabolomics datasets. It is possible to analyze many other types of data using Bioconductor, such as flow cytometry and biological sequence data, but these will not be covered here.

13.3.1 General Infrastructure

Since its beginning the Bioconductor project has emphasized the importance of self-describing data structures that contain experimental data as well as other data associated with the experiment, such as feature-level annotations, the experimental protocol, and a description of the experimental design. Our experience in dealing with high-throughput biological data suggested that there were a number of important commonalities that could most easily be captured by using an object-oriented programming approach by defining a small number of classes that are intended to provide building blocks for more specific uses. These classes are primarily intended to store data that have already been preprocessed and are ready for analysis. The idea here is that by having common data structures, for quite different input data, we can create analysis packages that are somewhat modular.

The basic class definitions and other infrastructure are provided in the Biobase package. The base class is the `eSet` and it has places to store assay data, phenotypic information, and data about the features that were measured and about the experiment that was performed to collect these data. This basic class can then be extended in many different ways, specializing in some of the inputs, and in some cases adding new slots that are relevant to a specific type of experiment. The `eSet` class has been extended to support expression data, SNP data, and genomic annotation. For expression data the `ExpressionSet` class has been defined, and it too is quite general. The class can be used for any sort of expression (and is in no way restricted to microarray experiments for mRNA expression, although that is where it is used most). Similar data structures have been used for representing data from experiments in protein mass spectrometry and flow cytometry.

13.3.2 Accessing Experimental Data

Experimental datasets are useful for prototyping, testing, and demonstrating statistical methods. There are a number of sources of sample datasets accessible from Bioconductor, including the Bioconductor repository and the GEO database. In the Bioconductor repository, there is a collection of R packages that contain experimental data from different high-throughput experiments and the current set can be found by

navigating to the ExperimentData link under BiocViews. The GEOquery (Davis and Meltzer, 2007) package provides an interface for accessing datasets in the GEO repository (<http://www.ncbi.nlm.nih.gov/geo>), which now contains many thousands of microarray experiments. GEOquery downloads the data and transforms it into instances of the ExpressionSet class, which can then be analyzed using appropriate Bioconductor packages.

13.3.3 Example Dataset

Throughout this chapter, we will make use of gene expression measurements from human embryonic stem cells (hESCs) undergoing differentiation (i.e., undifferentiated and differentiated cells). This experiment was performed on HG-U133plus2 Affymetrix GeneChip arrays and was a small part of a larger examination of the role of microRNAs, which are small RNAs, typically about 22 nucleotides long, in the differentiation of embryonic stem cells. MicroRNAs are known to inhibit translation of so-called target mRNAs by binding to the 3'UTR of the mRNA and designating the mRNA for either sequestration or degradation. The complete set of genes that any microRNA targets is not known, but predictions are available from miRBase (<http://microrna.sanger.ac.uk/>) and we will make use of these later in this chapter. It is believed that if a microRNA is expressed in one condition and not in a second, then there will be corresponding differences in the expression levels of its target genes (often down-regulated when the microRNA is expressed). Differentially expressed microRNAs were assessed by deep sequencing and the 10 most overexpressed microRNAs in the undifferentiated hESCs were hsa-miR-302b, hsa-miR-302c, hsa-miR-302d, hsa-miR-92b, hsa-miR-20b, hsa-miR-519d, hsa-miR-302a, hsa-miR-324-3p, hsa-miR-187, and hsa-miR-18b.

The preprocessed data, in the form of an ExpressionSet, are contained in the humanStemCell experimental data package available from the Bioconductor repository. The Affymetrix raw data files (CEL files) are also provided with the package, so interested readers may replicate the preprocessing if desired. The quality of the resulting arrays was assessed using the arrayQualityMetrics package, which generates statistical and graphical summary as an HTML quality report. Readers can access a version of that report at <http://www.bioconductor.org/Publications/2008/Wiley>. We would like to highlight here that quality assessment (QA) is an essential step for every analysis. For detailed examples and discussion of the preprocessing and QA methods, see Gentleman et al. (2005), Hahne et al. (2008), or the package vignettes. We reproduce one figure from the QA assessment in Figure 13.1.

The following code loads the ExpressionSet instance, named fhesc, into R:

```
> library("humanStemCell")
> data(fhesc)
> fhesc

ExpressionSet (storageMode: lockedEnvironment)
assayData: 15524 features, 6 samples
element names: exprs
```

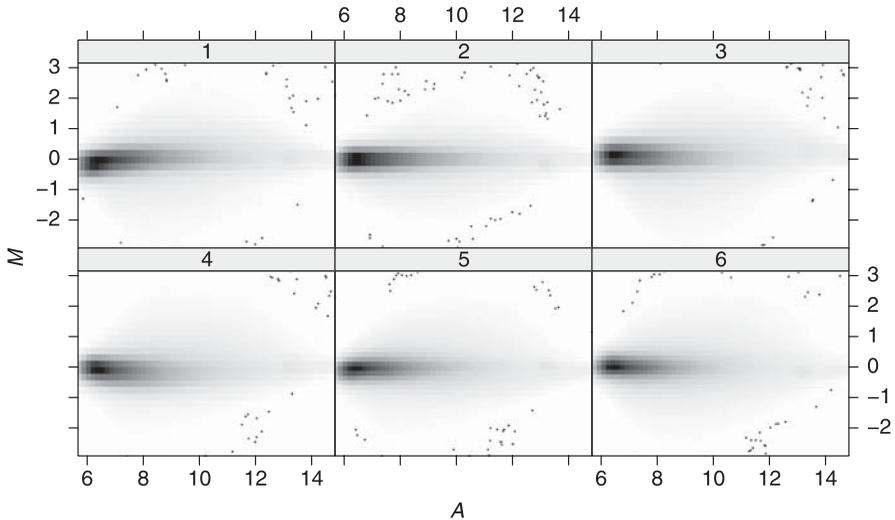


Figure 13.1 Mean-difference smooth scatterplot for each array (log base 2). For each array, the mean and difference are computed using the intensity of the array studied and the intensity of a pseudoarray, which is constructed to have the median value, across all arrays, for each probe. We expect to see values are concentrated along the $M = 0$ line, since the majority of genes are not differentially expressed. Patterns, such as trends, are indicative of potential problems.

```
phenoData
  sampleNames: 1_Hu-hESC-Undiff-I, 2_Hu-hESC-Undiff-II, ...,
               6_Hu-hESC-Diff-III (6 total)
  varLabels and varMetadata description:
    SampleID: Sample ID
    Diff: Diff: Differentiated or Not
featureData
  featureNames: 229819_at, 206797_at, ..., 227076_at (15524 total)
  fvarLabels and fvarMetadata description: none
experimentData: use 'experimentData(object)'
Annotation: hgu133plus2
```

As indicated in the output above, the data consist of six samples and two different biological states: differentiated and undifferentiated. There are three biological replicates for each state.

13.3.3.1 Nonspecific Filtering We next employ a nonspecific filtering approach to remove probesets that show little variation across samples and to ensure that each gene is represented by one probeset. Readers might take a different approach, but in general some form of nonspecific filtering is beneficial:

```
> library("genefilter")
> filtFhesc <- nsFilter(fhesc)[[1]]
```

13.3.3.2 Linear Modeling Next, we detect differentially expressed genes using the *limma* package (Smyth, 2004), which fits a linear model to each biological entity (in this case genes) and adjusts the variance in the calculation of the *t*-statistic using an empirical Bayes method. The code is given below, we did some checking to find that approximately 6700 genes had adjusted *p*-values below 0.01, indicating that there are significant changes. This of course can also reflect high reproducibility, so one might want to also require that the estimated ratio of change between the two conditions (fold change, or FC) be large. In order to reduce the number of genes to something manageable for this discussion, we arbitrarily select a cutoff of 2 (the FC is reported on a log 2 scale so we set the cutoff to 1):

```
> library("limma")
> design <- model.matrix(~filtFhesc$Diff)
> hesclim <- lmFit(filtFhesc, design)
> hesceb <- eBayes(hesclim)
> tab <- topTable(hesceb, coef = 2,
+               adjust.method = "BH",
+               n = 7676)
> tab2 <- tab[(tab$logFC > 1) & (tab$adj.P.Val < 0.01),]
```

This leaves us with 2049 genes that are up-regulated in differentiated cells compare to undifferentiated (Fig. 13.2), and we select them for further analysis.

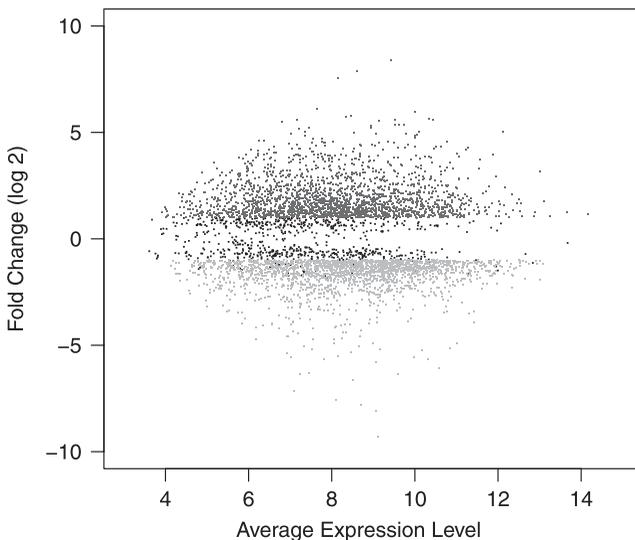


Figure 13.2 Mean-difference scatterplot of gene expression between differentiated and undifferentiated cells (log base 2). Each dot represents a gene: red dots highlight genes up-regulated in differentiated cells; green dots represent genes down-regulated in differentiated cells; black dots correspond to genes that are not differentially expressed. (See color insert.)

13.3.3.3 Graphical User Interfaces The code listed above for running limma is usually entered through a command line interface (CLI) to R. Biologists are often unfamiliar with programming computers and interacting with CLIs. A GUI, though generally less flexible than a CLI, is often employed to facilitate a specific task, such as fitting a model with limma. The limmaGUI and affyImGUI packages (Smyth, 2004) provide GUIs to the limma package and other functionality in R/Bioconductor for quality checking, preprocessing, and analyzing gene expression data. The oneChannelGUI package (Sanges et al., 2007) extends affyImGUI to handle more types of data (besides only Affymetrix Gene Chip data) and adds many analysis features. The exploRase package (Cook et al., 2008) is a GUI for the exploratory analysis of any dataset stored as an ExpressionSet. One component of exploRase provides a high-level interface to limma, where the contrasts are automatically determined and fitted for each experimental design factor selected by the user. The code below loads exploRase for the stem cell dataset:

```
> explorase(fhesc)
```

See Figure 13.3 for a screenshot of the resulting GUI.

13.3.4 Other Microarray Data Sources

Microarrays are also used quite extensively to assess single SNPs, copy number variation (via array-comparative genomic hybridization, aCGH), microRNA abundance, to name but a few of the very many uses. There is a great deal of interest in combining, or integrating, data from these different sources. There are specialized packages for some of these processes, such as the aCGH or DNACopy packages for aCGH analysis, as well as more general tools such as vsn and oligo that can be adapted to different applications.

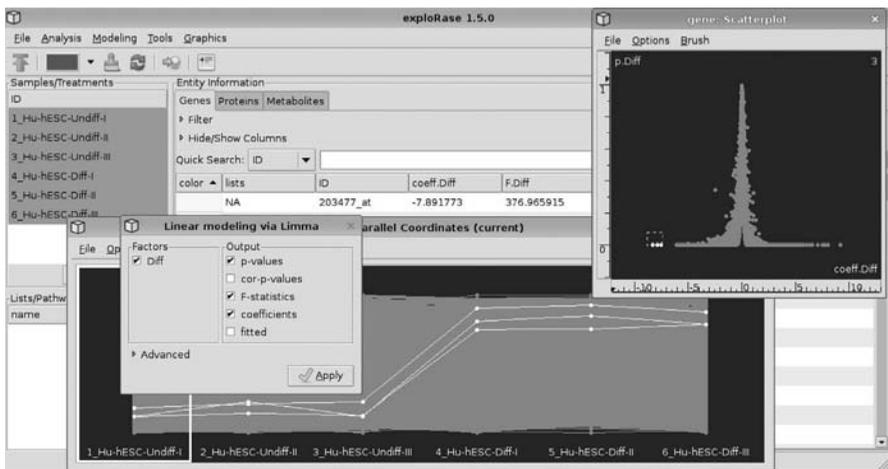


Figure 13.3 Screenshot of the exploRase GUI during limma analysis.

One interesting approach that has received some attention is that of genetical genomics, where gene expression data are treated as a quantitative trait and changes in gene expression are related to SNP variation (Cheung et al., 2005). The GGtools (Carey et al., 2007) provides a set of software tools for carrying out these different analyses given a set of SNP data and a corresponding set of gene expression data.

13.3.5 Proteomics Data

The transcripts measured in gene expression experiments, like the one described above, are translated into proteins, given the correct regulatory conditions. Situated at the end of the molecular biology paradigm, proteins are an important link between the genotype and phenotype of an organism. Proteomics datasets describe the levels of proteins in a biological sample.

Mass spectrometry is a popular technology for detecting the proteins present in a sample. Each protein yields a characteristic signal in the mass spectrometer output, known as a mass spectrum. As with most experimental technologies, the signal in the raw mass spectrum is shrouded in noise. Statistical tools are necessary to separate the signal from the noise during preprocessing.

Bioconductor contains two packages for preprocessing mass spectra. PROcess implements baseline subtraction and peak detection algorithms that are designed for surface-enhanced laser desorption/ionization time-of-flight (SELDI-TOF) data but likely applicable to mass spectra from other sources. MassSpecWavelet (Du et al., 2006) detects peaks in the continuous wavelet transform (CWT) of a mass spectrum. The CWT operation smooths the spectrum and removes the baseline in a single step. This improves the confidence of the peak detection by improving the signal-to-noise ratio.

Affinity purification coupled with mass spectrometry (AP-MS) and yeast two-hybrid (Y2H) are high-throughput technologies that detect protein–protein interactions (PPIs). Analysis of this type of data may improve our understanding of cellular systems and, in particular, the complex series of protein interactions that drive a cell. However, taking full advantage of such datasets is challenging due to the large error rates.

The Bioconductor project offers several packages to process and analyze PPI data. For example, the Rintact package provides access to PPI data in the IntAct repository (<http://www.ebi.ac.uk/intact>). The quality of the data can then be assessed using methods from the ppiStats package (Chiang et al., 2007). Next, we can use the apComplex package that proposes methods to estimate protein complex membership from AP-MS experiments (Scholtens et al., 2005) and, finally, packages like SLGI to integrate proteomic and genomic interactions data.

13.3.6 Metabolomics Data

Metabolomics is the study of the host of metabolites in a sample. Metabolites are small organic molecules, such as sugars, amino acids, and lipids, that are interconverted by enzymes (proteins) in a biochemical system. Metabolites have a wide variety of

purposes, such as energy storage and signaling, and they are the building blocks of proteins and DNA.

A metabolomics experiment typically aims to determine the levels of one or more metabolites across a set of samples. Metabolic profiling targets specific metabolites that are assumed to have a signature that distinguishes one biological state, such as a disease, from another. Untargeted metabolomics, by contrast, attempts to determine the levels and identity of every metabolite in a sample. This section will focus on support in Bioconductor for the analysis of untargeted metabolomics experiments.

A common experimental approach to untargeted metabolomics is chromatography coupled with mass spectrometry. Specific technologies include gas chromatography–mass spectrometry (GC-MS) and liquid chromatography mass spectrometry (LC-MS). The chromatography phase separates the complex biological sample along a time dimension, and the output of the chromatography is scanned by a mass spectrometer. From each scan, it may be possible to identify a metabolite and calculate its relative quantity.

The Bioconductor package `xcms` (Smith et al., 2006) provides data structures and methods for preprocessing raw GC-MS and LC-MS data to derive a set of metabolites and their relative quantities. Data may be input as NetCDF or mzXML files (network common data form for sharing of scientific data and common file format for MS data, respectively). The preprocessing steps include finding peaks in the chromatographic time dimension, aligning samples in time, grouping signatures that represent the same metabolite across samples, and quantifying the metabolites. The result may be represented as an `ExpressionSet` object and passed to other R and Bioconductor tools, such as the `limma` package described in Section 13.3.3. For identifying metabolites, the `Rdisop` package integrates with `xcms` and supports generating putative molecular formulas from mass spectral signatures.

13.4 ANNOTATION

Within a system biology perspective, relevant knowledge about the biological system under investigation is a prerequisite for an effective analysis and interpretation of experimental results. Knowledge acquisition corresponds to both annotation of the biological samples and the molecular species studied. However, annotation can be difficult to achieve as knowledge is often scattered and encoded in various formats. Bioconductor provides tools for creating or accessing annotation resources. For instance, the molecular species (e.g., gene, protein) under study can be retrieved either from a Bioconductor metadata package or an online database.

13.4.1 Sample Annotation

Sample annotation is usually provided by the biologist or clinician in a spreadsheet-like format (e.g., tabular-delimited file). The variables on the samples may be qualitative or quantitative. For instance, if the experiment deals with patients, each sample may be characterized by the age and the sex of the patient plus a variety of disease characteristics. For designed experiments, the samples are typically annotated

with the design factors, such as the times in a time-course experiment. In an `ExpressionSet` container, this information is stored in the `phenoData` object and is accessible in a number of ways. For example, we can retrieve the names of the samples using the simple method call `sampleNames`:

```
> sampleNames(fhesc)
[1] "1_Hu-hESC-Undiff-I" "2_Hu-hESC-Undiff-II" "3_Hu-hESC-Undiff-III"
[4] "4_Hu-hESC-Diff-I"  "5_Hu-hESC-Diff-II"  "6_Hu-hESC-Diff-III"
```

Sample annotations are also useful for subsetting a dataset in order to remove samples that are likely to be noninformative for our biological question.

13.4.2 Feature Annotation

It is difficult to analyze and interpret experimental data in a vacuum. Incorporating metadata on the experimental features, for example, the microarray probesets in our stem cell dataset, is useful as input to statistical algorithms, as well as for assigning biological meaning to results. For instance, chromosome locations are important for the analysis of sequence-oriented datasets, such as CGH and SNP data. Likewise, functional information such as the Gene Ontology (GO) annotation associates experimental data with cellular functions (Gene Ontology Consortium, 2000). The Bioconductor project provides metadata packages and remote-access tools for obtaining annotations on the measured features in an experiment. We describe each of these in turn.

13.4.2.1 Metadata Packages The annotation section of the Bioconductor repository provides an extensive and increasingly growing catalog of R packages that encode annotation resources. An annotation package might, for example, assemble knowledge data about a specific microarray platform or the genome of an organism. Each package strives to fill the gap between sequence information and function by integrating various annotation databases through identifier mappings. In our example dataset, we make use of the well-known Affymetrix platform, the HG-U133plus2 GeneChip. The annotation package maps each manufacturer identifier to, for example, an Entrez Gene identifier or chromosome location. To make this more concrete, we load the package and list its contents:

```
> library("hgu133plus2.db")
> ls("package:hgu133plus2.db")
[1] "hgu133plus2"           "hgu133plus2ACCNUM"
[3] "hgu133plus2ALIAS2PROBE" "hgu133plus2CHR"
[5] "hgu133plus2CHRLNGTHS"  "hgu133plus2CHRLOC"
[7] "hgu133plus2CHRLOCEND"  "hgu133plus2ENSEMBL"
[9] "hgu133plus2ENSEMBL2PROBE" "hgu133plus2ENTREZID"
[11] "hgu133plus2ENZYME"     "hgu133plus2ENZYME2PROBE"
[13] "hgu133plus2GENENAME"   "hgu133plus2GO"
[15] "hgu133plus2GO2ALLPROBES" "hgu133plus2GO2PROBE"
[17] "hgu133plus2MAP"       "hgu133plus2MAPCOUNTS"
```

```

[19] "hgu133plus2OMIM"           "hgu133plus2ORGANISM"
[21] "hgu133plus2PATH"          "hgu133plus2PATH2PROBE"
[23] "hgu133plus2PFAM"          "hgu133plus2PMID"
[25] "hgu133plus2PMID2PROBE"    "hgu133plus2PROSITE"
[27] "hgu133plus2REFSEQ"        "hgu133plus2SYMBOL"
[29] "hgu133plus2UNIGENE"       "hgu133plus2UNIPROT"
[31] "hgu133plus2_dbInfo"       "hgu133plus2_dbconn"
[33] "hgu133plus2_dbfile"       "hgu133plus2_dbschema"

```

Each of these, except the last four that provide programmatic access to the underlying database, is a map between Affymetrix probe IDs and some genomic entity of interest. They all have manual pages that describe in detail where and when the relevant data were obtained as well as explicit details on how the mappings were carried out. Users with specialized needs can always create their own annotation packages using the AnnotationDbi package.

While the large majority of annotation packages are oriented toward specific microarray chips from different manufacturers, Bioconductor also provides whole genome annotations for most model organisms. The names of these packages take the form `org.Hs.eg.db`. The prefix `org` indicates that the package contains whole organism annotation, the `Hs` indicates the package is for *Homo sapiens*, the `eg` component indicates that Entrez Gene identifiers are the primary keys, and the `db` suffix indicates that this package uses SQLite (<http://www.sqlite.org/>) databases.

13.4.2.2 Accessing Remote Data Sources Access to data and other resources through the Internet is another important characteristic of a suitable software tool for computational biology. The Bioconductor project provides a number of different tools that can be used to obtain data interactively. Perhaps the most flexible of these is the `biomaRt` package (Durinck et al., 2005), which provides an interface to BioMart databases (<http://www.biomart.org>), which are used by a number of model organism databases as well as other sources of genomic data (e.g., Ensembl).

One can also query PubMed through tools available in the `annotate` package to obtain information on papers, such as the abstract, authors, and so on, for which a PubMed ID is known. A simple use case would be to obtain all differentially expressed genes in a microarray experiment, map the platform identifiers of the genes to the corresponding PubMed IDs through the Bioconductor annotation package for the platform, and search the matching abstracts for specific terms of interest.

In the code segments below we demonstrate the incorporation of PubMed abstracts into an experimental data analysis. We first find all PubMed IDs associated with a given Affymetrix probe ID and then reverse that so that the variable `byGene` is a list of PubMed IDs, each containing the genes that have been discussed in the paper, as well as differentially expressed in our experiment:

```

> affyIDs <- tab2$ID
> PMIDs <- mget(affyIDs, hgu133plus2PMID)
> byGene <- reverseSplit(PMIDs)
> length(byGene)

[1] 51924

```

We see that there are 51,924 distinct PubMed IDs that are associated with our differentially expressed genes. We can then look to see how many genes are cited by each paper (we print only the 10 with the largest counts):

```
> lens <- sapply(byGene, length)
> head(sort(lens, decreasing = TRUE), 10)

12477932 15489334 14702039 16344560 9373149 8125298 8889548 17081983
      1923      1296      1040      398      208      202      195      184
16189514 18029348
      159      155
```

Notice that some papers reference very many genes, but typically these are not very informative. For example, PubMed ID 12477932 is a paper describing more than 15,000 full-length mouse and human complementary DNAs (cDNAs). In this case, since we want just one abstract, we will extract it manually:

```
> library("annotate")
> abs1 <- pubmed("12477932")
> buildPubMedAbst(xmlRoot(abs1)[[1]])
An object of class 'pubMedAbst':
Title: Generation and initial analysis of more than 15,000 full-length
      human and mouse cDNA sequences.
PMID: 12477932
Authors: RL Strausberg, EA Feingold, LH Grouse, JG Derge, RD Klausner,
      FS Collins, L Wagner, CM Shenmen, GD Schuler, SF Altschul, B
      Zeeberg, KH Buetow, CF Schaefer, NK Bhat, RF Hopkins, H Jordan, T
      Moore, SI Max, J Wang, F Hsieh, L Diatchenko, K Marusina, AA
      Farmer, GM Rubin, L Hong, M Stapleton, MB Soares, MF Bonaldo, TL
      Casavant, TE Scheetz, MJ Brownstein, TB Usdin, S Toshiyuki, P
      Carninci, C Prange, SS Raha, NA Loquellano, GJ Peters, RD
      Abramson, SJ Mullahy, SA Bosak, PJ McEwan, KJ McKernan, JA Malek,
      PH Gunaratne, S Richards, KC Worley, S Hale, AM Garcia, LJ Gay, SW
      Hulyk, DK Villalón, DM Muzny, EJ Sodergren, X Lu, RA Gibbs, J
      Fahey, E Helton, M Kettelman, A Madan, S Rodrigues, A Sanchez, M
      Whiting, A Madan, AC Young, Y Shevchenko, GG Bouffard, RW
      Blakesley, JW Touchman, ED Green, MC Dickson, AC Rodriguez, J
      Grimwood, J Schmutz, RM Myers, YS Butterfield, MI Krzywinski, U
      Skalska, DE Smailus, A Schnerch, JE Schein, SJ Jones, MA Marra, M
      LastName
Journal: Proc Natl Acad Sci U S A
Date: Dec 2002
```

Alternatively, to display the output in your web browser, rather than import it into R, we can pass the `disp` argument for the `pubmed` function as “browser,” as is shown below:

```
> pubmed("12477932", disp="browser")
```

One can also use biological pathways to annotate their data. For instance, we might want to determine whether the genes of interest are concentrated in a particular pathway. Or we might want to investigate those that are involved in more than one pathway. To help answer those questions, Bioconductor provides a preprocessed version of the widely used Kyoto Encyclopedia of Genes and Genomes (KEGG) (Kanehisa and Goto, 2000) in the `KEGG.db` package, which maps, for example, pathway information to Entrez Gene IDs and open-reading frame IDs. Bioconductor also provides access to the on-line version of KEGG through the `KEGGSOAP` package, which communicates via the SOAP protocol. With `KEGGSOAP`, users can query KEGG by various keys: species, pathways, genes, enzymes, and so on.

We first retrieve all KEGG pathway IDs associated with the set of Affymetrix probe IDs we selected, reverse the list so that the variable `pathbyGene` is a list of KEGG IDs, each containing the set of genes in the pathway, and identify the pathway that contains the largest number of selected genes:

```
> keggIDs <- mget(affyIDs, hgu133plus2PATH)
> pathbyGene <- reverseSplit(keggIDs)
> pwlens <- sapply(pathbyGene, length)
> which.max(pwlens)
04510
113
```

Using the `KEGGSOAP` package, we verify that the pathway 04510 exists in humans, and, since it does, we then retrieve its name and find, for example, all of the associated enzymes. First, we list all the pathways known in humans (of species code `hsa`) and next find the name of our pathway using the standardized pathway ID of the form `path:SpeciesKEGGid`, that is, `path:hsa04510`:

```
> library("KEGGSOAP")
> humanPathway <- list.pathways("hsa")
> humanPathway["path:hsa04510"]

path:hsa04510
"Focal adhesion - Homo sapiens (human)"
```

Next, we retrieve the enzymes involved in that pathway:

```
> enz <- get.enzymes.by.pathway("path:hsa04510")
> enz

[1] "ec:2.7.1.153" "ec:2.7.1.68" "ec:2.7.10.1" "ec:2.7.10.2" "ec:2.7.11.1"
[6] "ec:2.7.11.13" "ec:2.7.11.18" "ec:2.7.11.24" "ec:2.7.11.26" "ec:2.7.12.2"
[11] "ec:3.1.3.16" "ec:3.1.3.67" "ec:3.4.21.-" "ec:3.4.22.53"
```

We can display the KEGG pathway diagram in a browser with the elements corresponding to the enzymes highlighted in red. Such a plot is provided in our on-line supplements (<http://www.bioconductor.org/Publications/2008/Wiley/>):

```
> url <- mark.pathway.by.objects("path:hsa04510", enz)
> browseURL(url)
```

13.4.3 Genomic Annotations

Many of the feature annotations provided by Bioconductor metadata packages are genomic, in that they are tied to a specific location in the genome. A vast amount of genomic knowledge has been accumulated in databases of genomic annotations, and new annotations may be derived from many types of analysis results, such as that from DNA copy number experiments. Bioconductor provides the infrastructure for representing and visualizing genomic annotations from R, in the interest of supporting experimental data analysis. We demonstrate these features on the stem cell dataset.

13.4.3.1 Data Structures Precisely integrating genomic annotations into an experimental data analysis requires representing the annotations at the sequence level; direct annotations of experimental features no longer suffice. The Bioconductor package `rtracklayer` provides support for importing genomic annotations, also known as tracks, into R. Tracks are loaded as instances of the `RangedData` class, defined by the `IRanges` package for the storage of data on ranged features. The genomic features are defined in terms of their chromosome, start position, end position and strand (+ or -). With `rtracklayer`, a `RangedData` may be imported from data formatted as General Feature Format (GFF) (Durbin et al., 2000), Browser Extended Display (BED) (UCS, a) or Wiggle (WIG) (UCS, b).

13.4.3.2 Visualization Tracks are often visualized as parallel rectangles stacked on top of each other and horizontally aligned against a reference sequence (Fig. 13.4).

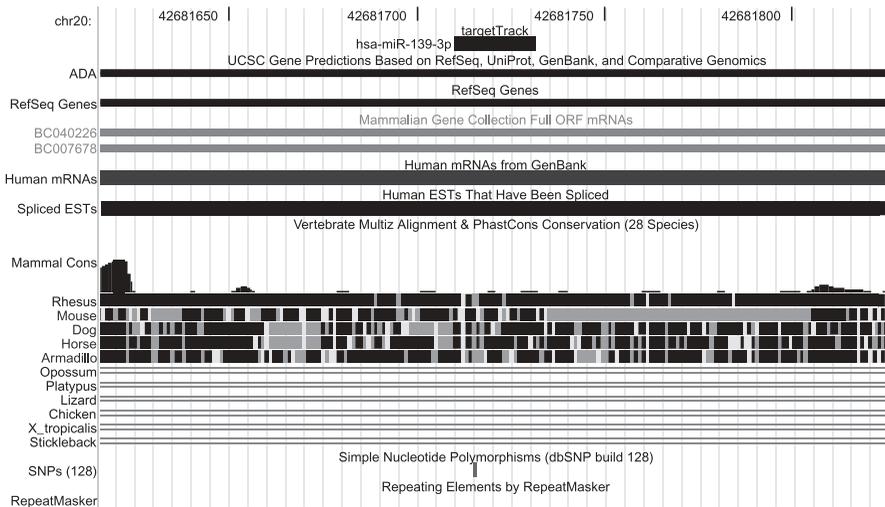


Figure 13.4 Genomic visualization produced by the UCSC genome browser by the `rtracklayer` example. The track named `targetTrack` showing microRNA target sites for the differentially expressed genes in the human stem cell experiment, was uploaded to the browser from R. We find, for example, that there is a SNP within the target site.

This provides the analyst with an overview of the relative positions and overlap of genomic annotations. From such an overview, the analyst might make inferences across genomic information sources. The GenomeGraphs package draws track-based visualizations using R graphics, while the exonmap package provides genomic visualization methods specific to exon array data. The rtracklayer package displays tracks by linking R/Bioconductor with existing, external genome browsers. The package defines a common interface through which the R user may control any genome browser for which a driver exists. The two primary capabilities of the genome browser interface are (1) transferring annotation tracks to and from genome browsers and (2) creating, manipulating, and querying the state of browser views. Currently, there are built-in drivers for the UCSC (Kent et al., 2002) and Argo (Engels, 2008) browsers; support may be added for any other browser by loading the appropriate driver (if one exists) into the R session.

13.4.3.3 Demonstration We demonstrate the use of rtracklayer by applying it to the analysis of the stem cell data. We create a track specifying the microRNA target sites associated with each differentially expressed gene and then view that track in the context of other genomic annotations using the UCSC genome browser.

The first step toward creating the microRNA track is to identify the target sites for each differentially expressed gene in the human stem cell microarray data. The differentially expressed genes have already been detected using limma, as described in Section 13.3.3, and stored as the `affyIDs`. To find genes regulated by a microRNA, we rely on the microRNA annotation package, which provides a mapping between each microRNA ID and the Ensembl Transcript ID for the transcripts the microRNA is predicted to regulate. We begin by loading the package and the mapping, a data.frame known as `hsTargets`, into R:

```
> library("microRNA")
> data(hsTargets)
> head(hsTargets)name target chrom start end strand
```

	name	target	chrom	start	end	strand
1	hsa-miR-647	ENST00000295228	2	120824263	120824281	+
2	hsa-miR-130a	ENST00000295228	2	120825363	120825385	+
3	hsa-miR-423-3p	ENST00000295228	2	120825191	120825213	+
4	hsa-miR-423-5p	ENST00000295228	2	120824821	120824843	+
5	hsa-miR-130b	ENST00000295228	2	120825364	120825385	+
6	hsa-miR-767-3p	ENST00000295228	2	120824258	120824278	+

We now encounter a complication: Each gene in the experimental dataset is identified by an Affymetrix probe ID for the HG-U133plus2 GeneChip, while the microRNA package is based on Ensembl Transcript IDs. There is no direct mapping in Bioconductor from an Affymetrix probe ID to an Ensembl Transcript ID, so we first map each Affymetrix ID to an Entrez Gene ID via the platform-specific annotation package, `hgu133plus2.db`. We then employ the Human Entrez Gene annotation package, `org.Hs.eg.db`, to map each Entrez Gene ID to an Ensembl Transcript ID.

The `hsTargets` data.frame contains columns for the ID, chromosome, start position, end position, and strand (+ or -) for each microRNA target site. In the code below, we subset `hsTargets` for the genes of interest, which are stored in the variable named `ensemblIDs`:

```
> targetMatches <- match(ensemblIDs, hsTargets$target, 0)
> targets <- hsTargets[targetMatches,]
```

We have now gathered enough information to construct the track indicating the positions of the microRNA target sites. We create the track from the annotations using the `RangedData` function:

```
> library(IRanges)
> targetTrack <- with(targets,
+                      RangedData(IRanges(start, end), strand, space = chrom))
```

To view the tracks in a genome browser, we first create an instance of `browserSession` for a specific external browser. The `browserSession` instance serves as a container of tracks as well as an interface for creating genomic views of specific segments of the genome. In this case, we will interact with the UCSC Genome Browser (the default). A session is created by calling the `browserSession` function:

```
> library("rtracklayer")
> session <- browserSession()
```

The second step is to load the track into the session by passing the track to session via the track accessor:

```
> track(session, "targets") <- targetTrack
```

As `targetTrack` does not contain any data values, it will be displayed as a black rectangle, by default.

The final step in this example is to display a view around the first target site. We wish to display the entire track, along with some context on either side. In the code below, we call `browserView` to create a view spanning the first feature of the track, zoomed out by a factor of 10:

```
> view <- browserView(session,
+                      ranges(targetTrack[1,]) * -10,
+                      pack = "targets")
```

This will open a web browser on the local machine and load the UCSC genome browser, with a view resembling the one in Figure 13.4. By default, the view will include our custom track (named `targetTrack`) along with the default UCSC tracks, including genes, mRNAs, cross-species alignments, and SNPs. We instruct the browser to show our custom microRNA target track in “pack” mode, so that the name of the microRNA is displayed in the track. We may now inspect the drawing generated by the genome browser. For example, we find that a SNP is within the target site of microRNA `hsa-miR-139-3p` for the gene named `ADA`.

13.4.4 Gene Set Enrichment Analysis

Gene set enrichment analysis (GSEA) is an important tool for analyzing gene expression data (Subramanian et al., 2005; Tian et al., 2005; Jiang and Gentleman, 2007). We consider the two-sample problem and compare the undifferentiated state to the differentiated state in the stem cell experiment. We want to provide some biological interpretation to the set of differentially expressed genes. The basic idea behind GSEA is to use predefined sets of genes, usually derived from functional annotation or from results of prior experiments, in order to better interpret the experiment that we are analyzing. With GSEA there is no need to make a hard cutoff between genes that are differentially expressed and those that are not. Rather, we use a score computed for all genes, for example, a t -statistic, and then determine whether these scores can be explained by gene set membership.

Any collection of gene sets can be used, and in many cases users will avail themselves of predefined gene sets, such as those available from GO (Gene Ontology Consortium, 2000) or KEGG (Kanehisa and Goto, 2000). In our example, we will use KEGG pathways as our gene sets.

The GSEABase package contains software infrastructure for performing GSEA analyses. The basic concepts are simple: the `GeneSet` class represents a set of genes that are all described by a common set of identifiers such as Entrez Gene IDs. A `GeneSetCollection` object is a collection of gene sets. GSEABase provides tools for performing set operations on gene sets, such as union and intersection, as well as ways to construct specific gene set collections.

There are a number of other Bioconductor packages that provide GSEA-type analyses. They include PGSEA, sigPathways, and GlobalAncova. For hypergeometric testing the packages GOSTats and topGO provide specialized methods that try to remove redundancy by taking advantage of the nested structure of the Gene Ontology.

Our KEGG analysis is quite simple:

```
> library("GSEABase")
> gsc <- GeneSetCollection(filtFhesc, setType = KEGGCollection())
> Am <- incidence(gsc)
> dim(Am)
```

```
[1] 202 1678
```

We next compute a reduced `ExpressionSet` object `nsF` that retains only those features (genes) that are mentioned in the incidence matrix `Am` and whose features are in the same order as the columns of `Am`:

```
> nsF <- filtFhesc[colnames(Am),]
```

In the next code segment, we reduce the incidence matrix by removing all gene sets that have fewer than five genes in them:

```
> selectedRows <- (rowSums(Am) > 5)
> Am2 <- Am[selectedRows, ]
```

Finding general trends requires gene sets with a reasonable number of genes, and here we have operationalized that by setting our cutoff at five. This cutoff is arbitrary; in general, the value of the cutoff, if any, should be chosen based on the question at hand.

Next we need to extract the t -statistics for the genes in Am2 from the summaries computed by limma:

```
> mm <- match(colnames(Am2), tab$ID)
> tts <- tab$t[mm]
> names(tts) <- colnames(Am2)
```

Now it is fairly easy to compute the per-gene set test statistics. High positive values indicate pathways which contain genes with positive t -statistics (and hence are higher in the differentiated group), while pathways with negative values correspond to ones where the expression of the genes is higher in the undifferentiated group. In the next code segment we compute the per-pathway summary statistics and then identify those that are interesting (our rather arbitrary cutoff is for the test statistic to exceed 3). We print out the names of three of the pathways that have higher expression in the differentiated cells:

```
> lms <- apply(Am2, 1, function(x) summary(lm(tts ~ x)))
> names(lms) <- rownames(Am2)
> ts <- sapply(lms, function(x) x$coefficients[2, 3])
> library("KEGG.db")
> lgPW <- ts[ts > 3]
> lgName <- mget(names(lgPW), KEGGPATHID2NAME)
> length(lgName)
```

```
[1] 11
```

```
> lgName [1:3]
```

```
$`00511`
```

```
[1] "N-Glycan degradation"
```

```
$`00531`
```

```
[1] "Glycosaminoglycan degradation"
```

```
$`01032`
```

```
[1] "Glycan structures - degradation"
```

These pathways are likely representative of the specialized functions performed by the newly differentiated cells. To find the pathways with decreased expression after differentiation, we list those in the opposite tail of the t -statistic distribution:

```
> smPW <- ts[ts < -3]
> smName <- mget(names(smPW), KEGGPATHID2NAME)
> length(smName)
```

```
[1] 11
```

```
> smName [1:3]
```

```

$`00230`
[1] "Purine metabolism"

$`00240`
[1] "Pyrimidine metabolism"

$`00271`
[1] "Methionine metabolism"

```

All of these pathways are directly related to cell replication, as regulated by the cell cycle. The synthesis of aminoacyl-tRNAs is important for protein production, and purines are necessary for the synthesis of nucleic acids (like DNA and RNA). These results are expected, given that the cells are in a growth medium (and thus dividing rather quickly) prior to differentiation.

13.5 MODELS OF BIOLOGICAL SYSTEMS

Bioconductor has now taken us from individual microarray probe intensities to a list of high-level pathways whose constituent elements seem to be expressed differently in differentiated and undifferentiated human stem cells. While many more complex questions about the sequence of events that trigger differentiation are of interest, we simply do not have enough knowledge of this system to address many of those questions. However, we do want to leverage the pathways as a framework for understanding the complex relationships between the experimental measurements at the systems level. For this, we turn to the field of systems biology. Models of biological systems often involve several different types of biological entities, such as genes, proteins, and metabolites, and in the future one anticipates that as candidates are identified more comprehensive experiments will be carried out to collect such data. However, for this discussion we are limited to the analysis of a single gene expression dataset, but when more data become available, these pathways will provide a means for integrating data from different sources.

Graphs and networks are among the richest and most amenable models to formalize the complex interactions and processes that drive a biological system (e.g., metabolic or signaling pathways). Graphs can be used to represent static relationships between biological entities, or, using systems of ordinary differential equations (ODEs), it can model the dynamic nature of life. There are three main Bioconductor packages for representing (graph), manipulating (RBGL), and rendering (Rgraphviz) graphs and networks. RBGL, among other features, provides a comprehensive interface to the Boost graph library (Siek et al., 2002). Rgraphviz interfaces with the Graphviz library (Gansner and North, 1999). The gaggle package can be used to send networks to Cytoscape, an open-source platform for network analysis and visualization (Shannon et al., 2003).

In this section, we illustrate the use of these and other Bioconductor tools for modeling biological systems. We describe support for computationally representing models, accessing remote sources of network data, drawing models as graphs, and analyzing models through qualitative graph-theoretic measures and quantitative simulation of dynamics.

13.5.1 Representing Models of Biological Systems

The Systems Biology Markup Language (SBML, <http://sbml.org>) is a widely used standard for annotating a reaction network with dynamics and biology-specific semantics. It is applicable to static and dynamic models of, for example, cell-signaling pathways, metabolic pathways, biochemical reactions, gene regulation, and many others. An SBML document contains a detailed specification of a biochemical model (e.g., the initial quantities for the entities, kinetic laws for the reactions, etc.).

Two Bioconductor packages, SBMLR (Radivoyevitch, 2004) and rsbml (Lawrence, 2007), support the loading, manipulating, simulating, and writing of SBML. The packages are similar in purpose but different in design: the rsbml leverages the external library libsbml (Bornstein et al., 2008), while SBMLR parses the SBML directly in R (XML file or character vector). In this chapter, we will demonstrate the use of rsbml, though most of these tasks are also supported, to some extent, by SBMLR.

The reader might recall that the pathways found by GSEA to be down-regulated by differentiation in our microarray experiment were directly tied to cell growth and division. Although the explanation for this, that the cells were encouraged to grow, seems rather obvious, for the purpose of this chapter we will assume that there is a biological interest in the changes. A simplified version of one of the pathways, purine metabolism, has been the subject of detailed mathematical modeling by Curto et al. (1998). The SBML model has been curated by the BioModels team and is available in the repository under the name BIOMD0000000015. We retrieve the model from the BioModels database using the functionality in R for accessing URLs:

```
> purineMetabolismUrl <-
+   "http://www.ebi.ac.uk/compneur-srv/biomodels/models-main/publ/BIOMD0000000015.xml"
> purineMetabolism <- paste(readLines(url(purineMetabolismUrl)), collapse = "")
```

To parse these data into an R object representing the SBML document, we call the function `rsbml_read` and store the result as `purineDom`:

```
> library("rsbml")
> purineDom <- rsbml_read(text = purineMetabolism)
```

Below, we show how various pieces, specifically the reaction and species IDs, of the model may be retrieved from the R representation of the SBML document:

```
> purineModel <- model(purineDom)
> reactionName <- names(reactions(purineModel))
> reactionName

[1] "ada"  "ade"  "adna" "adrnr" "ampd"  "aprt"  "arna" "asuc"
[9] "asli" "dada" "den"  "dgnuc" "dnaa"  "dnag"  "gdna" "gdrnr"
[17] "gmpr" "gmps" "gnuc" "gprr"  "grna"  "gua"   "hprr" "hx"
[25] "hxd"  "impd" "inuc" "mat"   "polyam" "prpps" "pyr"  "rnaa"
[33] "rnag" "trans" "ua"   "x"     "xd"

> speciesName <- names(species(purineModel))
> speciesName
```

```
[1] "PRPP" "IMP" "SAMP" "ATP" "SAM" "Ade" "XMP" "GTP" "dATP" "dGTP"
[11] "RNA" "DNA" "HX" "Xa" "Gua" "UA" "R5P" "Pi"
```

From the species names, one may recognize some of the various participants in purine metabolism. The important purines guanine (Gua) and adenine (Ade) are both synthesized from inosine monophosphate (IMP). They may then be derived to (d)GTP and (d)ATP, respectively, prior to incorporation into nucleic acids like DNA and RNA. This simple explanation betrays the true complexity of purine metabolism, and elucidating the properties of such complex systems depends on sophisticated computational tools, which we describe presently.

SBML models have an implicit graphical structure and hence rely on the `graph`, `RBGL`, and `Rgraphviz` packages for computational support. In order to make use of these packages, we reduce the SBML model to an instance of the `graph` class:

```
> purineGraph <- rsbml_graph(purineDom)
> purineGraph

A graphNEL graph with directed edges
Number of Nodes = 55
Number of Edges = 107
```

The model components, including both the chemical species and the reactions, are represented by nodes in the `graph`. Each species can be a reactant, product, or modifier of one or more reactions. Each such relationship is represented by an edge in the `graph`.

To make this more concrete, we will consider the subnetwork involving the synthesis of IMP from hypoxanthine (HX) and the sugar PRPP. This is one of the important first steps in purine metabolism. The reaction converting HX and PRPP to IMP is named `hprt`, from the enzyme that catalyzes the reaction. Here we retrieve the name of each of these nodes from the `purineGraph` object:

```
> sampleNodes <- nodes(purineGraph)[c(1, 13, 41, 2)]
> sampleNodes

[1] "PRPP" "HX" "hprt" "IMP"
```

Edges connect the `hprt` reaction to its reactants (PRPP and HX) and products (including IMP). Here we list the edges of the subgraph by indexing on the node names:

```
> edges(purineGraph)[sampleNodes]

$PRPP
[1] "aprt" "den" "gppt" "hprt" "prpps" "pyr"

$HX
[1] "hprt" "hx" "hxd"

$hprt
[1] "IMP"
```

```
$IMP
[1] "asuc" "den" "gmpr" "hprr" "impd" "inuc"
```

To better appreciate the complexity of the interactions between the components of this graph, we now render it. Laying out and rendering a graph are helpful for understanding its structure. Two steps are necessary for displaying graphs in `graph` and `Rgraphviz`: layout, which places nodes and edges in a (usually two-dimensional) space, and rendering, which is the actual drawing of the graph on a graphics device. Laying out a graph is often computationally expensive and relies on sophisticated algorithms that arrange the components based on different criteria (e.g., to minimize edge crossings). Rendering a graph applies aesthetic properties (such as the node and edge colors) to the geometric output of the layout algorithm.

First, we lay out the graph using the `layoutGraph` function from `Rgraphviz`, which by default delegates to the `dot` algorithm from `Graphviz`. As a hierarchical layout algorithm, `dot` is designed for use with directed graphs:

```
> library("Rgraphviz")
> purineGraph <- layoutGraph(purineGraph)
```

To render the graph layout on the screen, we call `renderGraph`, which yields Figure 13.5:

```
> renderGraph(purineGraph)
```

We have configured the aesthetic properties of the nodes in our graph object to better communicate the semantics of the biological reaction network. The fill color of the nodes is set to light gray for the reactions and light yellow for the species. Reactions are represented by rectangles and species by circles. If the reaction has no outgoing edges, it represents a sink, a flow of molecules exiting the system. We visually denote a sink by drawing the node border in dark gray, while all other nodes are given a red border. We also configure the dimensions of the node glyphs as well as the font used for the labels. We could similarly modify the edge attributes, which include the color, style, and shape of the edges, as well as whether an edge is directed or undirected. For more details on edge and node attributes we invite the reader to consult the `Graphviz` documentation (<http://www.graphviz.org/>).

In order to focus on the IMP synthesis reaction described above, we could lay out the subgraph surrounding the reaction node:

```
> smallNodes <- unique(c(unlist(edges(purineGraph)[sampleNodes]), sampleNodes))
> smallGraph <- subGraph(smallNodes, purineGraph)
> smallGraph <- layoutGraph(smallGraph)
```

Four different layouts of this graph are shown in Figure 13.6 and the user typically chooses the one that they feel best exposes the structures and relationships of interest:

```
> x <- layoutGraph(smallGraph, layoutType="circo")
> renderGraph(x, graph.pars=list(graph=list(main="circo")))

> x = layoutGraph(smallGraph, layoutType="fdp")
> renderGraph(x, graph.pars=list(graph=list(main="fdp")))
```

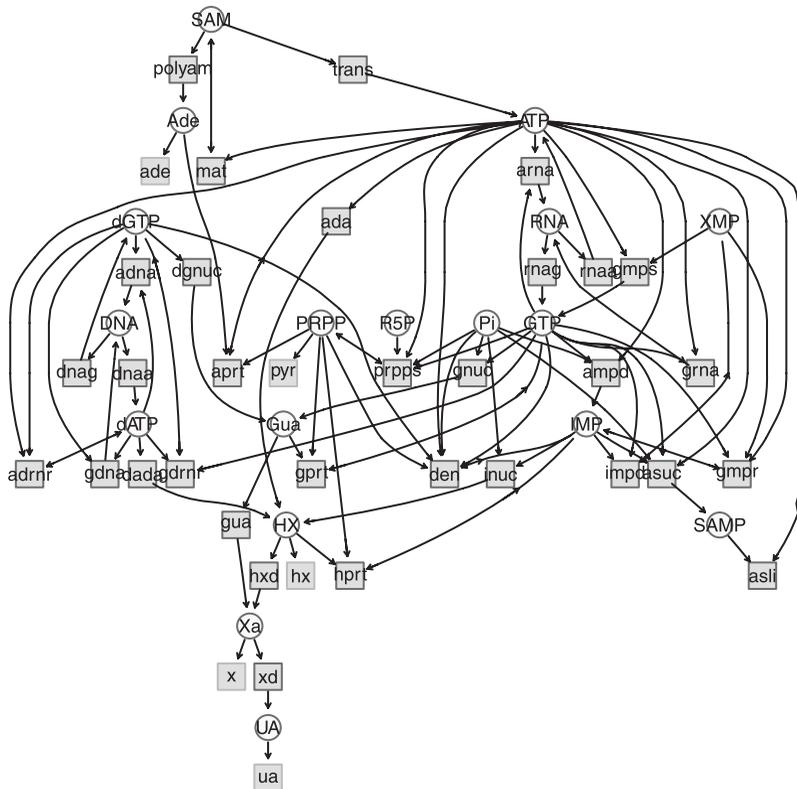


Figure 13.5 Graph drawing of human purine metabolism as proposed by Curto et al. (1998). Each node represents either a reaction (gray background boxes and lowercase labels) or a molecular species (yellow background boxes and uppercase names). An edge between a species and a reaction indicates that the species participates in the reaction. If the edge direction is toward the reaction, the species is a reactant; if the direction is toward the species, the species is a product. Bidirected edges indicate that the species is both a reactant and a product. (See color insert.)

As shown in the examples above we can color the graph components according to the graph properties. In our purine model we could also color the species nodes according to some metadata, such as the phosphorylated state of the proteins or their presence or absence at time t .

Alternatively we could use the **imageMap** function from the **geneplotter** package to produce a static graph that can be viewed in any web browser. In such an image one can embed tooltips, links to external resources such as KEGG, GO, or Entrez Gene. We provide an example of the purine metabolism network in our online supplement.

13.5.2 Accessing Remote Pathway Data

Computational modeling of biological systems involves the development and use of databases on cellular processes and functions of molecular species. The Pathguide

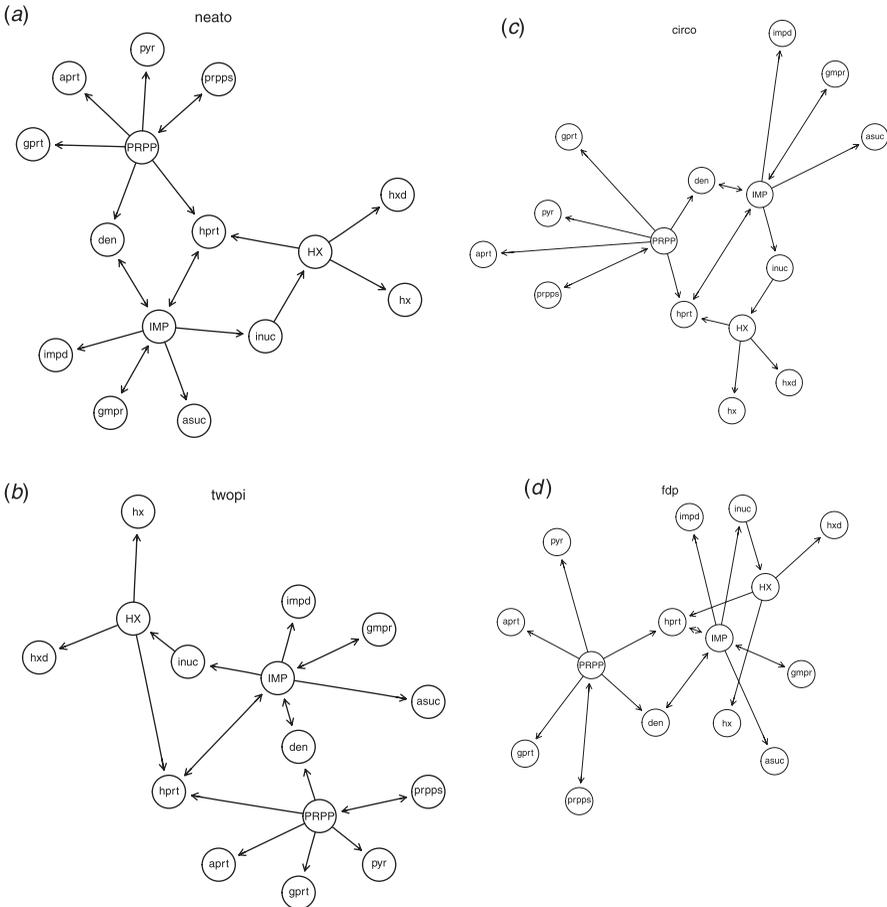


Figure 13.6 Different layouts, different information: four different layout algorithms (neato, twopi, circo, and fdp) applied to the same subgraph of the purine model proposed by Curto et al. (1998). The neato and fdp are “spring model” layouts, preferred for large undirected graphs. The twopi is a radial layout: the nodes are placed on concentric circles depending on their distance from a given root node. The circo is a circular layout, often suitable for diagrams of multiple cyclic structures.

repository currently lists more than 200 such databases (Bader et al., 2006). Among those databases, KEGG is probably one of the first developed and widely used source of data on pathways in different species (Kanehisa and Goto, 2000). More recently, the BioModels Database was developed to allow biologists to store, search, and retrieve published mathematical models of biological interest (<http://www.ebi.ac.uk/biomodels>). Models in the BioModels database are annotated and linked to relevant resources, such as publications, databases of compounds and pathways, and controlled vocabularies.

As for retrieving experimental data (Section 13.3.2), Bioconductor and R offer tools to access remote pathway databases. For instance, the basic url function creates,

opens, and closes connections to remote sources. Knowing the URL format used by the online repository, one can quickly download several models at once.

13.5.3 Simulating Quantitative Models

In the previous sections, we have focused on the representation, visualization, and analysis of static graphs and networks. Life, in contrast, is never static, so many models of biological systems are dynamic. A common way to analyze the dynamics of such models is simulation.

Bioconductor and R support the quantitative simulation of dynamic models at various levels. There are generic, low-level packages for solving differential equations in R. These include the `odesolve` and `R sundials` packages. One could leverage one of those packages, perhaps in combination with one or more nonlinear modeling functions R, such as `optim`, `nls`, `nlm` and `nlme`, to implement a custom simulation algorithm. At a higher level, the `rsbml` and `SBMLR` packages are both able to directly simulate SBML models.

When a model is in a steady state, perturbing the model, such as through a change in the concentration of a metabolite or enzyme, often incites informative time course responses. In the purine metabolism model, hypoxanthine (HX) reacts with phosphoribosylpyrophosphate (PRPP) to form inosine monophosphate (IMP). Let us assume we are interested in the response of HX and IMP to changes in the concentration of PRPP. We will investigate this by perturbing the model in its steady state such that the level of PRPP is set to 10 times its initial amount. Although this is a purely synthetic event, the perturbation allows us to analyze the dynamics of the model in response to external changes. These often occur in biological systems, such as through cell signaling or changes in gene expression.

The perturbation in the PRPP concentration will be represented in the model as an SBML Event. In the code below, we use `rsbml` to add an Event to the model that will cause the PRPP concentration to increase to 10 times its original amount after the simulation has run for 20 minutes, at which time we assume that the model has reached an approximate steady state:

```
> initialPRPP <- initialAmount(species(purineModel)$PRPP)
> assignment <- new("EventAssignment", variable = "PRPP",
+                   math = as.expression(10*initialPRPP))
> trigger <- new("Trigger", math = expression(.t == 20))
> event <- new("Event", trigger = trigger,
+             eventAssignments = list(assignment))
> purineDomPert <- purineDom
> events(model(purineDomPert)) <- list(event)
```

Next, we simulate the new `purineDomPert` model for 70 minutes, sampling each minute:

```
> simExpPert <- simulate(purineDomPert, 70)
```

The returned value `simExp` represents the simulation experiment and contains the simulated time courses for each participant in the model as well as the parameters of the simulation and the model itself.

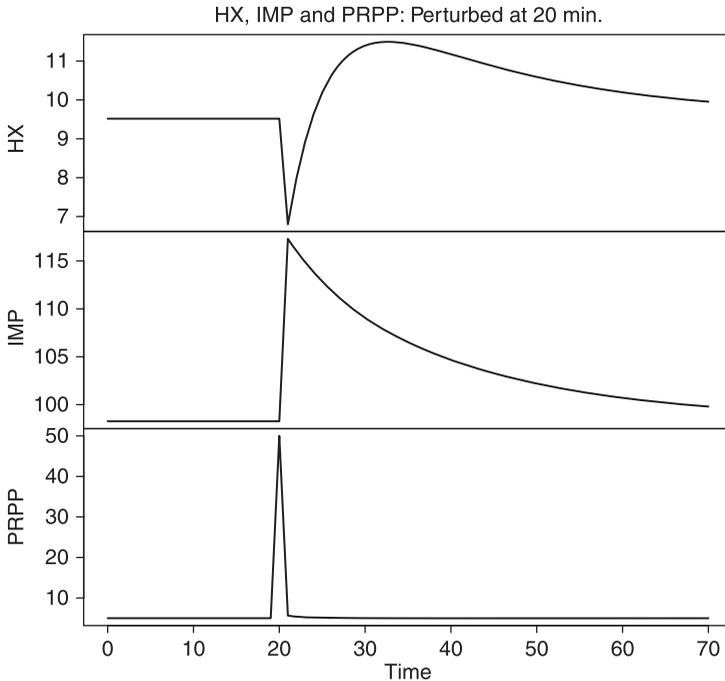


Figure 13.7 Kinetic response of HX and IMP concentration (μM) to a 10-fold increase of PRPP (Curto et al., 1998).

Finally, we coerce the result to an R time series (ts) object and plot the time courses for HX, IMP, and PRPP:

```
> simTsPert <- as.ts(result(simExpPert))
> plot(simTsPert[,c("HX", "IMP", "PRPP")],
+      main = "HX, IMP and PRPP: Perturbed at 20 min.")
```

The result is shown in Figure 13.7, and we observe dramatic shifts in the levels of HX and IMP in response to the perturbation in the PRPP level. As PRPP is a reactant, increasing its concentration accelerates the reaction, leading to an increase in the product IMP and a decrease in HX, the other reactant. The model then appears to slowly return to its steady state, after the excess PRPP has been consumed. Through this analysis we have gained a better understanding of the kinetics of IMP synthesis. If a change in PRPP concentration had actually been measured, for example, in a metabolomics experiment, we would have been able to make predictions on the rest of the purine metabolism pathway. These predictions might then have led to testable biological hypotheses.

13.6 CONCLUSION

We have demonstrated a number of tools in Bioconductor for the integrated analysis and visualization of experimental data, annotations, and networks in the context of the

biological system. Bioconductor provides data structures, preprocessing routines, analysis methods, and other infrastructure for a wide range of experimental data types, especially gene expression, but also metabolomics, proteomics, flow cytometry, and others. Additionally, there is support for accessing and representing annotations on experimental samples and features as well as general genomic annotations. Data analysis strategies can leverage this information to integrate experimental datasets and form hypotheses at the systems level. To refine and diagnose such a hypothesis and to explore its implications, the analyst can leverage Bioconductor data structures and functions for manipulating, visualizing, and simulating models of biological systems.

Among the many strengths of R and Bioconductor are its flexible data structures, sophisticated statistical methods, high-quality graphics, and infrastructure for accessing and importing data, including from remote databases. These features make R and Bioconductor a convenient and effective platform for exploratory data analysis and prototyping of data analysis approaches. In addition, the existence of more than 1000 add-on packages means that there is often at least a rudimentary implementation of a method available for experimentation. One deficiency of R is that its graphics system is not designed for interactivity, but one can control GGobi (Swayne et al., 2003) from R using `rggobi` (Temple Lang, 2001) or construct custom visualizations through bindings to graphical toolkits, like GTK+ (Lawrence and Temple Lang, 2008).

Computational biology provides a very rich set of interesting statistical and data-analytic problems, from preprocessing of experimental data to dynamical modeling. Addressing them will require developing effective statistical methods and delivering, in a convenient manner, the implementations of those methods to the biologists generating the data. We feel that R and Bioconductor provide a good environment in which to carry out this research.

13.7 ACKNOWLEDGMENTS

Funding for this research was provided by grants from the National Human Genome Research Institute (5P41 HG004059) and the Human Frontiers in Science program research Grant RGP0022/2005 to R. G. We also gratefully acknowledge funding from Oncology Training Grant 5 T32 CA009515-21/22 and Career Development in Pediatric and Medical Oncology award National Cancer Institute (NCI)/5 K12 CA076930 (to M. B.), and a Pilot Award from the NIH/NCI Cancer Center Support Grant 5 P30 CA015704 and Fred Hutchinson Cancer Research Center New Development funds (to M. T.). We would like to thank Wolfgang Huber and Vincent Carey and the reviewer for helpful comments.

REFERENCES

- Bader, G. D., Cary, M. P., Sander, C., and Journals, O. (2006). Pathguide: A pathway resource list. *Nucleic Acids Res.*, **34**(Database issue): D504–D506.
- Bar, M., et al. (2008). MicroRNA discovery and profiling in human embryonic stem cells by deep sequencing of small RNA libraries. *Stem Cells*, **26**: 2496–2505.

- Bornstein, B. J., Keating, S. M., Jouraku, A., and Hucka, M. (2008). LibSBML: An API Library for SBML. *Bioinformatics*, **24**: 880–881.
- Carey, V. J., Morgan, M., Falcon, S., Lazarus, R., and Gentleman, R. (2007). GGtools: Analysis of genetics of gene expression in Bioconductor. *Bioinformatics*, **15**: 522–523.
- Cheung, V. G., Spielman, R. S., Ewens, K. G., Weber, T. M., Morley, M., and Burdick, J. T. (2005). Mapping determinants of human gene expression by regional and genome-wide association. *Nature*, **437**: 1365–1369.
- Chiang, T., Scholtens, D., Sarkar, D., Gentleman, R., and Huber, W. (2007). Coverage and error models of protein–protein interaction data by directed graph analysis. *Genome Biol.*, **8**(9): R186.
- Cook, D., Lawrence, M., Lee, E.-K., Babka, H. and Syrkin Wurtele, E. (2008). Explorase: Multivariate exploratory analysis and visualization for systems biology. *J. Stat. Software*, **25**(9): 1–23.
- Curto, R., Voit, E. O., Sorribas, A., and Cascante, M. (1998). Mathematical models of purine metabolism in man. *Math. Biosci.*, **151**(1): 1–49.
- Davis, S., and Meltzer, P. S. (2007). GEOquery: A bridge between the Gene Expression Omnibus (GEO) and BioConductor. *Bioinformatics*, **23**(14): 1846–1847.
- Du, P., Kibbe, W. A., and Lin, S. M. (2006). Improved peak detection in mass spectrum by incorporating continuous wavelet transform-based pattern matching. *Bioinformatics*, **22**(17): 2059.
- Durbin, R., Haussler, D., Stein, L., Lewis, S., and Krogh, A. (2000). *General Feature Format*. Sanger Institute, September. Available: URL http://www.sanger.ac.uk/Software/formats/GFF/GFF_Spec.shtml.
- Durinck, S., Moreau, Y., Kasprzyk, A., Davis, S., De Moor, B., Brazma, A., and Huber, W. (2005). BioMart and Bioconductor: A powerful link between biological databases and microarray data analysis. *Bioinformatics*, **21**(16): 3439–3440.
- Engels, R. (2008). *Argo Genome Browser*, URL <http://www.broadinstitute.org/annotation/argo/Version 1.0.25>.
- Gansner, E. R., and North, S. C. (1999). An open graph visualization system and its applications to software engineering. *Software Practice and Experience*, **30**: 1203–1233.
- Gene Ontology Consortium (2000). Gene Ontology: Tool for the unification of biology. *Nat. Genet.*, **25**: 25–29.
- Gentleman, R., Huber, W., Carey, V., Irizarry, R., and Dudoit, S. (Eds.) (2005). *Bioinformatics and Computational Biology Solutions Using R and Bioconductor*. New York, NY: Springer.
- Hahne, F., Huber, W., Gentleman, R., and Falcon, S. (Eds.) (2008). *Bioconductor Case Studies*. New York: Springer.
- He, L., and Hannon, G. J. (2004). MicroRNAs: Small RNAs with a big role in gene regulation. *Nat. Rev. Genet.*, **5**: 522–531.
- Jiang, Z., and Gentleman, R. (2007). Extensions to gene set enrichment. *Bioinformatics*, **23**: 306–313.
- Kanehisa, M., and Goto, S. (2000). Kegg: Kyoto encyclopedia of genes and genomes. *Nucleic Acids Res.*, **28**: 27–30.
- Kent, W. J., Sugnet, C. W., Furey, T. S., Roskin, K. M., Pringle, T. H., Zahler, A. M., and Haussler, D. (2002). The human genome browser at ucsc. *Genome Res.*, **12**(6): 996–1006.
- Lawrence, M. (2007). Quick introduction to the rsbml package. Available: <http://bioconductor.org>.
- Lawrence, M., and Temple Lang, D. (2008). *RGtk2: R Bindings for Gtk 2.8.0 and Above*. Available: URL <http://cran.r-project.org/web/packages/RGtk2/index.html>. R package version 2.12.5-3. 5-3.
- Leisch, F. (2002). Sweave: Dynamic generation of statistical reports using literate data analysis. In *Compstat 2002—Proceedings in Computational Statistics*, W. Härdle and B. Rönz (Eds.). Heidelberg: Physika Verlag, pp. 575–580. Available: URL <http://www.ci.tuwien.ac.at/~leisch/Sweave>. ISBN 3-7908-1517-9.
- Radvoyevitch, T. (2004). BMC Bioinformatics. *BMC Bioinform.*, **5**: 190.
- Sanges, R., Cordero, F., and Calogero, R. A. (2007). oneChannelGUI: A graphical interface to Bioconductor tools, designed for life scientists who are not familiar with R language. *Bioinformatics*, **23**(24): 3406–3408.
- Scholtens, D., Vidal, M., and Gentleman, R. (2005). Local modeling of global interactome networks. *Bioinformatics*, **21**(17): 3548–3557.

- Shannon, P., Markiel, A., Ozier, O., Baliga, N. S., Wang, J. T., Ramage, D., Amin, N., Schwikowski, B., and Ideker, T. (2003). Cytoscape: A software environment for integrated models of biomolecular interaction networks. *Genome Res.*, **13**(11): 2498.
- Siek, J. G., Lee, L.-Q., and Lumsdaine, A. (2002). *The Boost Graph Library*. Reading, MA: Addison Wesley.
- Smith, C. A., Want, E. J., Maille, G. O., Abagyan, R., and Siuzdak, G. (2006). XCMS: Processing mass spectrometry data for metabolite profiling using nonlinear peak alignment, matching, and identification. *Anal. Chem.*, **78**(3): 779–787.
- Smyth, G. K. (2004). Linear models and empirical Bayes methods for assessing differential expression in microarray experiments. *Stat. Applicat. Genet. Mol. Bio.*, **3**: Article 3.
- Subramanian, A., Tamayo, P., Mootha, V. K., et al. (2005). Gene set enrichment analysis: A knowledge-based approach for interpreting genome-wide expression profiles. *Proc. Natl. Acad. Sci. U.S.A.*, **102**(43): 15545–15550.
- Swayne, D. F., Temple Lang, D., Buja, A., and Cook, D. (2003). GGobi: Evolving from XGobi into an extensible framework for interactive data visualization. *Computat. Stat. Data Anal.*, **43**: 423–444.
- Temple Lang, D. (2001). GGobi meets R: An extensible environment for interactive dynamic data visualization. In *Proceedings of the 2nd International Workshop on Distributed Statistical Computing*.
- Tian, L., Greenberg, S. A., Kong, S. W. et al. (2005). Discovering statistically significant pathways in expression profiling studies. *Proc. Natl. Acad. Sci. U.S.A.*, **102**(38): 13544–13549.
- UCSC Genome Bioinformatics Group, (2009a). *Browser Extended Display Format*, Available: URL <http://genome.ucsc.edu/goldenPath/help/customTrack.html#BED>.
- UCSC Genome Bioinformatics Group, (2009b). *Wiggle Format*, Available: URL <http://genome.ucsc.edu/goldenPath/help/wiggle.html>.

INDEX

- Accessing data (bioconductor packages):
 experimental data, 312–313
 remote data source annotation, 320–322
 remote pathway data, 332–334
- Ad hoc tests, statistical testing and
 significance (for large biological
 data analysis), 76–77
- Admixture mapping, homozygosity and,
 genomewide association studies, 298
- Advanced clustering algorithms, 111–115
- Algorithms, numerical, 191–192. *See also*
 specific algorithms
- Alleles, genomewide association studies,
 283–285
- Analysis of variance (ANOVA) models,
 statistical/probabilistic models,
 194–196
- Annotation (bioconductor packages),
 318–328
 feature annotation, 319–322
 sample annotation, 318–319
- ANOVA models, statistical/probabilistic
 models, 194–196
- Artificial neural networks, classification and
 prediction methods, 135–136
- Asymptotic distribution, sampling
 distribution, 47
- Axis-parallel projection methods,
 two-dimensional projections, 162–165
- Bayes classifier, naive, 132
- Bayesian belief network, 259–273
 applications, 269–270
 framework of, 259–264
 generally, 259
 hierarchical models, 270–271
 learning problems, 264–268
 software, 272
- Bayes's theorem, probability concepts and
 distributions, 12–13
- Bernoulli distribution, discrete probability
 distributions, random variables
 distributions, 21
- Best-fit extension problem, Boolean
 network modeling, 253
- Beta distribution, continuous distributions,
 random variables distributions, 37–39
- BIB design, microarray technology blocking,
 213–214
- Biclustering, advanced clustering algorithms,
 114–115
- Binomial distribution, discrete probability
 distributions, 21–23
- Bioconductor packages, 309–338. *See also*
 R packages; Software
 annotation, 318–328
 feature annotation, 319–322
 gene set enrichment, 326–328
 genomic annotation, 323–325
 sample annotation, 318–319
 biological system models, 328–335
 generally, 328
 remote pathway data access,
 332–334
 representing models, 329–332
 simulations, 334–335
 experimental data, 311–318
 accessing data, 312–313
 dataset example, 313–316
 generally, 311–312
 infrastructure, 312
 metabolomics data, 317–318
 microarray data sources, 316–317
 proteomics data, 317
 overview, 309–310
 project description, 310–311

- Biological system models (bioconductor packages), 328–335
 - generally, 328
 - remote pathway data access, 332–334
 - representing models, 329–332
 - simulations, 334–335
- Blocking:
 - experimental design, 210–214
 - microarray technology, 211–214
- Bonferroni procedure, multiple testing, error controlling, 80–81
- Boolean network modeling, 250–259
 - computation times, 253–254
 - formulation of, 251
 - generally, 250–251
 - inference algorithms for, 252–253
 - probabilistic networks, 254–258
- Boosting methods, classification and prediction methods, 138–139
- Bootstrap methods:
 - empirical distribution and, sampling distribution, 48–54
 - quantifying uncertainty, statistical resampling techniques, 236–238, 243
 - statistical resampling techniques, 232–233
- Breast cancer data, classification methods comparison, 147–148
- Classification. *See* Supervised learning
- Classification algorithms, statistical resampling-based, 226
- Classification and prediction methods, 132–140. *See also* Supervised learning
 - artificial neural networks, 135–136
 - Bayes classifier, naive, 132
 - boosting methods, 138–139
 - classification by ensembles from random partitions (CERP), 139–140
 - classification trees, 136–137
 - k -nearest-neighbor classifiers, 134
 - linear discriminant analysis, 133–134
 - logistic regression, 132–133
 - random forest, 137
 - shrunk centroid, 134–135
 - supervised learning:
 - comparisons, 147–150
 - breast cancer data, 147–148
 - gastrointestinal bleeding data, 148–150
 - software examples, 150–153
 - support vector machines, 138
- Classification by ensembles from random partitions (CERP), classification and prediction methods, 139–140
- Classification studies, experimental design, 214–215
- Classification trees, classification and prediction methods, 136–137
- Class prediction enhancement, ensemble voting methods, supervised learning, 145–147
- Clustering, 89–127
 - algorithms, 104–115
 - advanced, 111–115
 - conventional, 104–111
 - experimental design, 214–215
 - overview, 89–90
 - quality assessment, 115–122
 - external measures, 117–122
 - generally, 115–116
 - internal measures, 116–117
 - predictive strength and stability measures, 122
 - similarity measures, 90–99
 - multidimensional objects, 90–95
 - distance-based, 90–92
 - matching-based, 92–95
 - sequences, 95–99
 - global sequence alignment, 95–97
 - local sequence alignment, 97–99
 - structural data, 99
 - types of, 100–104
 - crisp versus fuzzy, 102
 - hierarchical versus nonhierarchical, 101–102
 - one-way versus two-way clustering, 102–104
 - unsupervised and supervised classification, 99–100
- Complete block design, experimental design, 210–211
- Computation times, Boolean network modeling, 253–254
- Conditional independence, Bayesian belief network, 261–263
- Conditional probability, probability concepts and distributions, 10–11
- Conditional probability distributions, Bayesian belief network, 261

- Consistency problem, Boolean network modeling, 252
- Constant shift, data normalization, 65–66
- Continuous distributions (random variables distributions), 30–39
 - beta distribution, 37–39
 - exponential distribution, 34–36
 - gamma distribution, 36–37
 - normal distribution, 32–34
 - uniform distribution, 30–32
- Continuous random variables, probability concepts and distributions, 15
- Control structures identification, metabolic network modeling, 277–278
- Conventional clustering algorithms, 104–111
- Copy number variation (CNV) analysis,
 - genomewide association studies, 292–293
- Correlation, covariance and, probability concepts and distributions, 18–19
- Correlation coefficient, scatterplot criteria, two-dimensional scatterplot ordering, 176
- Covariance, correlation and, probability concepts and distributions, 18–19
- Crisp clustering, fuzzy clustering versus, 102
- Crisp clustering concept, advanced clustering algorithms, 112
- Cross-validation methods:
 - statistical resampling techniques, 228–232, 244–247
 - supervised learning, 144–145
- Curvilinear regression, least-squares error for, scatterplot criteria, 176

- Data access. *See* Accessing data (bioconductor packages)
- Data mining, multidimensional visualization analysis, 165–166
- Data normalization, high-throughput biological data quality control, 65–66
- Data processing, genomewide association studies, 290–292
- Data structure, genomic annotation, bioconductor packages, 323
- Data transformation, high-throughput biological data quality control, 59–65
- Demonstration, genomic annotation, bioconductor packages, 324–325
- Dirichlet [Dir(alpha)] distribution,
 - multivariate distribution, 45–46
- Discrete probability distributions (random variables distributions), 20–30
 - Bernoulli distribution, 21
 - binomial distribution, 21–23
 - geometric distribution, 24–27
 - negative binomial distribution, 27–29
 - Poisson distribution, 29–30
 - uniform distribution, 23–24
- Discrete random variables, probability concepts and distributions, 14–15
- Disease risk estimates, genomewide association studies, 301
- Distance-based similarity measures, multidimensional objects, 90–92

- Embedded methods, feature selection, statistical resampling techniques, 225–226
- Empirical distribution, bootstrapping and, sampling distribution, 48–54
- Ensembles from random partitions, classification by (CERP), 139–140
- Ensemble voting methods, class prediction enhancement by, 145–147
- Enzyme-centric graph, metabolic network modeling, 274
- Error controlling, 78–81
 - multiple testing, 79–81
 - p*-values, 78–79
- Error sources, high-throughput biological data quality control, 57–58
- Estimation methods, 189–191
- Expectation–maximization (EM) algorithm, numerical algorithms, 191–192
- Expected value, probability concepts and distributions, 16–17
- Experimental design, 201–217
 - blocking, 210–214
 - complete and incomplete block design, 210–211
 - microarray technology, 211–214
 - classifications and clustering studies, 214–215
 - defined, 201
 - pooling, 209
 - quantitative trait locus (QTL) studies, 215–216
 - randomization, 201–202

- Experimental design (*Continued*)
 - replication, 202–209
 - general principle, 202–204
 - multilevel, resource allocation and, 208–209
 - sample size and power, 204–205
 - software packages, 205–208
 - time course experiments, 215
- Experimental unit, replication, experimental design, 202–203
- Exponential distribution, continuous
 - distributions, random variables
 - distributions, 34–36
- External measures, clustering quality assessment, 117–122

- False discovery, genomewide association studies, 293–294
- Feature annotation, bioconductor packages, 319–322
- Feature selection:
 - statistical resampling techniques, 225–226
 - supervised learning, 140–144
 - comparison of, 143–144
 - variable importance ranking algorithms, 140–143
- Filter methods, feature selection, statistical resampling techniques, 225–226
- Fold change approach, statistical testing for large biological data analysis, 72–73
- Fuzzy clustering:
 - advanced clustering algorithms, 112–113
 - crisp clustering versus, 102

- Gamma distribution, continuous distributions, random variables distributions, 36–37
- Gap size:
 - one-dimensional histogram ordering, 173–174
 - two-dimensional scatterplot ordering, 181
- Gastrointestinal bleeding data, classification methods comparison, 148–150
- Gaussian distribution. *See* Multivariate normal (Gaussian) distribution
- Gene and pathway-based analysis, genomewide association studies, 299–300
- Gene set enrichment, bioconductor packages annotation, 326–328
- Gene \times gene and gene \times environment interactions, genomewide association studies, 298–299
- Genomewide association studies, 283–308
 - alleles, linkage disequilibrium, and haplotype, 283–285
 - disease risk estimates, 301
 - gene and pathway-based analysis, 299–300
 - gene \times gene and gene \times environment interactions, 298–299
 - genotyping platforms, 286–287
 - haplotype analysis, 296–298
 - homozygosity and admixture mapping, 298
 - international HapMap project, 285–286
 - meta-analysis, 301–302
 - overview, 283, 302–303
 - rare variants and sequence-based analysis, 302
 - results overview, 287–289
 - statistical issues in, 290–296
 - CNV analysis, 292–293
 - data processing and quality control, 290–292
 - imputation methods, 295–296
 - multiple-comparison adjustments, 293–294
 - sample heterogeneity, 294–295
- Genomic annotation, bioconductor packages, 323–325
- Genotyping platforms, genomewide association studies, 286–287
- Geometric distribution, discrete probability distributions, 24–27
- Gibbs sampler, Markov chain Monte Carlo methods, 239–240
- Global sequence alignment, similarity measures, 95–97
- Graphical network structure, Bayesian belief network, 259–260
- Graphical representation, metabolic network modeling, 273–274
- Graphical user interface (GUI):
 - bioconductor package, dataset example, 316
 - one-dimensional histogram ordering, 170–171

- Haplotype analysis, genomewide association studies, 283–285, 296–298
- HapMap project, genomewide association studies, 285–286, 295–296
- Hardy–Weinberg equilibrium (HWE), genomewide association studies, 290–292
- HEM (hierarchical error model), ad hoc tests, 77
- Heterogeneous, multiple data integration, statistical bioinformatics, 3–4
- Hierarchical Bayesian models, 270–271
- Hierarchical clustering:
 - conventional clustering algorithms, 104–106
 - nonhierarchical clustering versus, 101–102
- High-dimensional biological data, statistical bioinformatics, 2–3
- High-throughput biological data quality control, 57–70
 - error sources, 57–58
 - experimental design, 201–217. *See also* Experimental design
 - microarray gene expression experiments, 66–69
 - statistical techniques, 59–66
 - data normalization, 65–66
 - data transformation, 59–65
 - generally, 59
- Homozygosity, admixture mapping and, genomewide association studies, 298
- Imputation methods, genomewide association studies, 295–296
- Incomplete block design, experimental design, 210–211
- Independence, probability concepts and distributions, 11–12
- Inference, Bayesian belief network, 263–264
- Inference algorithms, Boolean network modeling, 252–253
- Integration problems, multiple, heterogeneous data, statistical bioinformatics, 3–4
- Integrative network modeling, statistical network analysis, 250
- Interactive tools, multidimensional visualization analysis, 165–166
- Internal measures, clustering quality assessment, 116–117
- International HapMap project, genomewide association studies, 285–286, 295–296
- Interquartile range normalization, data normalization, high-throughput biological data quality control, 66
- Jointly distributed random variables, probability concepts and distributions, 39–40
- Joint probability density function, probability concepts and distributions, 40–41
- Joint probability distributions, Bayesian belief network, 261
- Joint probability mass function, probability concepts and distributions, 40–41
- K*-means clustering, conventional clustering algorithms, 106–108, 111
- k*-Nearest-neighbor classifiers, classification and prediction methods, 134
- Kruskal–Wallis test, statistical testing for large biological data analysis, 76
- Large biological data analysis, statistical testing and significance for, 71–88. *See also* Statistical testing and significance (for large biological data analysis)
- Large-*p* problem, small-*n* problem and, statistical bioinformatics, 3
- Learning problems, Bayesian belief network, 264–268
- Learning-test split (holdout) method, statistical resampling techniques, 227–228
- Least-squares error, for curvilinear regression, scatterplot criteria, two-dimensional scatterplot ordering, 176
- Leave-one-out cross-validation methods, statistical resampling techniques, 230–231
- Linear discriminant analysis, classification and prediction methods, 133–134
- Linear modeling, bioconductor package, dataset example, 315
- Linkage disequilibrium, genomewide association studies, 283–285
- Local pooled error (LPE) test, ad hoc tests, 77
- Local sequence alignment, similarity measures, 97–99

- Logistic regression, classification and prediction methods, 132–133
- Loop design, microarray technology blocking, 211–213
- Loss functions, statistical resampling techniques, 235–236
- Low-dimensional projections, multidimensional visualization analysis, 166–170
- LPE (local pooled error) test, ad hoc tests, 77
- Lymphoma example, statistical resampling techniques, 226–227

- Mann–Whitney–Wilcoxon test, statistical testing for large biological data analysis, 75–76
- Marginal distribution, probability concepts and distributions, 41
- Markov chain Monte Carlo methods:
 - Bayesian belief network, 265
 - statistical resampling techniques, 238–240, 243–244
- Matching-based similarity measures, multidimensional objects, 92–95
- Mathematical derivations, sampling distribution, 47
- Maximum-likelihood estimation, numerical algorithms, 191–192
- Memoryless property:
 - exponential distribution, continuous distributions, 35–36
 - geometric distribution, discrete probability distributions, 26–27
- Meta-analysis, genomewide association studies, 301–302
- Metabolic network modeling, 273–279
 - concepts, 274–277
 - control structures and robustness identification, 277–278
 - future prospects, 279
 - generally, 273
 - graphical representation, 273–274
 - topology, 278–279
- Metabolomics data, bioconductor packages, 317–318
- Metadata packages, bioconductor packages, feature annotation, 319–320
- Metropolis algorithm, Markov chain Monte Carlo methods, statistical resampling techniques, 239–240
- Metropolis–Hastings algorithms, Markov chain Monte Carlo methods, statistical resampling techniques, 239–240
- Michaelis–Menten kinetics, statistical/probabilistic models, 187–188
- Microarray gene expression experiments:
 - ad hoc tests, 76–77
 - high-throughput biological data quality control, 66–69
 - replication:
 - experimental design, 203–204
 - software packages for, 205–208
- Microarray technology. *See also* Statistical resampling techniques
 - bioconductor packages, 316–317
 - blocking, experimental design, 211–214
 - statistical resampling techniques, 219
- Model-based clustering, advanced clustering algorithms, 113–114
- Model selection, performance assessment and, statistical resampling techniques, 222–224, 240–243
- Monte Carlo cross-validation methods, statistical resampling techniques, 231–232, 243–244
- Motivation, statistical/probabilistic models, 187–188
- Multidimensional objects, 90–95
 - distance-based, 90–92
 - matching-based, 92–95
- Multidimensional scaling, multidimensional visualization analysis, 160–161
- Multidimensional visualization analysis, 157–184
 - classical techniques, 158–161
 - multidimensional scaling, 160–161
 - principal-component analysis, 158–160
 - future prospects, 165–166
 - low-dimensional projections, 166–170
 - one-dimensional histogram ordering, 170–174
 - graphical user interface, 170–171
 - ordering criteria, 171–174
 - overview, 157–158
 - two-dimensional projections, 161–165
 - axis-parallel projection methods, 162–165
 - generally, 161

- non-axis-parallel projection methods, 161–162
 - two-dimensional scatterplot ordering, 174–183
 - generally, 174–176
 - scatterplot criteria, 176–180
 - transformations and potential ranking criteria, 180–181
- Multilevel replication, experimental design, resource allocation and, 208–209
- Multinomial distribution, multivariate distribution, 43–44
- Multiple-comparison issue:
 - adjustments, genomewide association studies, 293–294
 - statistical bioinformatics, 1–2
- Multiple data, heterogeneous data integration, statistical bioinformatics, 3–4
- Multiple testing, error controlling, 79–81
- Multivariate distribution, 42–46
 - Dirichlet [Dir(α)] distribution, 45–46
 - generally, 42–43
 - multinomial distribution, 43–44
 - multivariate normal (Gaussian) distribution, 44–45
- Multivariate models, random variables distributions, 19
- Multivariate normal (Gaussian) distribution, multivariate distribution, 44–45
- Naive Bayes classifier, 132
- Negative binomial distribution, discrete probability distributions, 27–29
- Noisy data:
 - advanced clustering algorithms, 111
 - high-throughput, statistical bioinformatics, 3
- Non-axis-parallel projection methods, two-dimensional projections, 161–162
- Nonhierarchical clustering, hierarchical clustering versus, 101–102
- Nonparametric bootstrap, statistical resampling techniques, 237–238
- Nonparametric regression-based normalization, data normalization, 66
- Nonparametric statistical models, 188
- Nonparametric statistical tests, 75–76
- Nonspecific filtering, bioconductor package, dataset example, 314
- Normal distribution:
 - continuous distributions, random variables distributions, 32–34
 - one-dimensional histogram ordering, 172
- Numerical algorithms, 191–192
- One-dimensional histogram ordering, 170–174
 - graphical user interface, 170–171
 - ordering criteria, 171–174
- One-way clustering:
 - advanced clustering algorithms, 112
 - two-way clustering versus, 102–104
- Ordering criteria, one-dimensional histogram ordering, 171–174
- Outliers:
 - advanced clustering algorithms, 111
 - one-dimensional histogram ordering, 172
 - scatterplot criteria, two-dimensional scatterplot ordering, 177–178, 180–181
- Paired local pooled error (PLPE) test, ad hoc tests, 77
- Parameter estimation methods, 189–191
- Parametric bootstrap, statistical resampling techniques, 238
- Parametric statistical models, 188–189
- Parametric statistical tests, 73–75
- Pathway-based analysis, genomewide association studies, 299–300
- Performance assessment, model selection and, 222–224
- PLPE (paired local pooled error) test, ad hoc tests, 77
- Poisson distribution, discrete probability distributions, 29–30
- Pooling, experimental design, 209
- Potential ranking criteria, two-dimensional scatterplot ordering, 180–181
- Power, sample size, experimental design, 204–205
- Prediction error assessment, statistical resampling techniques, 221–222, 240–243, 244–247
- Predictive strength measures, clustering, quality assessment, 122

- Principal-component analysis (PCA):
 genomewide association studies, 295
 multidimensional visualization analysis,
 classical techniques, 158–160
 survival models, 197–198
- Probabilistic Boolean networks, modeling,
 254–258
- Probability concepts and distributions, 7–55.
See also Random variables distributions
 Bayes's theorem, 12–13
 concepts, 9–10
 conditional probability, 10–11
 covariance and correlation, 18–19
 expected value, 16–17
 independence, 11–12
 jointly distributed random variables,
 39–40
 joint probability mass or density function,
 40–41
 marginal distribution, 41
 multivariate distribution, 42–46
 Dirichlet [Dir(α)] distribution,
 45–46
 generally, 42–43
 multinomial distribution, 43–44
 multivariate normal (Gaussian)
 distribution, 44–45
 overview, 7–8
 probability, 9–10
 random variables, 13–15
 continuous, 15
 discrete, 14–15
 random variables distributions, 19–39
 continuous distributions, 30–39
 beta distribution, 37–39
 exponential distribution, 34–36
 gamma distribution, 36–37
 normal distribution, 32–34
 uniform distribution, 30–32
 discrete probability distributions, 20–30
 Bernoulli distribution, 21
 binomial distribution, 21–23
 geometric distribution, 24–27
 negative binomial distribution, 27–29
 Poisson distribution, 29–30
 uniform distribution, 23–24
 generally, 19–20
 sampling, 8–9
 sampling distribution, 46–54
 asymptotic distribution, 47
 definitions, 46
 empirical distribution and bootstrapping,
 48–54
 mathematical derivations, 47
 simulation studies, 47–48
 variance and standard deviation, 17
- Proteomics data, bioconductor
 packages, 317
- P*-values, error controlling, 78–79
- Quadracity, scatterplot criteria,
 two-dimensional scatterplot
 ordering, 176–177
- Quality assessment (clustering):
 external measures, 117–122
 generally, 115–116
 internal measures, 116–117
 predictive strength and stability
 measures, 122
- Quality control, genomewide association
 studies, 290–292. *See also*
 High-throughput biological data quality
 control
- Quantitative gene network modeling,
 statistical network analysis, 249–250
- Quantitative network modeling, statistical
 network analysis, 250
- Quantitative trait locus (QTL) studies,
 experimental design, 215–216
- Random forest, classification and prediction
 methods, 137
- Randomization, experimental design,
 201–202
- Random variables, 13–15
 continuous, probability concepts and
 distributions, 15
 discrete, probability concepts and
 distributions, 14–15
 jointly distributed, probability concepts and
 distributions, 39–40
- Random variables distributions, 19–39. *See
 also* Probability concepts and
 distributions
 continuous distributions, 30–39
 beta distribution, 37–39
 exponential distribution, 34–36
 gamma distribution, 36–37
 normal distribution, 32–34
 uniform distribution, 30–32

- discrete probability distributions, 20–30
 - Bernoulli distribution, 21
 - binomial distribution, 21–23
 - geometric distribution, 24–27
 - negative binomial distribution, 27–29
 - Poisson distribution, 29–30
 - uniform distribution, 23–24
- generally, 19–20
- Rare variants, sequence-based analysis
 - and, genomewide association studies, 302
- Reaction graph, metabolic network modeling, 273
- Real data analysis, 81–87
 - multiple samples, 85–87
 - paired samples, 87
 - tables, 81–84
 - two samples, 84–85
- Reference design, microarray technology blocking, 211
- Region of interest, scatterplot criteria, two-dimensional scatterplot ordering, 178–179
- Regression models, statistical/probabilistic models, 193–194
- Remote data source annotation, accessing data (bioconductor packages), 320–322
- Remote pathway data access, biological system models, 332–334
- Repeated measurement, replication contrasted, experimental design, 202–203
- Replication (experimental design), 202–209
 - general principle, 202–204
 - multilevel, resource allocation and, 208–209
 - sample size and power, 204–205
 - software packages, 205–208
- Resampling techniques. *See* Statistical resampling techniques
- Resource allocation, multilevel replication and, experimental design, 208–209
- R function, random variables distributions, 20
- Robustness, advanced clustering algorithms, 111
- Robustness identification, metabolic network modeling, 277–278
- R packages. *See also* Bioconductor packages; Software
 - Bayesian belief network, 272
 - statistical testing for large biological data analysis, 77–78, 81
 - supervised learning, software examples, 150–153
- Sammon’s mapping, multidimensional visualization analysis, 160–161
- Sample annotation, bioconductor packages, 318–319
- Sample heterogeneity, genomewide association studies, 294–295
- Sample size:
 - replication, experimental design, 204–205
 - statistical resampling techniques, 233–235
- Sampling, probability concepts, 8–9
- Sampling distribution, 46–54
 - asymptotic distribution, 47
 - definitions, 46
 - empirical distribution and bootstrapping, 48–54
 - mathematical derivations, 47
 - simulation studies, 47–48
- Scaling, data normalization, high-throughput biological data quality control, 66
- Scatterplot criteria, two-dimensional scatterplot ordering, 176–180
- Self-organizing maps, conventional clustering algorithms, 108–111
- Semiparametric statistical models, 188
- Sequence-based analysis, rare variants and, 302
- Sequences (similarity measures), 90, 95–99
 - global sequence alignment, 95–97
 - local sequence alignment, 97–99
- Shrunken centroid, classification and prediction methods, 134–135
- Significance analysis of microarrays (SAM), ad hoc tests, 76–77. *See also* Microarray gene expression experiments
- Similarity measures, 90–99
 - multidimensional objects, 90–95
 - distance-based, 90–92
 - matching-based, 92–95
 - sequences, 95–99
 - global sequence alignment, 95–97
 - local sequence alignment, 97–99
 - structural data, 99
- Simulation studies, sampling distribution, 47–48

- Single-nucleotide polymorphisms (SNP):
 genomewide association studies, 284,
 295–296
 multidimensional visualization analysis,
 157
- Small- n problem, large- p problem and,
 statistical bioinformatics, 3
- Software. *See also* Bioconductor packages;
 R packages
 Bayesian belief network, 272
 replication, experimental design, 205–208
 supervised learning, classification
 methods, 150–153
- Split sample method, statistical resampling
 techniques, 227–228
- Stability measures, clustering, quality
 assessment, 122
- Standard deviation, variance and, probability
 concepts and distributions, 17
- Statistical bioinformatics, 1–6
 high-dimensional biological data, 2–3
 integration problems, multiple,
 heterogeneous data, 3–4
 multiple-comparisons issue, 1–2
 noisy high-throughput data, 3
 small- n and large- p problem, 3
- Statistical network analysis, 249–282
 Bayesian belief network, 259–273
 applications, 269–270
 framework of, 259–264
 generally, 259
 hierarchical models, 270–271
 learning problems, 264–268
 software, 272
- Boolean network modeling, 250–259
 computation times, 253–254
 formulation of, 251
 generally, 250–251
 inference algorithms for, 252–253
 probabilistic networks, 254–258
- metabolic network modeling, 273–279
 concepts, 274–277
 control structures and robustness identi-
 fication, 277–278
 future prospects, 279
 generally, 273
 graphical representation, 273–274
 topology, 278–279
 overview, 249–250
- Statistical/probabilistic models, 187–189
 categories of, 188–189
 estimation methods, 189–191
 examples, 192–198
 ANOVA models, 194–196
 regression models, 193–194
 survival models, 196–198
 motivation, 187–188
 numerical algorithms, 191–192
- Statistical resampling techniques, 219–248
 bootstrap resampling for quantifying
 uncertainty, 236–238
 classification algorithms based on, 226
 conclusions, 240–247
 bootstrap methods for uncertainty
 quantification, 243
 Markov chain Monte Carlo methods,
 243–244
 prediction error assessment and model
 selection, 240–243
 prediction error estimation, pseudocode
 for, via cross-validation, 244–247
 feature selection, 225–226
 loss functions, 235–236
 lymphoma example, 226–227
 Markov chain Monte Carlo methods,
 238–240
 methods, 227–232
 bootstrap methods, 232–233
 cross-validation, 228–232
 split sample, 227–228
 model selection and performance
 assessment, 222–224
 overview, 219–220
 prediction error assessment, 221–222
 resampling framework, 224–225
 sample size, 233–235
 statistical resampling techniques, 224–225
 tests, statistical testing for large biological
 data analysis, 76
- Statistical testing and significance (for large
 biological data analysis), 71–88
 ad hoc tests, 76–77
 error controlling, 78–81
 multiple testing, 79–81
 p -values, 78–79
 fold change approach, 72–73
 nonparametric statistical tests, 75–76
 overview, 71–72
 parametric statistical tests, 73–75
 real data analysis, 81–87

- multiple samples, 85–87
- paired samples, 87
- tables, 81–84
- two samples, 84–85
- resampling-based tests, 76
- R packages, 77–78
- Steady-state probability distribution,
 - probabilistic Boolean networks, 257–258
- Structural data, similarity measures, 90, 99
- Structure analysis, genomewide association studies, 295
- Substrate-enzyme bipartite graph, metabolic network modeling, 274
- Substrate graph, metabolic network modeling, 273
- Supervised classification, unsupervised classification and, clustering, 99–100
- Supervised learning, 129–156. *See also*
 - Classification and prediction methods
 - classification and prediction methods, 132–140
 - artificial neural networks, 135–136
 - Bayes classifier, naive, 132
 - boosting methods, 138–139
 - classification by ensembles from random partitions (CERP), 139–140
 - classification trees, 136–137
 - k*-nearest-neighbor classifiers, 134
 - linear discriminant analysis, 133–134
 - logistic regression, 132–133
 - random forest, 137
 - shrunk centroid, 134–135
 - support vector machines, 138
 - classification methods, software examples, 150–153
 - classification methods comparison, 147–150
 - breast cancer data, 147–148
 - gastrointestinal bleeding data, 148–150
 - cross-validation, 144–145
 - ensemble voting methods, class prediction enhancement by, 145–147
 - feature selection and ranking, 140–144
 - comparison of, 143–144
 - variable importance ranking algorithms, 140–143
 - overview, 129–132
- Supervised principal-component analysis (SPCA), survival models, 198
- Support vector machines, classification and prediction methods, 138
- Survival models, statistical/probabilistic models, 196–198
- T-cell immune response data, real data analysis, 81–87
- Test set, training set and, resampling framework, 224–225, 227–228
- Time course experiments, experimental design, 215
- Topology, metabolic network modeling, 278–279
- Training set, test set and, resampling framework, 224–225, 227–228
- Transformations and potential ranking criteria, two-dimensional scatterplot ordering, 180–181
- Two-dimensional projections, 161–165
 - axis-parallel projection methods, 162–165
 - generally, 161
 - non-axis-parallel projection methods, 161–162
- Two-dimensional scatterplot ordering, 174–183
 - generally, 174–176
 - scatterplot criteria, 176–180
 - transformations and potential ranking criteria, 180–181
- Two-way clustering, one-way clustering versus, 102–104
- Uncertainty quantification, bootstrap resampling, 236–238, 243
- Uniform distribution:
 - continuous distributions, random variables distributions, 30–32
 - discrete probability distributions, random variables distributions, 23–24
 - one-dimensional histogram ordering, 172
- Uniformity, scatterplot criteria, two-dimensional scatterplot ordering, 179–180
- Unique values, one-dimensional histogram ordering, 173

- Univariate models, random variables distributions, 19
- Unsupervised classification, supervised classification and, clustering, 99–100
- Variable importance ranking algorithms, supervised learning, 140–143
- Variance, standard deviation and, probability concepts and distributions, 17
- v*-fold cross-validation methods, statistical resampling techniques, 228–230
- Visualization, genomic annotation, bioconductor packages, 323–324
- Wilcoxon rank-sum test, statistical testing for large biological data analysis, 75–76
- Wrapper methods, feature selection, statistical resampling techniques, 225–226

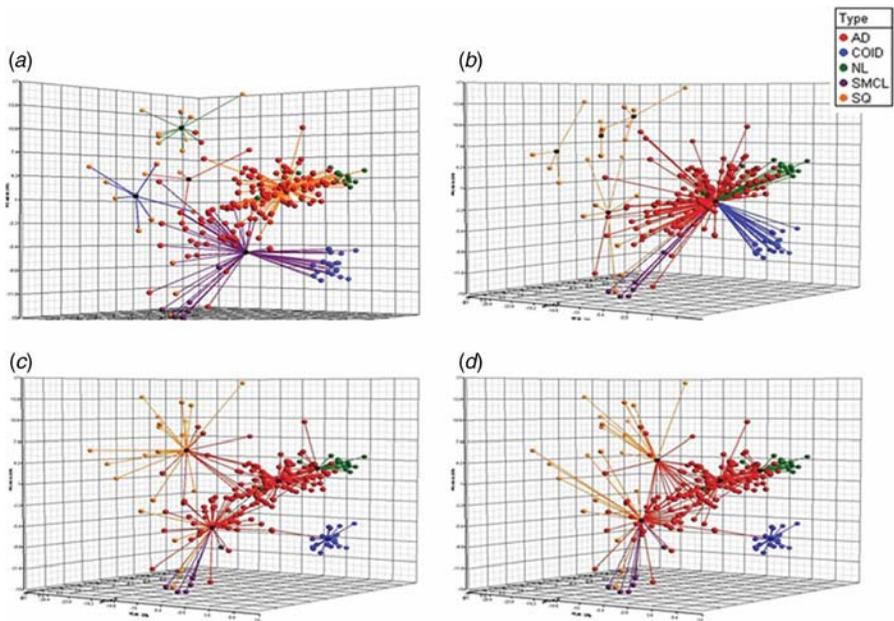


Figure 5.2 Demonstration of using different distance measures. The dataset used for this demonstration is the subset of genes from microarray study by Bhattacharjee et al. (2001). The clustering method used is the K -means algorithm. Different distance measures result in different clustering solutions: (a) Euclidean, (b) city block, (c) Pearson, and (d) cosine distances.

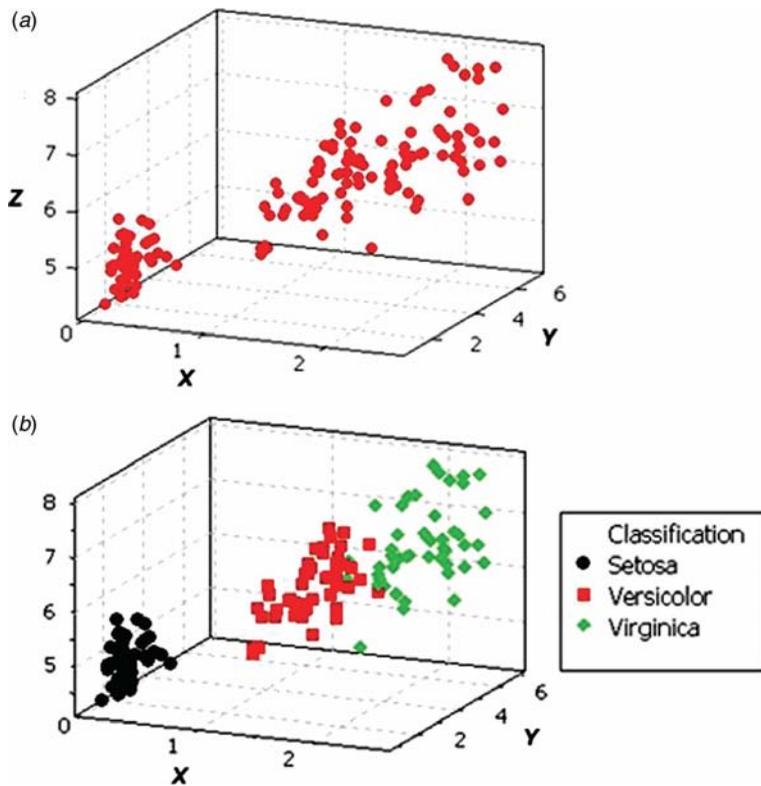


Figure 5.6 Unsupervised learning procedure of clustering: (a) before clustering (Iris dataset without class labels); (b) after clustering (three clusters: Setosa, Versicolor, and Virginica).

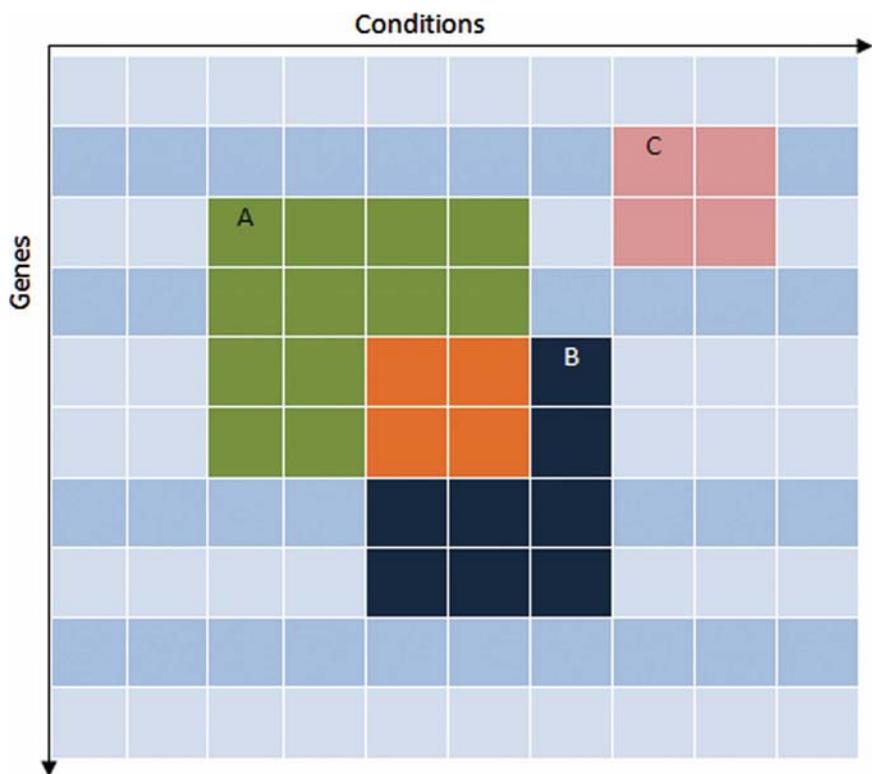


Figure 5.8 Example of biclusters in gene expression matrix.

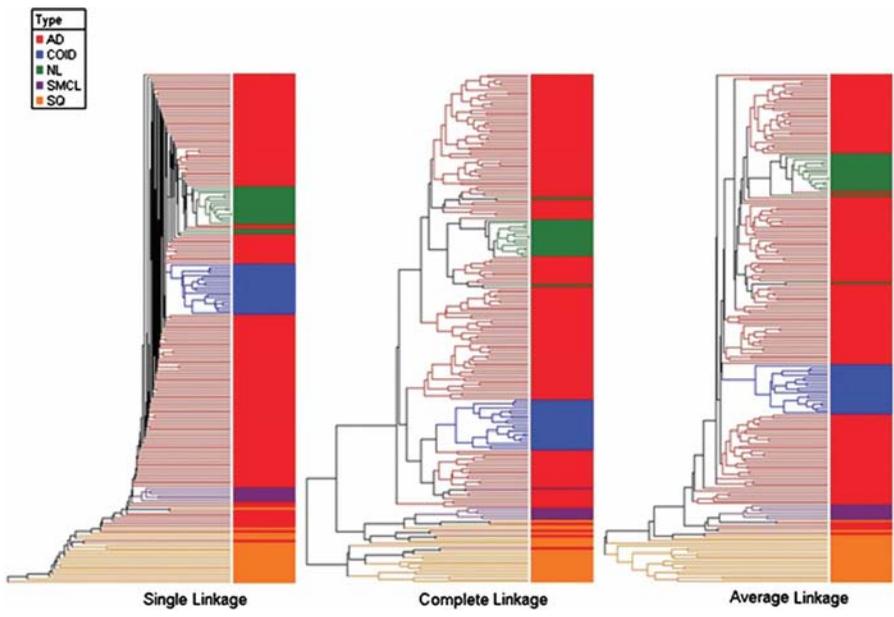


Figure 5.11 Agglomerative Hierarchical Clustering Algorithm with different linkage methods. The dataset used is the subset of the lung cancer dataset published by Bhattacharjee et al. (2001). The clustering method used is the agglomerative hierarchical clustering algorithms. The figure shows that different results generated due to different linkage methods.

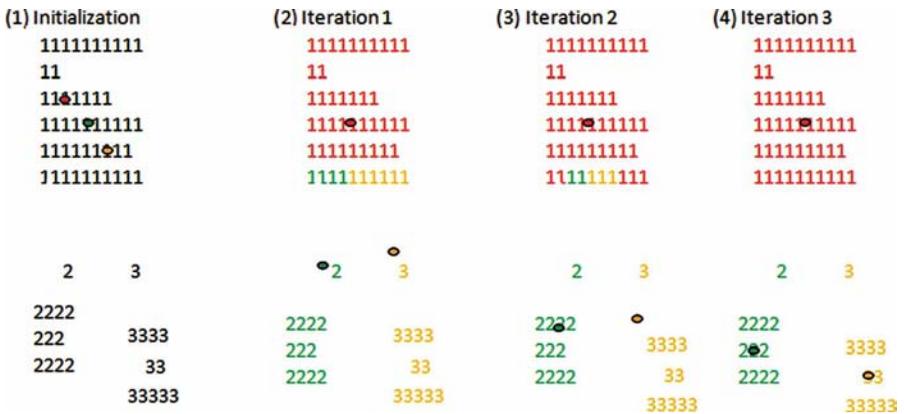


Figure 5.12 General procedure of *K*-means. The colored dots represent the centroids of clusters. To cluster the points into predefined three clusters: 1. The process is initialized by randomly selecting three objects as cluster centroids. 2. After the first iteration, points are assigned to the closest centroid, and cluster centroids are recalculated. 3. The iterative procedures of point reassignment and centroid update are repeated. 4. Centroids are stable; the termination condition is met.

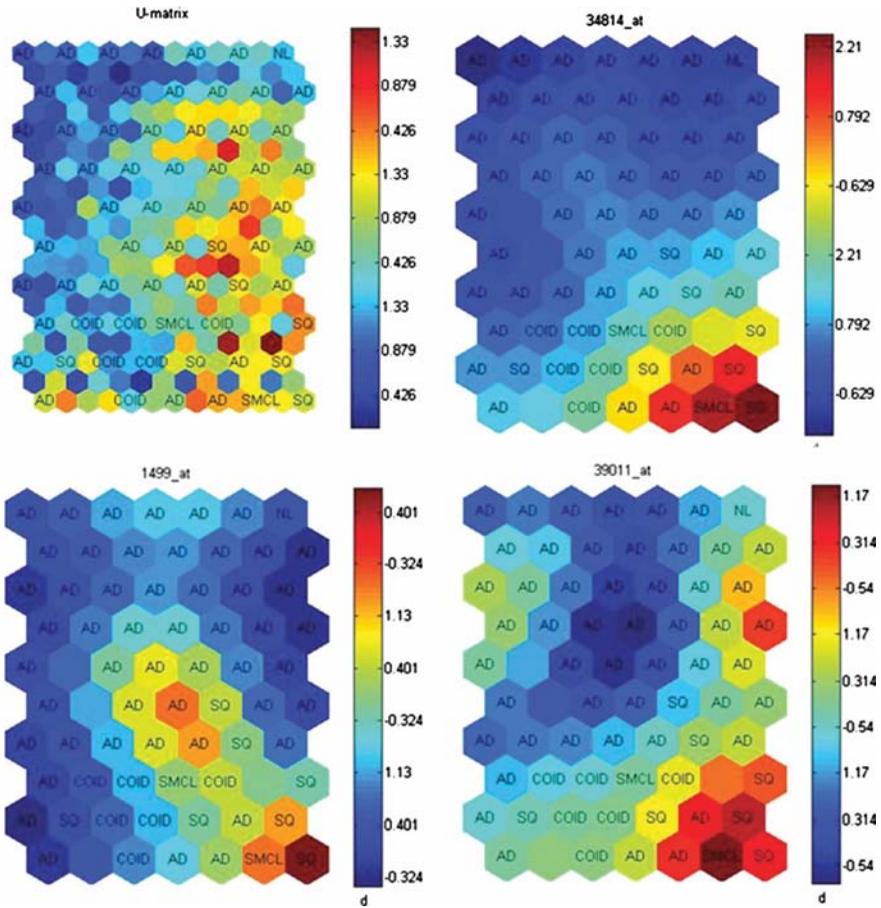


Figure 5.14 SOM Toolbox clustering of lung cancer samples using gene expression data for three representative genes measured by Bhattacharjee et al. (2001).

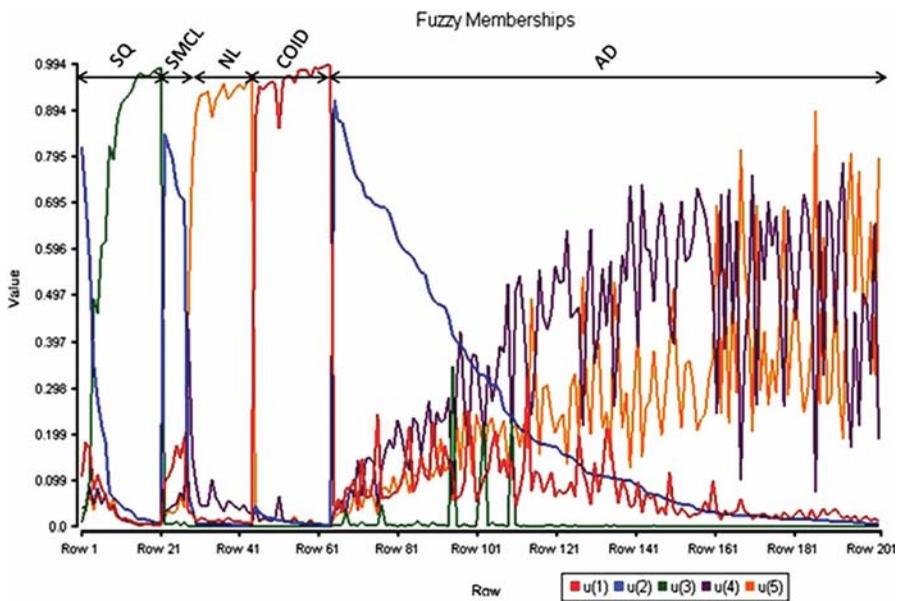


Figure 5.15 Membership values obtained from FCM clustering on lung cancer samples based on gene expression information. Samples were clustered in five groups.

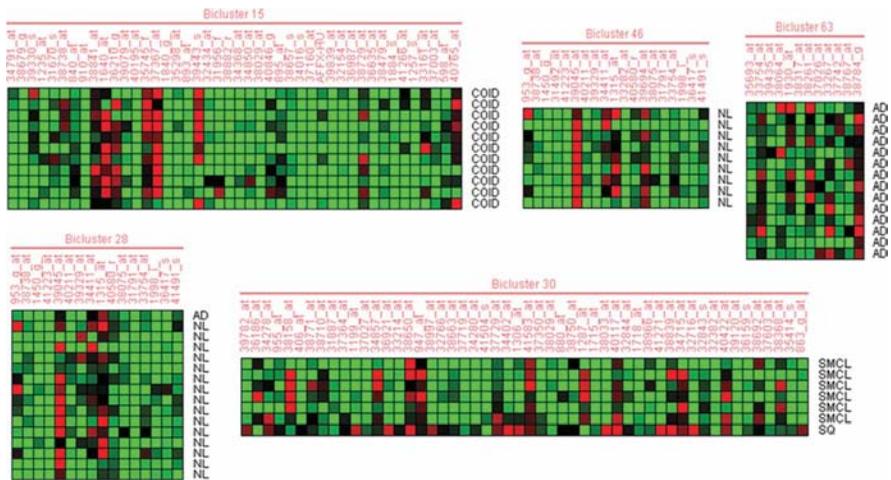


Figure 5.16 Result of SAMBA biclustering on lung cancer data. The different clusters show the sample clustering as well as the subsets of genes that are most relevant for the distinction of any given sample subgroup.

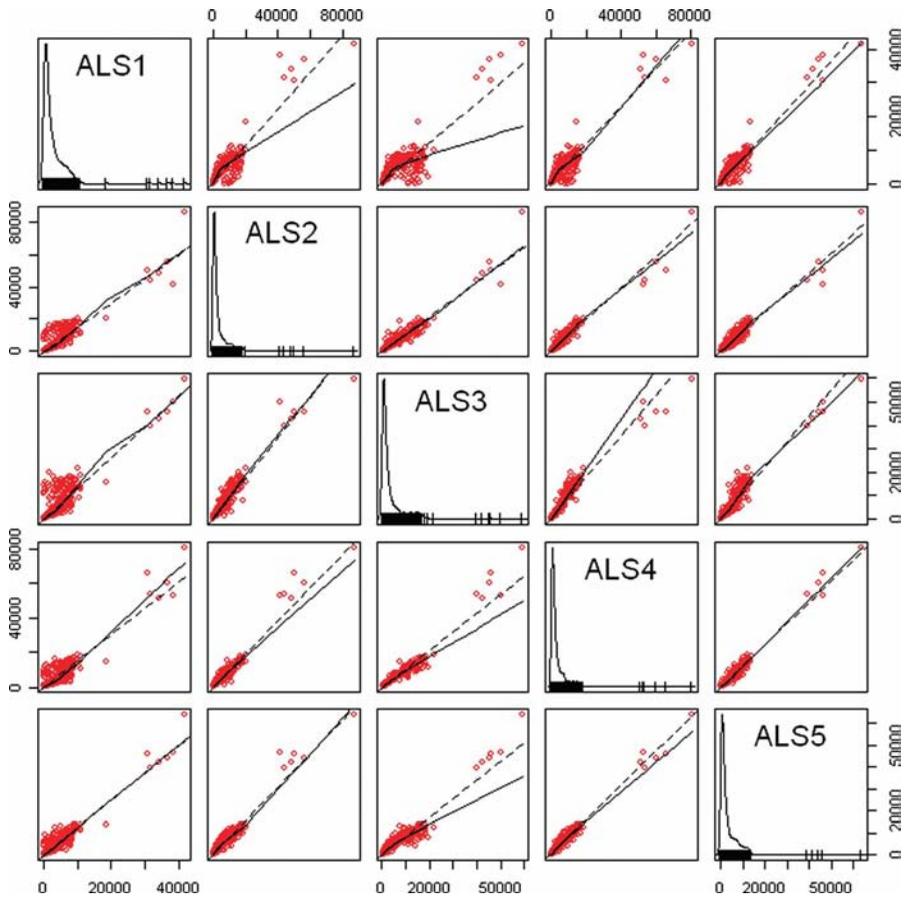


Figure 7.3 Scatterplot matrix visualization for first five dimensions of multidimensional dataset with 33 muscular dystrophy samples and 878 genes.

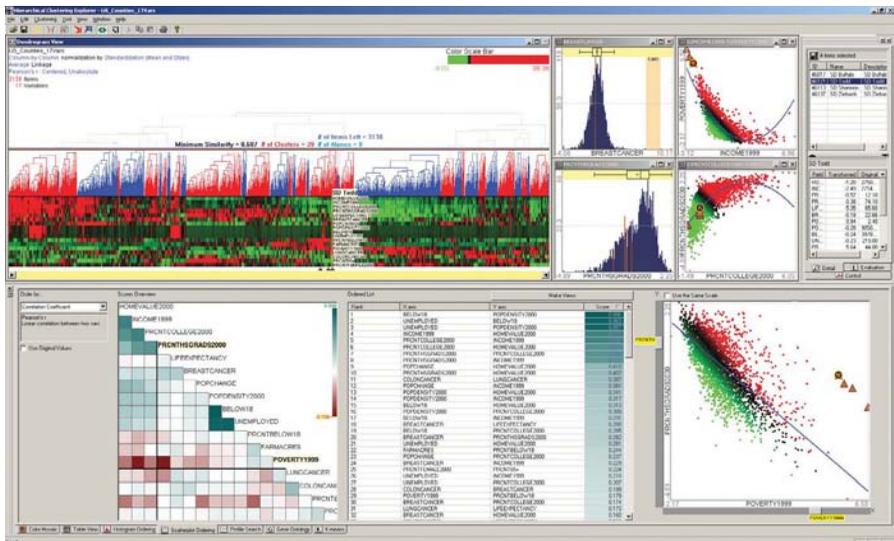


Figure 7.4 Rank-by-feature framework in HCE.

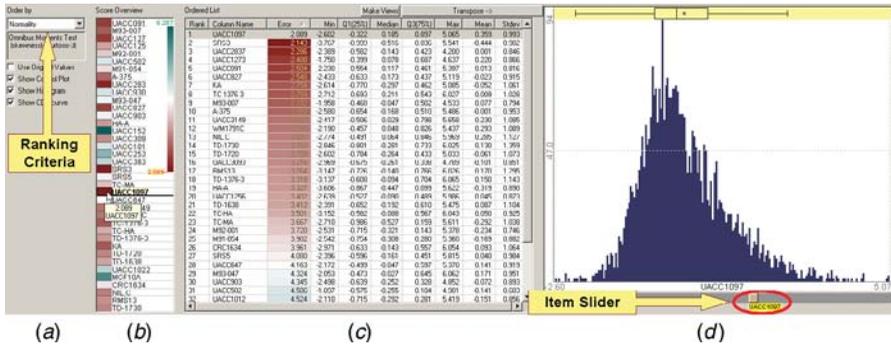


Figure 7.5 Rank-by-feature framework for 1D projections: histogram ordering.

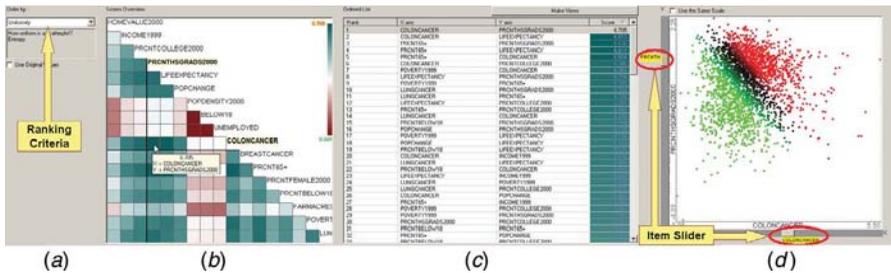


Figure 7.7 Rank-by-feature framework interface for scatterplots (2D): Scatterplot Ordering. All 2D scatterplots are ordered according to the current ordering criterion (a) in the ordered list (c). Users can select multiple scatterplots at the same time and generate separate scatterplot windows to compare them in a screen. The score overview (b) shows an overview of scores of all scatterplots. A mouseover event activates a cell in the score overview, highlights the corresponding item in the ordered list (c), and shows the corresponding scatterplot in the scatterplot browser (d) simultaneously. A click on a cell in the score overview selects the cell and the selection is fixed until another click event occurs in the score overview or another selection event occurs in other views. A selected scatterplot is shown in the scatterplot browser (d), where it is also easy to traverse scatterplot space by changing the x- or y-axis using item sliders on the horizontal or vertical axis. (The dataset shown is demographic- and health-related statistics for 3138 U.S. counties with 17 attributes.)

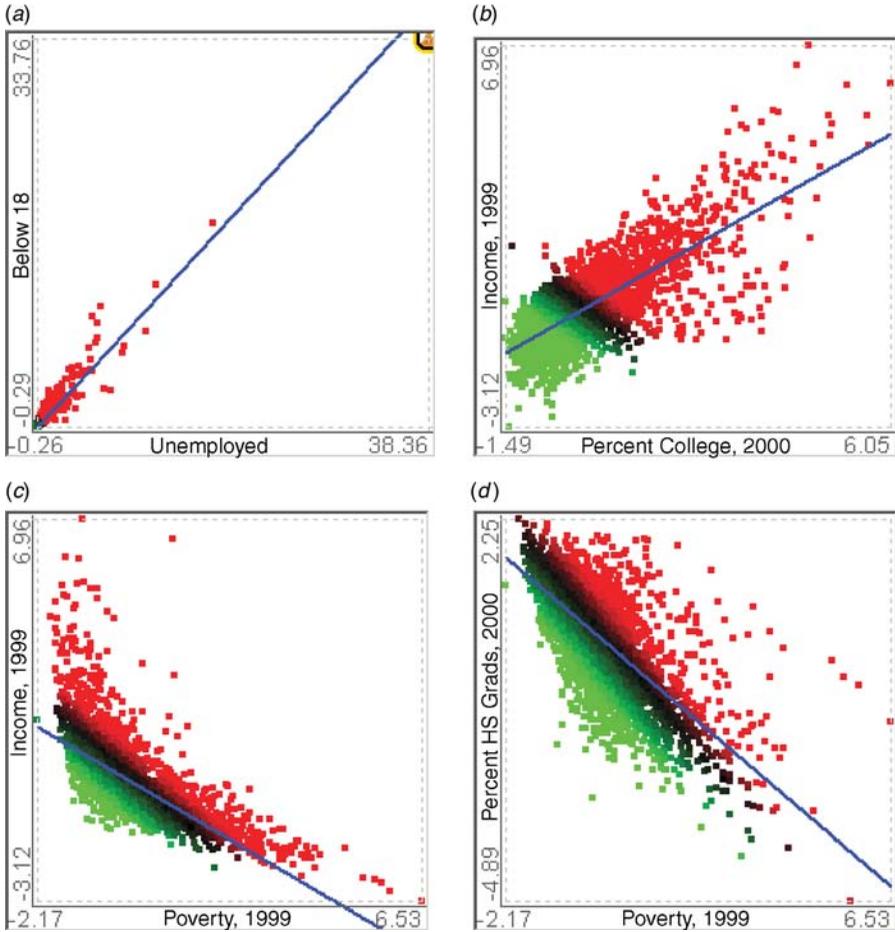


Figure 7.8 Four selected scatterplots ordered by correlation coefficient. The line of best fit is drawn as a blue line.

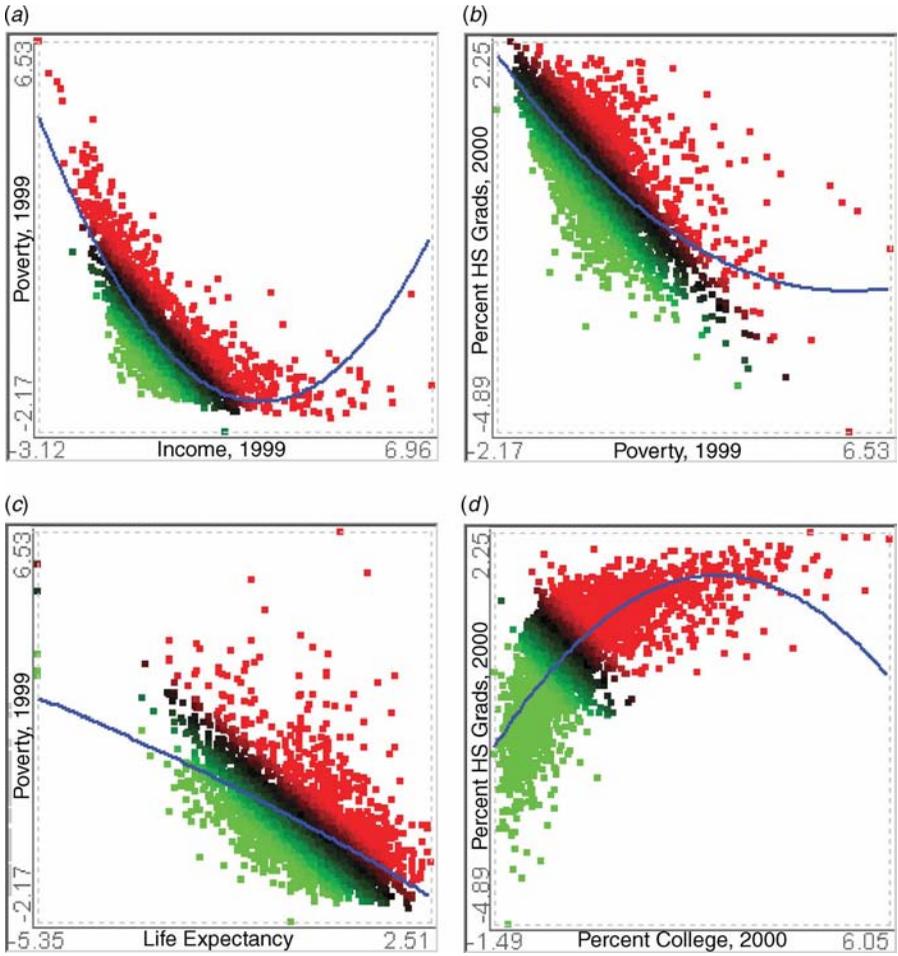


Figure 7.9 Quadraticity (coefficient of x_2 term). The regression curve is drawn as a blue parabola.

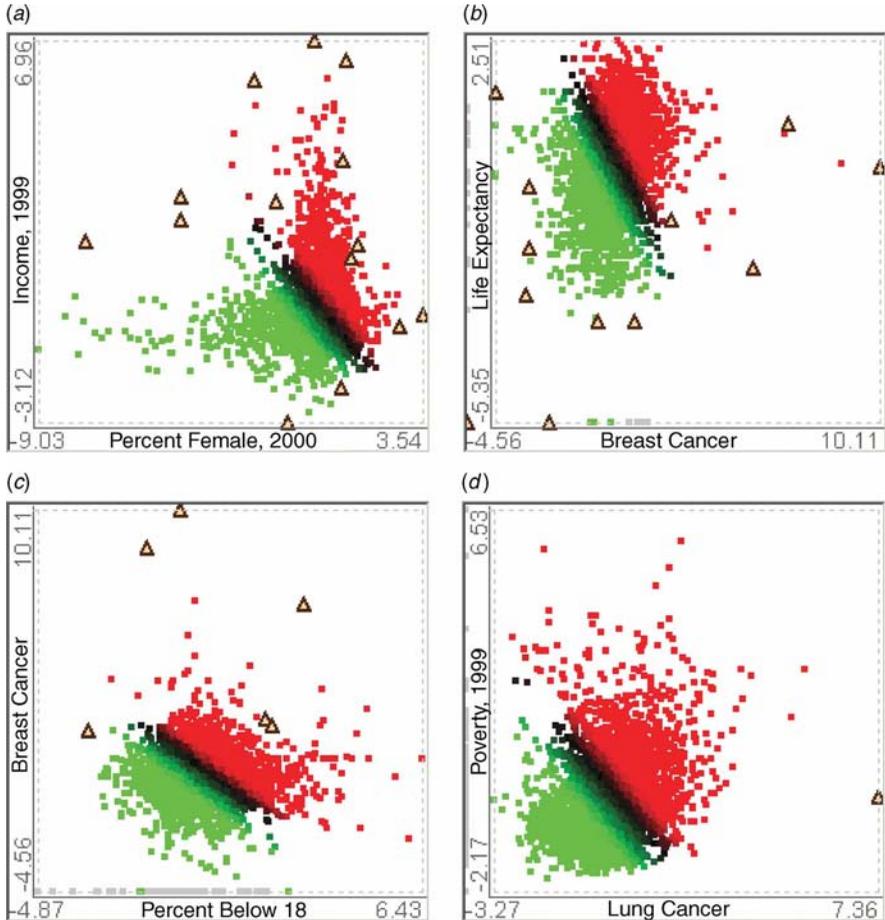


Figure 7.10 Number of outliers. Outliers whose LOF is greater than $(\text{minimum LOF} + \text{maximum LOF})/2$ are highlighted as triangles.

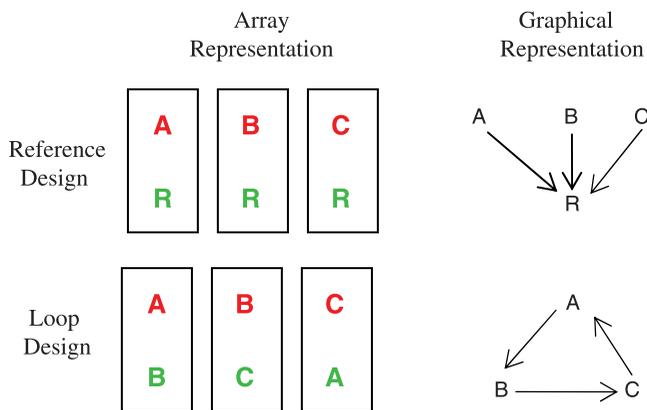


Figure 9.2 Array and graphical representations of reference design and loop designs for comparing three samples. In the array representation, the arrays are represented by rectangles. The samples used for hybridization on the array are indicated by letters in the rectangles. The dyes are represented by colors of the letters, green for Cy3 dye, red for Cy5 dye. In the graphical representation, the array is represented by arrow with head pointing to the Cy3 dye and tail pointing to the Cy5 dye. The samples are the nodes of the graphs.

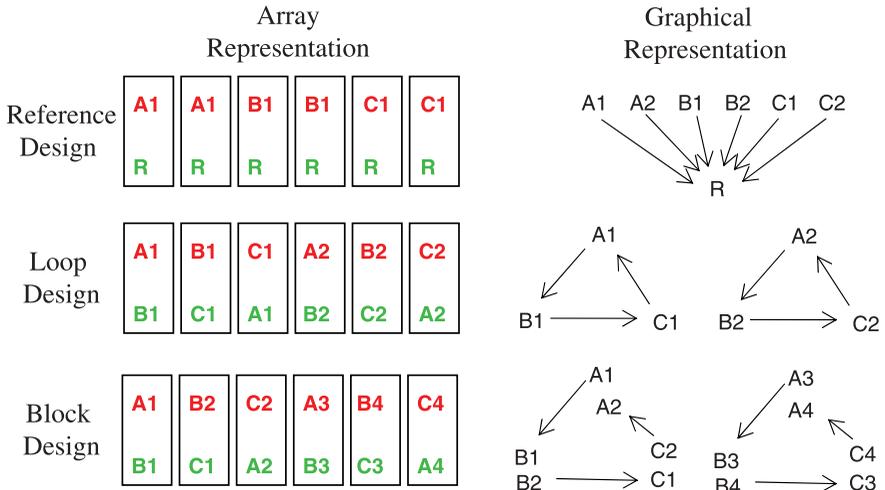


Figure 9.3 Array and graphical representations of designs with biological replicates. The three treatments are represented by three letters. Biological replicates within treatments are represented by the combination of letters and numbers. In the array presentation, arrays are represented by rectangles and dyes are represented by colors, green for Cy3 dye, red for Cy5 dye. In the graphical representation, arrays are represented by arrows with head pointing to the Cy3 dye and tail pointing to the Cy5 dye. For the same number of arrays, the balanced block design can accommodate four biological replicates while the other two designs can only accommodate two biological replicates.

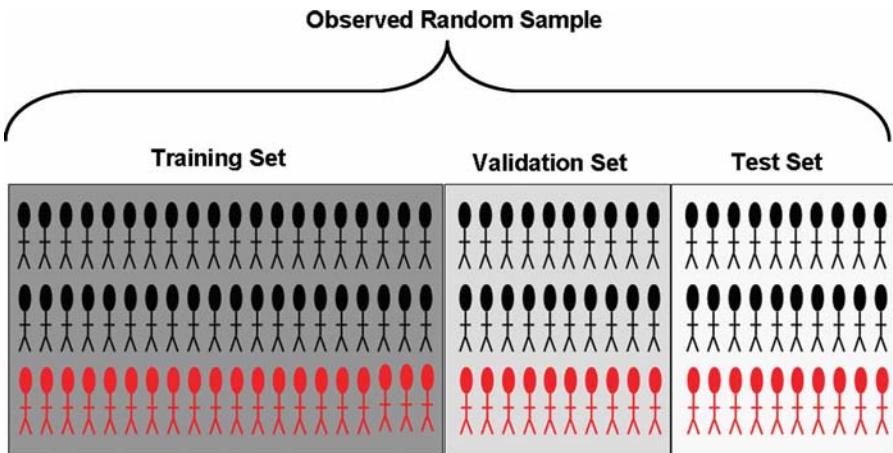


Figure 10.3 Large observed sample split into three sets: training for model building, validation for model selection, and test for assessment of prediction accuracy.

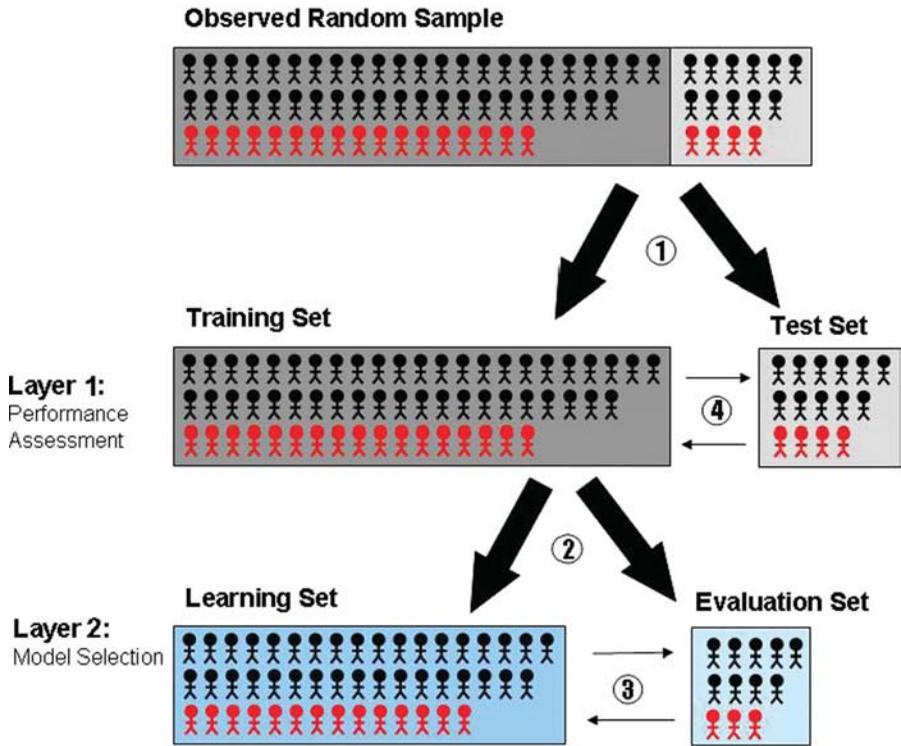


Figure 10.4 Flow chart for allocating data into training, learning, evaluation, and test sets.

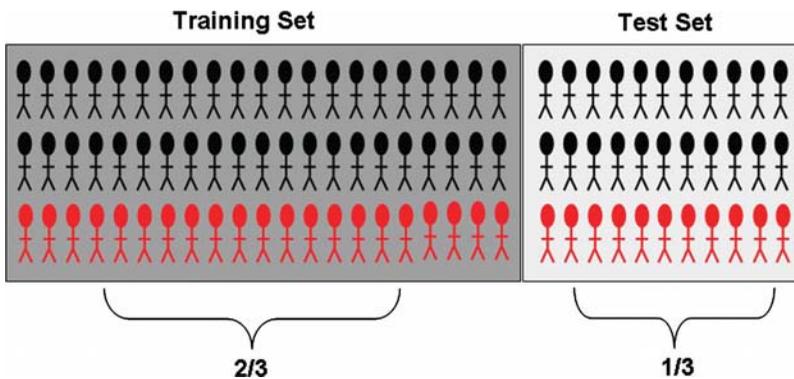


Figure 10.5 Depiction of split sample separated in a training ($\frac{2}{3}$ of observed data) and a test (remaining $\frac{1}{3}$) set.

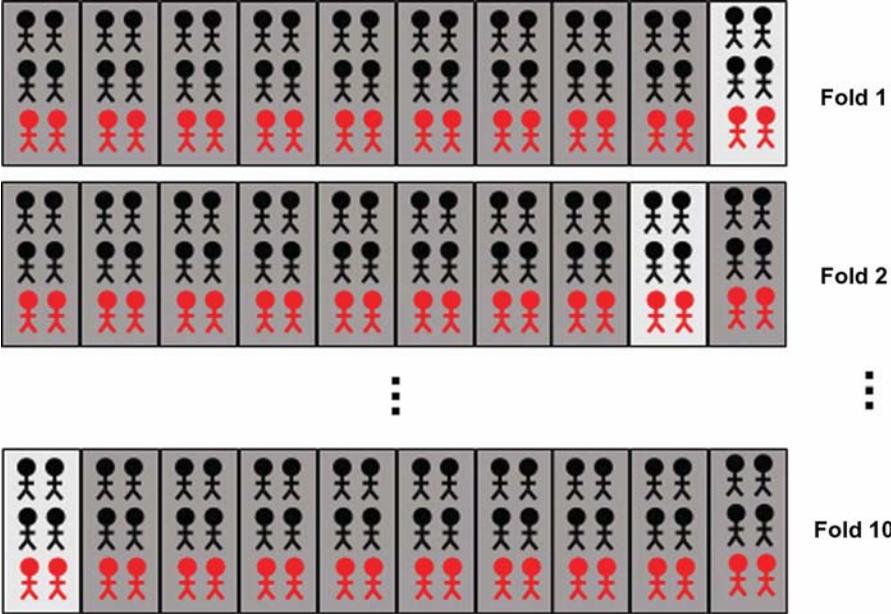


Figure 10.6 Depiction of training and test sets for the iterations of 10-fold CV.

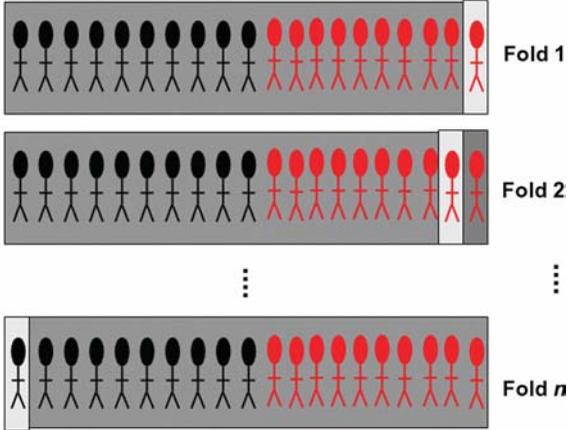


Figure 10.7 Depiction of training and test set splits for LOOCV.

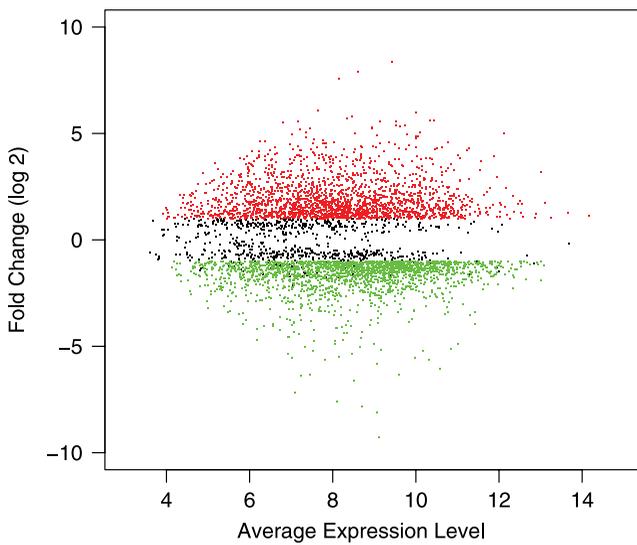


Figure 13.2 Mean-difference scatterplot of gene expression between differentiated and undifferentiated cells (log base 2). Each dot represents a gene: red dots highlight genes up-regulated in differentiated cells; green dots represent genes down-regulated in differentiated cells; black dots correspond to genes that are not differentially expressed.

