

Computational Systems Biology

Edited by

Jason McDermott
Ram Samudrala
Roger E. Bumgarner
Kristina Montgomery
Reneé Ireton

 Humana Press

METHODS IN MOLECULAR BIOLOGY™

Series Editor
John M. Walker
School of Life Sciences
University of Hertfordshire
Hatfield, Hertfordshire, AL10 9AB, UK

For other titles published in this series, go to
www.springer.com/series/7651

METHODS IN MOLECULAR BIOLOGY™

Computational Systems Biology

Edited by

Jason McDermott

Pacific Northwest National Laboratory, Richland, WA, USA

Ram Samudrala

University of Washington, Seattle, WA, USA

Roger E. Bumgarner

University of Washington, Seattle, WA, USA

Kristina Montgomery

University of Washington, Seattle, WA, USA

Reneé Ireton

Fred Hutchinson Cancer Research Center, Seattle, WA, USA

 **Humana Press**

Editors

Jason McDermott
Pacific Northwest National
Laboratory
Computational Biology
& Bioinformatics Group
Richland, WA, USA
jason.mcdermott@pnl.gov

Ram Samudrala
Department of Microbiology
University of Washington
960 Republican St., WA, USA
ram@compbio.washington.edu

Roger E. Bumgarner
Department of Microbiology
University of Washington
960 Republican St., WA, USA
rogerb@u.washington.edu

Kristina Montgomery
Department of Microbiology
University of Washington
960 Republican St., WA, USA
kmontgom@u.washington.edu

Reneé Ireton
Fred Hutchinson Cancer Research
Center
Public Health Sciences Div.
Molecular Diagnostics Program
WA, USA
rireton@fhcrc.org

ISSN 1064-3745

e-ISSN 1940-6029

ISBN 978-1-58829-905-5

e-ISBN 978-1-59745-243-4

DOI 10.1007/978-1-59745-243-4

Library of Congress Control Number: 2009920380

© Humana Press, a part of Springer Science+Business Media, LLC 2009

All rights reserved. This work may not be translated or copied in whole or in part without the written permission of the publisher (Humana Press, c/o Springer Science+Business Media, LLC, 233 Spring Street, New York, NY 10013, USA), except for brief excerpts in connection with reviews or scholarly analysis. Use in connection with any form of information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed is forbidden.

The use in this publication of trade names, trademarks, service marks, and similar terms, even if they are not identified as such, is not to be taken as an expression of opinion as to whether or not they are subject to proprietary rights.

Cover illustration: Figure 1 in Chapter1.

Printed on acid-free paper

springer.com

Preface

Computational systems biology is the term that we use to describe computational methods to identify, infer, model, and store relationships between the molecules, pathways, and cells (“systems”) involved in a living organism. Based on this definition, the field of computational systems biology has been in existence for some time. However, the recent confluence of high-throughput methodology for biological data gathering, genome-scale sequencing, and computational processing power has driven a reinvention and expansion of this field. The expansions include not only modeling of small metabolic (1–3) and signaling systems (2, 4) but also modeling of the relationships between biological components in very large systems, including whole cells and organisms (5–15). Generally, these models provide a general overview of one or more aspects of these systems and leave the determination of details to experimentalists focused on smaller subsystems. The promise of such approaches is that they will elucidate patterns, relationships, and general features, which are not evident from examining specific components or subsystems. These predictions are either interesting in and of themselves (e.g., the identification of an evolutionary pattern) or interesting and valuable to researchers working on a particular problem (e.g., highlight a previously unknown functional pathway).

Two events have occurred to bring the field of computational systems biology to the forefront. One is the advent of high-throughput methods that have generated large amounts of information about particular systems in the form of genetic studies, gene and protein expression analyses and metabolomics. With such tools, research to consider systems as a whole are being conceived, planned, and implemented experimentally on an ever more frequent and wider scale. The other event is the growth of computational processing power and tools. Methods to analyze large data sets of this kind are often computationally demanding and, as is the case in other areas, the field has benefited from continuing improvements in computational hardware and methods.

The field of computational biology is very much like a telescope with two sequential lenses: one lens represents the biological data and the other represents a computational and/or mathematical model of the data. Both lenses must be properly coordinated to yield an image that reflects biological reality. This means that the design parameters for both lenses must be designed in concert to create a system that yields a model of the organism, which provides both predictive and mechanistic information. The chapters in this book describe the construction of subcomponents of such a system. Computational systems biology is a rapidly evolving field and no single group of investigators has yet developed a complete system that integrates both data generation and data analysis in such a way so as to allow full and accurate modeling of any single biological organism. However, the field is rapidly moving in that direction. The chapters in this book represent a snapshot of the current methods being developed and used in the area of computational systems biology. Each method or database described within represents one or more steps on the path to a complete description of a biological system. How

these tools will evolve and ultimately be integrated is an area of intense research and interest. We hope that readers of this book will be motivated by the chapters within and become involved in this exciting area of research.

Organization of the Book

This volume is organized into five major parts: Network Components, Network Inference, Network Dynamics, Function and Evolutionary Systems Biology, and Computational Infrastructure for Systems Biology. Each section is described briefly below.

Part I – Network Components

This section focuses on methods to identify subcomponents of the complete networks. Ultimately, such subcomponents will need to be integrated with each other or used to inform other methods to arrive at a complete description of a biological system. This section begins with two methods for the prediction of transcription factor binding sites. In the first, Chapter 1, Mariño-Ramirez et al. describe a method for the prediction of transcription factor binding sites using a Gibbs sampling approach. In Chapter 2, Liu and Bader show how DNA-binding sites and specificity can be predicted using sophisticated structural analysis. Chapters 3–5 discuss methods to predict protein–protein interaction (PPI) networks, and Chapter 6 builds on predicted PPIs to identify potential regulatory interactions. Finally, Chapter 7 discusses the inherent modularity that is observed in biological networks with a focus on networks of PPIs.

Part II – Network Inference

This section focuses on methodologies to infer transcriptional networks on a genome-wide scale. In general, the methods described within focus on using either mRNA expression data or mRNA expression data coupled with expression quantitative trait locus (eQTL) data. To a large extent, method development in this area is driven primarily by the ubiquitous mRNA expression data that are available in the public domain or that are relatively easily generated within a single laboratory. These methods have been tremendously enabled by the development of array technology and hence predominately model mRNA levels (as that is the most ubiquitous data type). Chapters 8 and 9 present two methods for identifying and modeling transcriptional regulatory networks, while Chapter 10 focuses on inferring mRNA expression networks from eQTL data. Chapter 11 is a review of different methods for inferring and modeling large scale networks from expression and eQTL data.

Part III – Network Dynamics

Systems are not static entities. They change over time and in response to a variety of perturbations. Ultimately, computational systems biology will have to develop methods and corresponding data sets that allow one to infer and model the kinetics and

dynamics of reactions between all the chemical moieties in a cell. The chapters in this section focus on such methods. Chapter 12 discusses methods to infer both static co-expression networks and a finite-state Markov chain model for mimicking the dynamic behavior of a transcriptional network. Chapter 13 focuses on quantitative models of system behavior based on differential equations using biochemical control theory, whereas Chapter 14 focuses on the use of stochastic kinetic simulations. Both approaches have applications where one is superior to the other. At this point in time, it is not clear which methods will turn out to be most useful in dynamically modeling the largest number of biological systems. In general, this is likely the case for most of the technologies described in this book, so it is useful for readers to familiarize themselves with several concepts. Specifically, both Chapters 13 and 14 provide an excellent discussion of a variety of historical approaches to the dynamical modeling of biological systems and the relative merits and downsides to each. Chapter 15 provides an excellent introduction to considerations for the interplay between experimental design and dynamic modeling using lambda phage as an example system. The methods and considerations described within are generally applicable to other biological systems and highlight the importance of integrating the direction of wet bench work and computational modeling to more rapidly refine the models.

Part IV – Function and Evolutionary Systems Biology

The ultimate representation of the function of a given biological moiety is a complete description of all the reactions in which it participates and the relative rates of said reactions. At present, we are quite distant from this goal for most biological molecules or systems. However, we are able to use computational methods to predict the most likely functions of a given protein and even predict which portions and specific sequences of the protein contribute most to that function. This section is focused on methods used to infer protein function and on the relationships between function and evolution.

Ultimately, the reason to study and research “systems” biology is to understand biological function at a given hierarchical level (be it a single catalytic site or entire pathways). The interplay between the detailed atomic study of function and the large-scale study of systems will enable us to achieve this goal. This section contains chapters that address the interdependence of these two aspects: individual algorithms or techniques to understand the functional role of atoms or residues in single molecules (e.g., proteins), which in turn are extrapolated to understand their greater role in terms of biological or organismal function. Conversely and complementarily, the role of larger systems and their influence on single molecules is also explored. Together, all these chapters illustrate the strong dependence between single molecules and entire pathways or systems.

Part V – Computational Infrastructure for Systems Biology

To represent and organize the large amounts of experimental data and software tools, database frameworks must be created and made available to the larger biological

community. This chapter focuses on computational methods and databases as well as data representations necessary to both integrate and export systems biology information to an end user. The user may be the biologist searching for their gene of interest or they may be the bioinformatician looking for trends in protein function among higher eukaryotes. Several groups are working on this extremely difficult task of providing semantic meaning to the large amounts of underlying biological data collected from single and high-throughput experiments, as well as computational predictions. (As a parenthetical comment, this is a significantly much harder problem than one faced by Internet search engines such as a Google, which at this point do not provide any semantic meaning to a query.) We present only a few such examples in this section (and in this book). One primary focus is on the Bioverse framework, database, and web application, which was developed by the editors of this book. However, we also describe the Biozon as well as the SEBIN and CABIN frameworks. The abstract representations required to model biological systems are still in fruition, and a complement of many tools, technologies, databases, and algorithms will have to be integrated in the future as our knowledge expands.

Acknowledgments

It goes without saying that this volume would not have been possible without the efforts of countless biologists who provided the raw data that enable modeling of biological systems. We first and foremost thank all these researchers who provide the raw data for all the computational modeling that enables the field of “computational” systems biology. We also thank all the researchers who investigate these problems using sophisticated modeling techniques, particularly those described in this volume. We thank two particular editors of this volume, René Iretton and Kristina Montgomery, who have dealt with the thankless job of undertaking the proofreading the chapters in this book and associated bureaucratic requirements. We finally thank the McDermott, Bumgarner, and Samudrala groups for their critical comments as this volume was being prepared, as well as our respective families for their patience. The administrative aspects of this work were in part supported by the NSF CAREER award to Dr. Ram Samudrala.

References

1. Ishii N, Robert, M, Nakayama, Y, Kanai, A and Tomita, M. Toward large-scale modeling of the microbial cell for computer simulation. *J Biotechnol* 2004;113(1-3):281–94.
2. Ekins S, Nikolsky, Y, Bugrim, A, Kirillov, E and Nikolskaya, T. Pathway mapping tools for analysis of high content data. *Methods Mol Biol* 2006;356:319–50.
3. Lafaye A, Junot, C, Pereira, Y, et al. Combined proteome and metabolite-profiling analyses reveal surprising insights into yeast sulfur metabolism. *J Biol Chem* 2005;280(26):24723–30.
4. Stevenson-Paulik J, Chiou, ST, Frederick, JP, et al. Inositol phosphate metabolomics: merging genetic perturbation with modernized radiolabeling methods. *Methods* 2006;39(2):112–21.
5. Ideker T, Thorsson, V, Ranish, JA, et al. Integrated genomic and proteomic analyses of a systematically perturbed metabolic network. *Science* 2001;292(5518):929–34.
6. Pe'er D, Regev, A, Elidan, G and Friedman, N. Inferring subnetworks from perturbed expression profiles. *Bioinformatics* 2001;17 Suppl 1:S215–24.

7. Pilpel Y, Sudarsanam, P and Church, GM. Identifying regulatory networks by combinatorial analysis of promoter elements. *Nat Genet* 2001;29(2):153–9.
8. Ideker T, Ozier, O, Schwikowski, B and Siegel, AF. Discovering regulatory and signalling circuits in molecular interaction networks. *Bioinformatics* 2002;18 Suppl 1:S233–40.
9. Kelley BP, Sharan, R, Karp, RM, et al. Conserved pathways within bacteria and yeast as revealed by global protein network alignment. *Proc Natl Acad Sci U S A* 2003;100(20):11394–9.
10. Shannon P, Markiel, A, Ozier, O, et al. Cytoscape: a software environment for integrated models of biomolecular interaction networks. *Genome Res* 2003;13(11):2498–504.
11. Ideker T. A systems approach to discovering signaling and regulatory pathways—or, how to digest large interaction networks into relevant pieces. *Adv Exp Med Biol* 2004;547:21–30.
12. Schadt EE, Monks, SA, Drake, TA, et al. Genetics of gene expression surveyed in maize, mouse and man. *Nature* 2003;422(6929):297–302.
13. Schadt EE and Lum, PY. Reverse engineering gene networks to identify key drivers of complex disease phenotypes. *J Lipid Res* 2006.
14. McDermott J and Samudrala R. Bioverse: Functional, structural and contextual annotation of proteins and proteomes. *Nucleic Acids Res* 2003;31(13):3736–7.
15. McDermott J, Bumgarner, R and Samudrala, R. Functional annotation from predicted protein interaction networks. *Bioinformatics* 2005;21(15):3217–26.

Contents

<i>Preface</i>	<i>v</i>
<i>Contributors</i>	<i>xiii</i>
<i>Color Plates</i>	<i>xvii</i>

PART I: NETWORK COMPONENTS

1. Identification of <i>cis</i> -Regulatory Elements in Gene Co-expression Networks Using A-GLAM	3
<i>Leonardo Mariño-Ramírez, Kannan Tharakaraman, Olivier Bodenreider, John Spouge, and David Landsman</i>	
2. Structure-Based <i>Ab Initio</i> Prediction of Transcription Factor–Binding Sites	23
<i>L. Angela Liu and Joel S. Bader</i>	
3. Inferring Protein–Protein Interactions from Multiple Protein Domain Combinations	43
<i>Simon P. Kanaan, Chengbang Huang, Stefan Wuchty, Danny Z. Chen, and Jesús A. Izaguirre</i>	
4. Prediction of Protein–Protein Interactions: A Study of the Co-evolution Model	61
<i>Itai Sharon, Jason V. Davis, and Golan Yona</i>	
5. Computational Reconstruction of Protein–Protein Interaction Networks: Algorithms and Issues	89
<i>Eric Franzosa, Bolan Linghu, and Yu Xia</i>	
6. Prediction and Integration of Regulatory and Protein–Protein Interactions.	101
<i>Duangdao Wichadakul, Jason McDermott, and Ram Samudrala</i>	
7. Detecting Hierarchical Modularity in Biological Networks.	145
<i>Erzsébet Ravasz</i>	

PART II: NETWORK INFERENCE

8. Methods to Reconstruct and Compare Transcriptional Regulatory Networks	163
<i>M. Madan Babu, Benjamin Lang, and L. Aravind</i>	
9. Learning Global Models of Transcriptional Regulatory Networks from Data	181
<i>Aviv Madar and Richard Bonneau</i>	
10. Inferring Molecular Interactions Pathways from eQTL Data	211
<i>Imran Rashid, Jason McDermott, and Ram Samudrala</i>	
11. Methods for the Inference of Biological Pathways and Networks	225
<i>Roger E. Bumgarner and Ka Yee Yeung</i>	

PART III: NETWORK DYNAMICS

12. Exploring Pathways from Gene Co-expression to Network Dynamics	249
<i>Huai Li, Yu Sun, and Ming Zhan</i>	
13. Network Dynamics	269
<i>Herbert M. Sauro</i>	

14.	Kinetic Modeling of Biological Systems	311
	<i>Haluk Resat, Linda Petzold, and Michel F. Pettigrew</i>	
15.	Guidance for Data Collection and Computational Modelling of Regulatory Networks	337
	<i>Adam Christopher Palmer, and Keith Edward Shearwin</i>	
PART IV: FUNCTION AND EVOLUTIONARY SYSTEMS BIOLOGY		
16.	A Maximum Likelihood Method for Reconstruction of the Evolution of Eukaryotic Gene Structure	357
	<i>Liran Carmel, Igor B. Rogozin, Yuri I. Wolf, and Eugene V. Koonin</i>	
17.	Enzyme Function Prediction with Interpretable Models	373
	<i>Umar Syed and Golan Yona</i>	
18.	Using Evolutionary Information to Find Specificity-Determining and Co-evolving Residues	421
	<i>Grigory Kolesov and Leonid A. Mirny</i>	
19.	Connecting Protein Interaction Data, Mutations, and Disease Using Bioinformatics	449
	<i>Jake Y. Chen, Eunseog Youn, and Sean D. Mooney</i>	
20.	Effects of Functional Bias on Supervised Learning of a Gene Network Model	463
	<i>Insuk Lee and Edward M. Marcotte</i>	
PART V: COMPUTATIONAL INFRASTRUCTURE FOR SYSTEMS BIOLOGY		
21.	Comparing Algorithms for Clustering of Expression Data: How to Assess Gene Clusters	479
	<i>Golan Yona, William Dirks, and Shafquat Rahman</i>	
22.	The Bioverse API and Web Application	511
	<i>Michal Guerquin, Jason McDermott, Zach Frazier, and Ram Samudrala</i>	
23.	Computational Representation of Biological Systems	535
	<i>Zach Frazier, Jason McDermott, Michal Guerquin, and Ram Samudrala</i>	
24.	Biological Network Inference and Analysis Using SEBINI and CABIN	551
	<i>Ronald Taylor and Mudita Singhal</i>	
	<i>Index</i>	577

Contributors

- L. ARAVIND • *National Center for Biotechnology Information, National Library of Medicine, National Institutes of Health, Bethesda, MD, USA*
- M. MADAN BABU • *MRC Laboratory of Molecular Biology, Cambridge, UK*
- JOEL S. BADER • *Department of Biomedical Engineering and High-Throughput Biology Center, Johns Hopkins University, Baltimore, MD, USA*
- OLIVIER BODENREIDER • *National Library of Medicine, National Institutes of Health, Bethesda, MD, USA*
- RICHARD BONNEAU • *Department of Computer Science, Courant Institute of Mathematical Sciences, New York University, New York, NY, USA*
- ROGER E. BUMGARNER • *Department of Microbiology, University of Washington, Seattle, WA, USA*
- LIRAN CARMEL • *National Center for Biotechnology Information, National Library of Medicine, National Institutes of Health, Bethesda, MD, USA*
- DANNY Z. CHEN • *Department of Computer Science, University of Notre Dame, Notre Dame, IN, USA*
- JAKE Y. CHEN • *Informatics and Technology Complex (IT), Indiana University School of Informatics, IUPUI, Indianapolis, IN, USA*
- JASON V. DAVIS • *Department of Computer Science, University of Texas at Austin, Austin, TX, USA*
- WILLIAM DIRKS • *Center for Integrative Genomics, University of California, Berkeley, Berkeley, CA, USA*
- ERIC FRANOSA • *Bioinformatics Program, Boston University, Boston, MA, USA*
- ZACH FRAZIER • *Department of Microbiology, University of Washington, Seattle, WA, USA*
- MICHAL GUERQUIN • *Department of Microbiology, University of Washington, Seattle, WA, USA*
- CHENGBANG HUANG • *Department of Computer Science, University of Notre Dame, Notre Dame, IN, USA*
- JESUS IZAGUIRRE • *Department of Computer Science, University of Notre Dame, Notre Dame, IN, USA*
- SIMON P. KANAAN • *Department of Computer Science, University of Notre Dame, Notre Dame, IN, USA*
- GRIGORY KOLESOV • *Harvard-MIT Division of Health Sciences and Technology, Massachusetts Institute of Technology, Cambridge, MA, USA*
- EUGENE V. KOONIN • *National Center for Biotechnology Information, National Library of Medicine, National Institutes of Health, Bethesda, MD, USA*
- DAVID LANDSMAN • *Computational Biology Branch, National Center for Biotechnology Information, National Institutes of Health, Bethesda, MD, USA*
- BENJAMIN LANG • *MRC Laboratory of Molecular Biology, Cambridge, UK*

- INSUK LEE • *Center of Systems and Synthetic Biology, Institute for Molecular Biology, University of Texas at Austin, Austin, TX, USA*
- HUAI LI • *Bioinformatics Unit, Branch of Research Resources, National Institute on Aging, National Institutes of Health, Baltimore, MD, USA*
- BOLAN LINGHU • *Bioinformatics Program, Boston University, Boston, MA, USA*
- L. ANGELA LIU • *Department of Biomedical Engineering and Institute for Multiscale Modeling of Biological Interactions, John Hopkins University, Baltimore, MD, USA*
- AVIV MADAR • *Center for Comparative Functional Genomics, New York University, New York, NY, USA*
- EDWARD M. MARCOTTE • *Center for Systems and Synthetic Biology and Department of Chemistry and Biochemistry, Institute for Molecular Biology, University of Texas at Austin, Austin, TX, USA*
- LEONARDO MARINO-RAMIREZ • *Computational Biology Branch, National Center for Biotechnology Information, National Institutes of Health, Bethesda, MD, USA*
- JASON MCDERMOTT • *Computational Biology and Bioinformatics, Pacific Northwest National Laboratory, Richland, WA, USA*
- LEONID A. MIRNY • *Harvard-MIT Division of Health Sciences and Technology, Massachusetts Institute of Technology, Cambridge, MA, USA*
- SEAN D. MOONEY • *Department of Medical and Molecular Genetics, Center for Computational Biology and Bioinformatics, IUPUI, Indianapolis, IN, USA*
- ADAM CHRISTOPHER PALMER • *School of Molecular and Biomedical Science, The University of Adelaide, Adelaide, Australia*
- MICHEL F. PETTIGREW • *ScienceOps, Bothell, WA, USA*
- LINDA PETZOLD • *Department of Computer Science, University of California Santa Barbara, Santa Barbara, CA, USA*
- SHAFQUAT RAHMAN • *Mathworks Inc., Natick, MA, USA*
- IMRAN RASHID • *Department of Computer Science and Engineering, University of Washington, Seattle, WA, USA*
- ERZSÉBET RAVASZ REGAN • *Beth Israel Deaconess Medical Center, Harvard Medical School, Boston, MA, USA*
- HALUK RESAT • *Pacific Northwest National Laboratory, Richland, WA, USA*
- IGOR B. ROGOZIN • *National Center for Biotechnology Information, National Library of Medicine, National Institutes of Health, Bethesda, MD, USA*
- RAM SAMUDRALA • *Department of Microbiology, University of Washington, Seattle, WA, USA*
- HERBERT M. SAURO • *Department of Bioengineering, University of Washington, Seattle, WA, USA*
- ITAI SHARON • *Department of Computer Science, Technion - Israel Institute of Technology, Haifa, Israel*
- KEITH EDWARD SHEARWIN • *School of Molecular and Biomedical Science, The University of Adelaide, Adelaide, Australia*
- MUDITA SINGHAL • *Computational Biology and Bioinformatics Group, Computational and Informational Sciences Directorate, Pacific Northwest National Laboratory, Richland, WA, USA*
- JOHN SPOUGE • *Computational Biology Branch, National Center for Biotechnology Information, National Institutes of Health, Bethesda, MD, USA*

- YU SUN • *Bioinformatics Unit, Branch of Research Resources, National Institute on Aging, National Institutes of Health, Baltimore, MD, USA*
- UMAR SYED • *Department of Computer Science, Princeton University, Princeton, NJ, USA*
- RONALD TAYLOR • *Computational Biology and Bioinformatics Group, Computational and Informational Sciences Directorate, Pacific Northwest National Laboratory, Richland, WA, USA*
- KANNAN THARAKARAMAN • *Computational Biology Branch, National Center for Biotechnology Information, National Institutes of Health, Bethesda, MD, USA*
- DUANGDAO WICHADAKUL • *BIOTEC, Pathumthani, Thailand*
- YURI I. WOLF • *National Center for Biotechnology Information, National Library of Medicine, National Institutes of Health, Bethesda, MD, USA*
- STEFAN WUCHTY • *Northwestern Institute of Complexity, Northwestern University, Evanston, IL, USA*
- YU XIA • *Bioinformatics Program and Department of Chemistry, Boston University, Boston, MA, USA*
- KA YEE YEUNG • *Department of Microbiology, University of Washington, Seattle, WA, USA*
- GOLAN YONA • *Department of Biological Statistics and Computational Biology, Cornell University, Ithaca, NY, USA; Department of Computer Science, Technion – Israel Institute of Technology, Haifa, Israel*
- EUNSEOG YOUN • *Department of Computer Science, Texas Tech University, Lubbock, TX, USA*
- MING ZHAN • *Bioinformatics Unit, Branch of Research Resources, National Institute on Aging, National Institutes of Health, Baltimore, MD, USA*

Color Plates

- Color Plate 1 Uncovering the underlying modularity of a complex network. (a) Topological overlap illustrated on a small hypothetical network. On each link, we indicate the topological overlap for the connected nodes; and in parentheses next to each node, we indicate the node's clustering coefficient. (b) The topological overlap matrix corresponding to the small network shown in (a). The *rows and columns* of the matrix were reordered by the application of an average linkage clustering method to its elements, allowing us to identify and place close to each other those nodes that have high topological overlap. The *color code* denotes the degree of topological overlap between the nodes. The associated tree reflects the three distinct modules built into the model, as well as the fact that the EFG and HIJK modules are closer to each other in the topological sense than to the ABC module (Chapter 7, Fig. 3; *see* discussion on p. 151).
- Color Plate 2 Topological modules in the *Escherichia coli* metabolism: the topologic overlap matrix, together with the corresponding hierarchical tree (*top and right*) that quantifies the relation between the different modules. The branches of the tree are *color-coded* to reflect the predominant biochemical classification of their substrates. The *color code* of the matrix denotes the degree of topological overlap shown in the matrix. The large-scale functional map of the metabolism, as suggested by the hierarchical tree, is also shown (*bottom*) (Chapter 7, Fig. 5; *see* discussion on p. 153).
- Color Plate 3 Enlarged view of the substrate module of pyrimidine metabolism, along with a detailed diagram of the metabolic reactions that surround and incorporate it. The *colored boxes* in the background denote the first two levels of the three levels of nested modularity suggested by the hierarchical tree. *Red-outlined boxes* denote the substrates directly appearing in the reduced metabolism and thus on the tree (Chapter 7, Fig. 6; *see* discussion on p. 154 and full caption on p. 155).
- Color Plate 4 Structural localization of putative SDRs and CERs in two-component system domains. (a) RR Spo0F (*red-brown ribbon*) bound to structural analog of the DD in Spo0B protein. The conserved His is shown in *purple*, the conserved Asp in RR in *magenta*. SDRs and CERs are shown in *yellow* or, when located on the $\alpha 4$ helix, in *white* (PDB entry 1F51). (b) The non-catalytic conformation of HK homodimer. ADP is shown as a *purple* wireframe, the phosphate-accepting conserved His residue in *magenta spacefill*. SDRs and CERs on the ATPase are shown in *yellow*, or in *white* if located on the unresolved ATP-lid loop that was

superimposed from PhoQ kinase (PDB entry *IID0*), or in *green* in the RR-specific CERs side patch. SDRs and CERs on the DD are shown in *red* on one homodimer and *orange* on another (PDB entry *2C2A*) (Chapter 18, Fig. 6; *see* discussion on p. 435).

Color Plate 5 Localization of putative SDRs and CERs on computationally obtained models (models provided by Marina et al (27)). (a) HK in the active conformation, the ATPase is docked on the DD so that transfer of the phosphoryl group is possible. SDRs and CERs on the ATPase domain are shown in *yellow* or *green* when located in the RR-specific CERs side patch. SDRs and CERs on the DD are shown in *red* on one homodimer and orange on another. (b) Spo0F computationally docked on HK and subsequently superimposed with RR from OmpR. RR (*brown-red ribbon*) (PDB entry *IKGS*) with its 4 helix swung $\sim 90^\circ$: the phosphorylated Asp in the RR is shown in *magenta*, SDR and CERs are shown in *light red* or, when located the 4 helix in *white*. DD (*dark blue* and *dark green ribbon*): SDRs and CERs are shown in *light blue* on one dimer and in *light green* on another. ATPase (*yellow-green ribbon* on the *left* and *light-blue* on the *right*): the colors are the same as in (a) (Chapter 18, Fig. 7; *see* discussion on p. 439).

Color Plate 6 Valine aminoacyl-tRNA synthetase (PDB entry *IGAX*). The tRNA is shown as a *purple* wireframe structure, SDRs and CERs are *red* balls, and amino acid (valyl-adenylate analog) is in *yellow* wireframe (Chapter 18, Fig. 8; *see* discussion on p. 443).

Chapter 1

Identification of *cis*-Regulatory Elements in Gene Co-expression Networks Using A-GLAM

Leonardo Mariño-Ramírez, Kannan Tharakaraman, Olivier Bodenreider, John Spouge, and David Landsman

Abstract

Reliable identification and assignment of *cis*-regulatory elements in promoter regions is a challenging problem in biology. The sophistication of transcriptional regulation in higher eukaryotes, particularly in metazoans, could be an important factor contributing to their organismal complexity. Here we present an integrated approach where networks of co-expressed genes are combined with gene ontology-derived functional networks to discover clusters of genes that share both similar expression patterns and functions. Regulatory elements are identified in the promoter regions of these gene clusters using a Gibbs sampling algorithm implemented in the A-GLAM software package. Using this approach, we analyze the cell-cycle co-expression network of the yeast *Saccharomyces cerevisiae*, showing that this approach correctly identifies *cis*-regulatory elements present in clusters of co-expressed genes.

Key words: Promoter sequences, transcription factor-binding sites, co-expression, networks, gene ontology, Gibbs sampling.

1. Introduction

The identification and classification of the entire collection of transcription factor-binding sites (TFBSs) are among the greatest challenges in systems biology. Recently, large-scale efforts involving genome mapping and identification of TFBS in lower eukaryotes, such as the yeast *Saccharomyces cerevisiae*, have been successful (1). On the other hand, similar efforts in vertebrates have proven difficult due to the presence of repetitive elements and an increased regulatory complexity (2–4). The accurate prediction and identification of regulatory elements in higher eukaryotes remains a challenge for computational biology, despite recent

progress in the development of algorithms for this purpose (5). Typically, computational methods for identifying *cis*-regulatory elements in promoter sequences fall into two classes, enumerative and alignment techniques (6). We have developed algorithms that use enumerative approaches to identify *cis*-regulatory elements statistically significantly over-represented in promoter regions (7). Subsequently, we developed an algorithm that combines both enumeration and alignment techniques to identify statistically significant *cis*-regulatory elements positionally clustered relative to a specific genomic landmark (8).

Here, we will present a systems biology framework to study *cis*-regulatory elements in networks of co-expressed genes. This approach includes a network comparison operation, namely the intersection between co-expression and functional networks to reduce complexity and false positives due to co-expression linkage but absence of functional linkage. First, co-expression (9, 10) and functional networks (11, 12) are created using user-selected thresholds. Second, the construction of a single network is obtained from the intersection between co-expression and functional networks (13). Third, the highly interconnected regions in the intersection network are identified (14). Fourth, upstream regions of the gene clusters that are linked by both co-expression and function are extracted. Fifth, candidate *cis*-regulatory elements using A-GLAM (8) present in dense cluster regions of the intersection network are identified. In principle, the calculation of intersections for other types of networks with co-expression and/or functional networks could also be used to identify groups of co-regulated genes of interest (15) that may share *cis*-regulatory elements.

2. Materials

2.1. Hardware Requirements

1. Personal computer with at least 512 MB of random access memory (RAM) connected to the Internet.
2. Access to a Linux or UNIX workstation.

2.2. Software Requirements

1. The latest version of the Java Runtime Environment (JRE) freely available at <http://www.java.com/>.
2. The latest version of Cytoscape – a bioinformatics software platform for visualizing molecular interaction networks (13) freely available at <http://www.cytoscape.org/>.
3. The latest version of the MCODE plug-in for Cytoscape – finds clusters or highly interconnected regions in any network loaded into Cytoscape (14) freely available at <http://cbio.mskcc.org/~bader/software/mcode/>.

4. A modern version of the Perl programming language installed on the Linux or UNIX workstation freely available at <http://www.perl.com/>.
5. The A-GLAM package (8) freely available at <ftp://ftp.ncbi.nih.gov/pub/spouge/papers/archive/AGLAM/>.

3. Methods

The size of co-expression networks depends on the number of nodes in the network and the threshold used to define an edge between two nodes. There are a number of distance measures that are often used to compare gene expression profiles (16).

Here we use the Pearson correlation coefficient (PCC) as a metric to measure the similarity between expression profiles and to construct gene co-expression networks (17, 18). We establish a link by an edge between two genes, represented by nodes, if the PCC value is higher or equal to 0.7; this is an arbitrary cut-off that can be adjusted depending on the dataset used. The microarray dataset used here is the yeast cell-cycle progression experiment from Cho et al. (9) and Spellman et al. (10). The semantic similarity method (11) was used to quantitatively assess the functional relationships between *S. cerevisiae* genes.

The A-GLAM software package uses a Gibbs sampling algorithm to identify functional motifs (such as TFBSs, mRNA splicing control elements, or signals for mRNA 3'-cleavage and polyadenylation) in a set of sequences. Gibbs sampling (or more descriptively, successive substitution sampling) is a respected Markov-chain Monte Carlo procedure for discovering sequence motifs (19). Briefly, A-GLAM takes a set of sequences as input. The Gibbs sampling step in A-GLAM uses simulated annealing to maximize an 'overall score', a figure of merit corresponding to a Bayesian marginal log-odds score. The overall score is given by

$$s = \sum_{i=1}^m \left(\log_2 \frac{(a-1)!}{(c+a-1)!} + \sum_{(j)} \left\{ \log_2 \left[\frac{(c_{ij} + a_j - 1)!}{(a_j - 1)!} \right] - c_{ij} \log_2 p_j \right\} \right). \quad [1]$$

In Eq. [1], $m! = m(m-1) \dots 1$ denotes a factorial; a_j , the pseudo-counts for nucleic acid j in each position; $a = a_1 + a_2 + a_3 + a_4$, the total pseudo-counts in each position; c_{ij} , the count of nucleic acid j in position i ; and $c = c_{i1} + c_{i2} + c_{i3} + c_{i4}$, the total number of aligned windows, which is independent of the position i . The rationale behind the overall score s in A-GLAM is explained in detail elsewhere (8).

To initialize its annealing maximization, A-GLAM places a single window of arbitrary size and position at every sequence, generating a gapless multiple alignment of the windowed subsequences. It then proceeds through a series of iterations; on each iteration step, A-GLAM proposes a set of adjustments to the alignment. The proposal step is either a repositioning step or a resizing step. In a repositioning step, a single sequence is chosen uniformly at random from the alignment; and the set of adjustments include all possible positions in the sequence where the alignment window would fit without overhanging the ends of the sequence. In a resizing step, either the right or the left end of the alignment window is selected; and the set of proposed adjustments includes expanding or contracting the corresponding end of all alignment windows by one position at a time. Each adjustment leads to a different value of the overall score s . Then, A-GLAM accepts one of the adjustments randomly, with probability proportional to $\exp(s/T)$. A-GLAM may even exclude a sequence if doing so would improve alignment quality. The temperature T is gradually lowered to $T = 0$, with the intent of finding the gapless multiple alignment of the windows maximizing s . The maximization implicitly determines the final window size. The randomness in the algorithm helps it avoid local maxima and find the global maximum of s . Due to the stochastic nature of the procedure, finding the optimum alignment is not guaranteed. Therefore, A-GLAM repeats this procedure ten times from different starting points (ten runs). The idea is that if several of the runs converge to the same best alignment, the user has increased confidence that it is indeed the optimum alignment. The steps (below) corresponding to E -values and post-processing were then carried out with the PSSM corresponding to the best of the ten scores s .

The individual score and its E-value in A-GLAM: The Gibbs sampling step produces an alignment whose overall score s is given by **Eq. [1]**. Consider a window of length w that is about to be added to A-GLAM's alignment. Let $\delta_i(j)$ equal 1 if the window has nucleic acid j in position i , and 0 otherwise. The addition of the new window changes the overall score by

$$\Delta s = \sum_{i=1}^w \sum_{(j)} \delta_i(j) \left\{ \log_2 \left[\left(\frac{c_{ij} + a_j}{c + a} \right) / p_j \right] \right\}. \quad [2]$$

The score change corresponds to scoring the new window according to a position-specific scoring matrix (PSSM) that assigns the 'individual score'

$$s_i(j) = \log_2 \left[\left(\frac{c_{ij} + a_j}{c + a} \right) / p_j \right] \quad [3]$$

to nucleic acid j in position i . **Equation [3]** represents a log-odds score for nucleic acid j in position i under an alternative hypothesis with probability $(c_{ij} + a_j)/(c + a)$ and a null hypothesis with

probability p_{ij} . PSI-BLAST (20) uses Eq. [3] to calculate E -values. The derivation through Eq. [2] confirms the PSSM in Eq. [3] as the natural choice for evaluating individual sequences.

The assignment of an E -value to a subsequence with a particular individual score is done as follows: consider the alignment sequence containing the subsequence. Let n be the sequence length, and recall that w is the window size. If ΔS_i denotes the quantity in Eq. [2] if the final letter in the window falls at position i of the alignment sequence, then $\Delta S^* = \max\{\Delta S_i : i = w, \dots, n\}$ is the maximum individual score over all sequence positions i . We assigned an E -value to the actual value $\Delta S^* = \Delta s^*$, as follows. Staden's method (21) yields $\mathbb{P}\{\Delta S_i \Delta s^*\}$ (independent of i) under the null hypothesis of bases chosen independently and randomly from the frequency distribution $\{p_j\}$. The E -value $E = (n - w + 1)\mathbb{P}\{\Delta S_i \Delta s^*\}$ is therefore the expected number of sequence positions with an individual score exceeding Δs^* . The factor $n - w + 1$ in E is essentially a multiple test correction.

More recently, the A-GLAM package has been improved to allow the identification of multiple instances of an element within a target sequence (22). The optional 'scanning step' after Gibbs sampling produces a PSSM given by Eq. [3]. The new scanning step resembles an iterative PSI-BLAST search based on the PSSM. First, it assigns an 'individual score' to each subsequence of appropriate length within the input sequences using the initial PSSM. Second, it computes an E -value from each individual score to assess the agreement between the corresponding subsequence and the PSSM. Third, it permits subsequences with E -values falling below a threshold to contribute to the underlying PSSM, which is then updated using the Bayesian calculus. A-GLAM iterates its scanning step to convergence, at which point no new subsequences contribute to the PSSM. After convergence, A-GLAM reports predicted regulatory elements within each sequence in the order of increasing E -values; users then have a statistical evaluation of the predicted elements in a convenient presentation. Thus, although the Gibbs sampling step in A-GLAM finds at most one regulatory element per input sequence, the scanning step can now rapidly locate further instances of the element in each sequence.

3.1. Co-expression Network Construction

1. The yeast cell-cycle-regulated expression data are obtained from <http://cellcycle-www.stanford.edu/> (see Note 1).
2. Pairwise Pearson correlation coefficient (PCC) values are calculated using a subroutine implemented in the Perl programming language (23) (see Note 2).
3. The co-expression network is constructed with all gene pairs with a PCC greater or equal to 0.7 and is formatted according to the simple interaction file (SIF) described in the Cytoscape manual available at <http://www.cytoscape.org/> (see Note 3).

4. The co-expression network can be loaded in Cytoscape, which is an open-source software for integrating biomolecular interaction networks. Cytoscape is available for a variety of operating systems, including Windows, Linux, Unix, and Mac OS X.

3.2. Functional Similarity Network Construction

1. Gene ontology (GO) annotations for yeast gene products come from the *Saccharomyces* Genome Database (SGD) and were downloaded from http://www.geneontology.org/cgi-bin/downloadGOGA.pl/gene_association.sgd.gz. The evidence supporting such annotations is captured by evidence codes, including TAS (Traceable Author Statement) and IEA (Inferred from Electronic Annotation). While TAS refers to peer-reviewed papers and indicates strong evidence, IEA denotes automated predictions, not curated by experts, i.e., generally less reliable annotations. For this reason, IEA annotations were excluded from this study.
2. Functional relationships between *S. cerevisiae* genes were assessed quantitatively using a semantic similarity method (11) based on the gene ontology (GO). We first computed semantic similarity among GO terms from the *Biological Process* hierarchy using the Lin metric. This metric is based on information content and defines **term-term similarity**, i.e., the semantic similarity $\text{sim}(c_i, c_j)$ between two terms c_i and c_j as

$$\text{sim}(c_i, c_j) = \frac{2 \times \max_{c \in S(c_i, c_j)} [\log(p(c))]}{\log(p(c_i)) + \log(p(c_j))}, \quad [4]$$

where $S(c_i, c_j)$ represents the set of ancestor terms shared by both c_i and c_j , ‘max’ represents the maximum operator, and $p(c)$ is the probability of finding c or any of its descendants in the SGD database. It generates normalized values between 0 and 1. **Gene-gene similarity** results from the aggregation of term-term similarity values between the annotation terms of these two genes. In practice, given a pair of gene products, g_k and g_p , with sets of annotations A_k and A_p comprising m and n terms, respectively, the gene-gene similarity, $\text{SIM}(g_k, g_p)$, is defined as the *highest average* (inter-set) *similarity* between terms from A_i and A_j :

$$\text{SIM}(g_i, g_j) = \frac{1}{m+n} \times \left\{ \sum_k \max_p [\text{sim}(c_k, c_p)] + \sum_p \max_k [\text{sim}(c_k, c_p)] \right\}, \quad [5]$$

where $\text{sim}(c_i, c_j)$ may be calculated using Eq. [1]. This aggregation method (12) can be understood as a variant of the Dice similarity.

3. The functional similarity network is constructed using semantic similarity greater or equal to 0.7 and is formatted according to the simple interaction file (SIF).

4. Functional relationships in a group of genes can be further explored in Cytoscape using the BiNGO plug-in (24). Here we have used the hypergeometric test to assess the statistical significance ($p < 0.05$) and the Benjamini & Hochberg False Discovery Rate (FDR) correction (25).

3.3. Intersection Network Construction

1. The yeast co-expression and functional similarity networks are loaded in Cytoscape and the intersection network can be obtained by using the Graph Merge plug-in, freely available at the Cytoscape Web site. The nodes that are connected by having similar expression profiles and GO annotations are present in the intersection network (**Fig. 1.1**) (*see Note 4*).

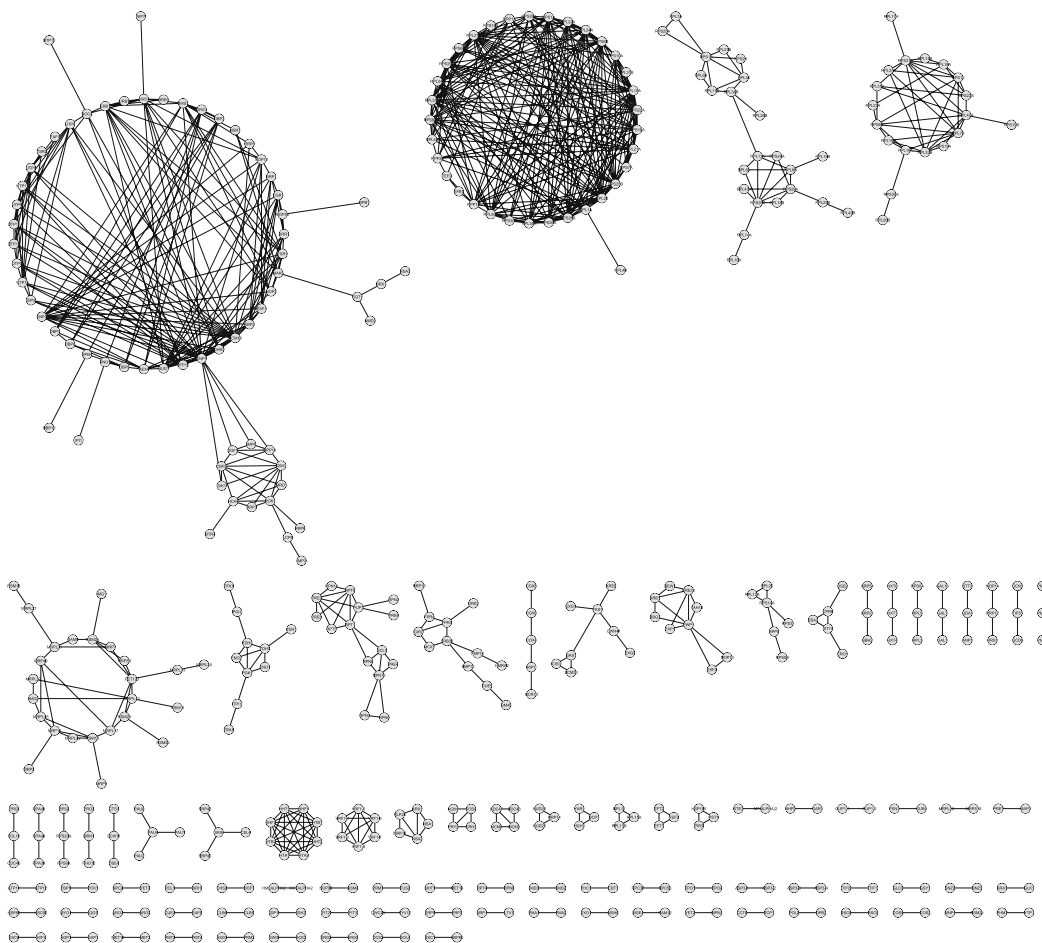


Fig. 1.1. Yeast cell-cycle gene co-expression and GO intersection network. The intersection network topology is shown for yeast genes, represented by nodes linked by one or more edges as described in the text. An edge represents both co-expression and functional linkage between the nodes connected.

2. The intersection network can be visualized using a variety of layouts in Cytoscape. A circular layout of the intersection network using the *yFiles Layouts* plug-in is depicted in **Fig. 1.1**.

3.4. Identification of Highly Interconnected Regions

1. The identification of dense gene clusters in the intersection network is done using the MCODE Cytoscape plug-in (14) (see **Note 5**). The clusters identified share similar expression patterns and functions as described by GO (**Fig. 1.2**).

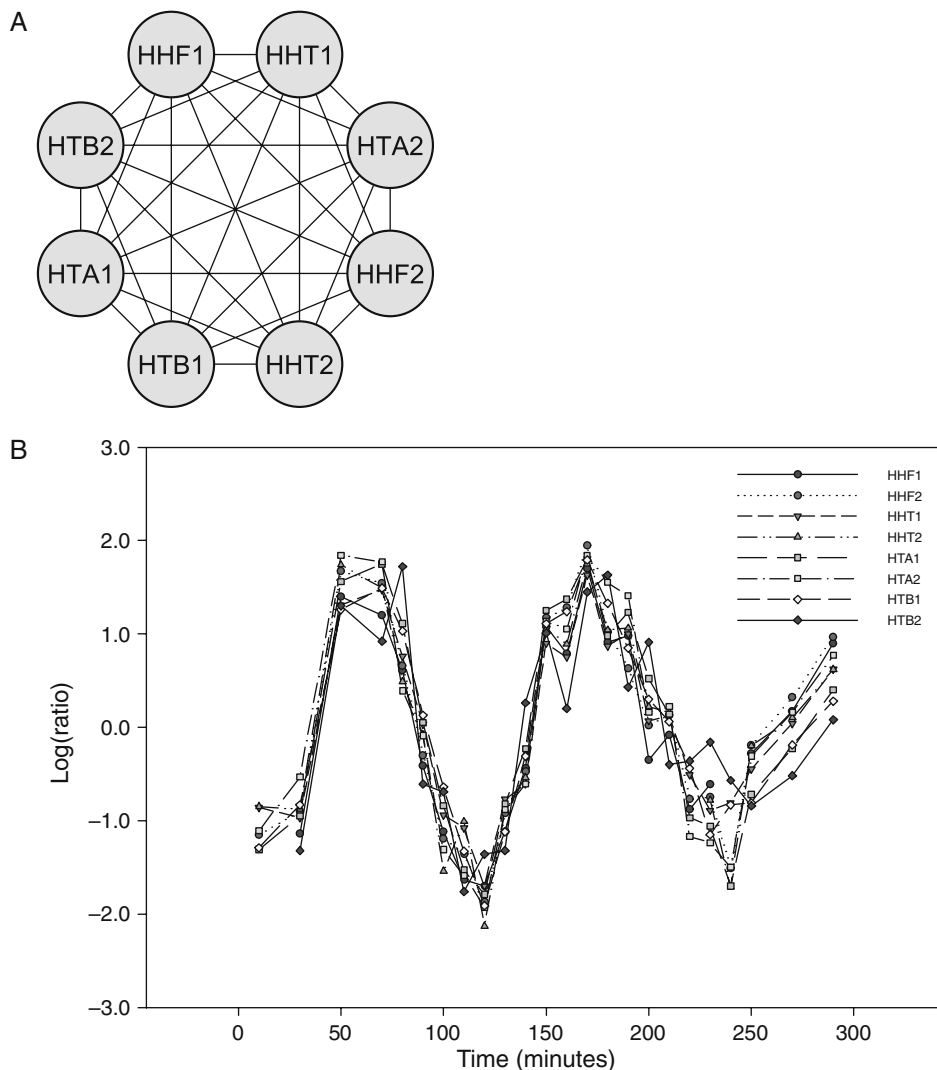


Fig. 1.2. Core histone gene cluster in the intersection network. A. Highly connected cluster identified by MCODE corresponds to eight core histone genes present in the yeast genome. The eight nodes are connected by 28 co-expression and functional edges. **B.** Expression profiles of the core histone genes over the cell cycle. **C.** Over-represented GO terms in the Biological Process category for the core histone genes. The statistical significance of each GO term is related to the intensity of the colored circles (see **Note 5**).

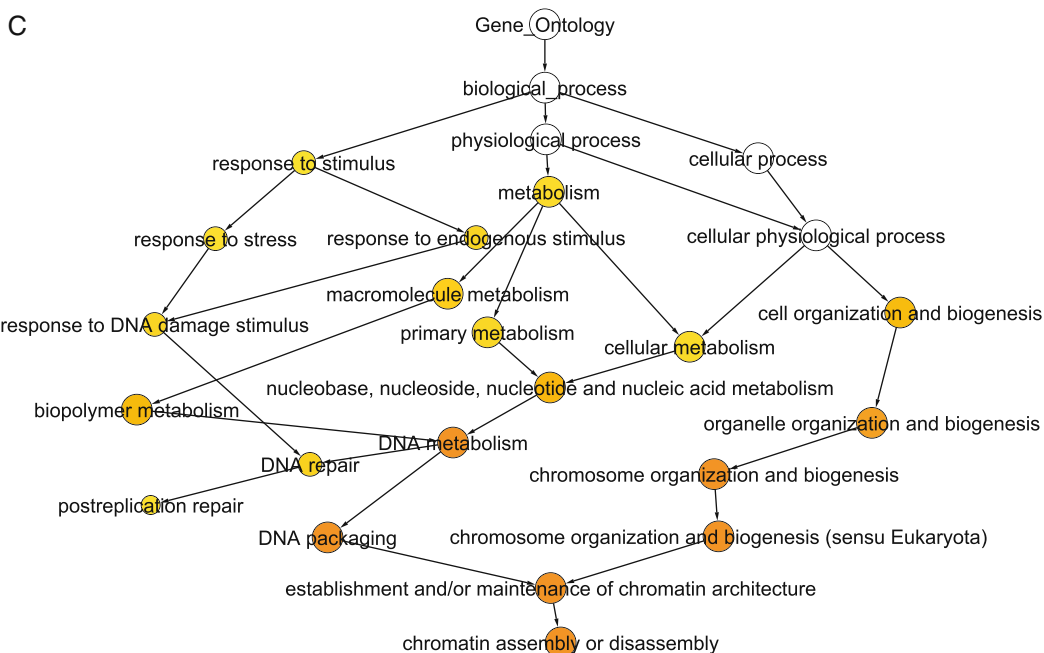


Fig. 1.2. (continued)

3.5. Identification of Proximal Promoter Regions

1. The *Saccharomyces* Genome Database (SGD) maintains the most current annotations of the yeast genome (see <http://www.yeastgenome.org/>). The SGD FTP site contains the DNA sequences annotated as intergenic regions in FASTA format (available at ftp://genome-ftp.stanford.edu/pub/yeast/sequence/genomic_sequence/intergenic/), indicating the 5' and 3' flanking features. Additionally, a tab-delimited file with the annotated features of the genome is necessary to determine the orientation of the intergenic regions relative to the genes (available at ftp://genome-ftp.stanford.edu/pub/yeast/chromosomal_feature/). The two files can be used to extract upstream intergenic regions (26) for the genes present in the intersection network clusters (see Note 6).

3.6. Identification of cis-Regulatory Elements in Promoter Regions

1. Construct FASTA files for each of the gene clusters identified by MCODE.
2. Install the A-GLAM package (see Note 7).
3. The A-GLAM package has a number of options that can be used to adjust search parameters (see Note 8).

```
$ aglam
```

Usage summary: aglam [options] myseqs.fa

Options:

- h help: print documentation
- n end each run after this many iterations without improvement (10,000)
- r number of alignment runs (10)
- a minimum alignment width (3)
- b maximum alignment width (10,000)
- j examine only one strand
- i word seed query ()
- f input file containing positions of the motifs ()
- z turn off ZOOPS (force every sequence to participate in the alignment)
- v print all alignments in full
- e turn off sorting individual sequences in an alignment on p-value
- q pretend residue abundances = 1/4
- d frequency of width-adjusting moves (1)
- p pseudocount weight (1.5)
- u use uniform pseudocounts: each pseudocount = p/4
- t initial temperature (0.9)
- c cooling factor (1)
- m use modified Lam schedule (default = geometric schedule)
- s seed for random number generator (1)
- w print progress information after each iteration
- l find multiple instances of motifs in each sequence
- k add instances of motifs that satisfy the cutoff e-value (0)
- g number of iterations to be carried out in the post-processing step (1,000)

4. Run A-GLAM to identify the regulatory elements present in the gene clusters with similar expression patterns and GO annotations (*see Note 9*). A-GLAM correctly identifies an experimentally characterized element known to regulate core histone genes in yeast (27). The alignments produced by A-GLAM can be represented by sequence logos (28, 29) and the positional preferences of the elements can be evaluated by plotting the score against relative positions, normalized by sequence length, in the promoter sequences (**Fig. 1.3**).

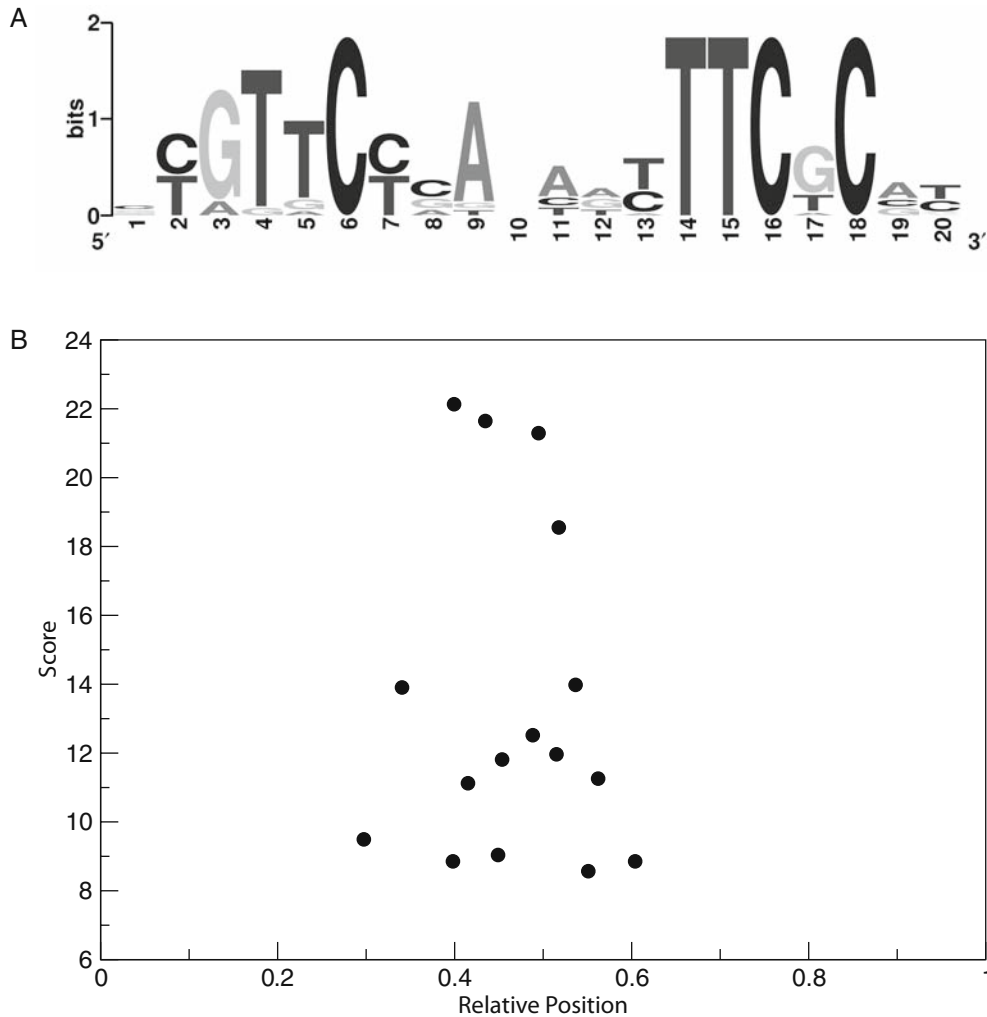


Fig. 1.3. **Core histone regulatory element identified with A-GLAM.** **A.** Sequence logo representation of the motif obtained from the ungapped multiple sequence alignment identified by A-GLAM (*see Note 9*). **B.** Positional preference plot for the elements identified by A-GLAM where the score in bits is plotted against the relative position of the element in the upstream regions of the core histone genes.

4. Notes



1. The yeast cell cycle data from the Web site include the experiments from Cho et al. (9) and Spellman et al. (10).
2. The following Perl code can be used to calculate the PCC:

```
my$r = correlation(\@{$values{$probe1}}, \@{$values
{$probe2}});
```

```

sub covariance {
  my ($array1ref,$array2ref) = @_;
  my ($i,$result);
  for ($i = 0;$i < @$array1ref;$i++) {$result += $array1
    ref->[$i] * $array2ref->[$i];
  }
  $result /= @$array1ref;
  $result -= mean($array1ref) * mean($array2ref);
}

sub correlation {
  my ($array1ref,$array2ref) = @_;
  my ($sum1,$sum2);
  my ($sum1_squared, $sum2_squared);
  foreach (@$array1ref) {$sum1 += $_;$sum1_squared
    += $_ ** 2 }
  foreach (@$array2ref) {$sum2 += $_;$sum2_squared
    += $_ ** 2 }
  return (@$array1ref ** 2) * covariance($array1ref,
    $array2ref) / sqrt(((@$array1ref * $sum1_squared) -
    ($sum1 ** 2)) * ((@$array1ref * $sum2_squared) -
    ($sum2 ** 2)));
}

sub mean {
  my ($arrayref) = @_;
  my $result;
  foreach (@$arrayref) {$result += $_ }
  return $result / @$arrayref;
}

```

3. The simple interaction file (SIF or .sif format) consists of lines where each node, representing a protein, is connected by an edge to a different protein in the network. Lines from the simple interaction file from the co-expression network:

```

RPL12A pp THR1
RPL12A pp TIF2
RPL12A pp TIF1
RPL12A pp GUK1
RPL12A pp URA5
RPL12A pp RPL1B
RPL12A pp SSH1
RPL12A pp SNU13
RPL12A pp RPL23B
SHU1 pp DON1

```


Two nodes are connected by a relationship type that in this case is pp. The nodes and their relationships are delimited by a space or a tab (see the Cytoscape manual for more detailed information).

4. Two or more networks can be used to calculate their intersection as needed to select for connections that meet certain criteria. The researcher can overlay protein–protein interactions, co-expression and functional networks to identify the protein complexes created under specific experimental conditions.
5. The MCODE plug-in ranks the clusters according to the average number of connections per protein in the complex (Score). The top five clusters identified by MCODE in the intersection network are shown below:

Cluster	Score	Proteins	Interactions
1	6.6	15	99
2	3.5	8	28
3	2.267	15	34
4	2	5	10
5	2	5	10

The BiNGO plug-in can be used to determine the GO terms statistically over-represented in a group of genes. Here we show the results for cluster 2:

Selected statistical test : Hypergeometric test
 Selected correction : Benjamini & Hochberg False Discovery Rate (FDR) correction
 Selected significance level : 0.05

Testing option : Test cluster versus complete annotation

The selected cluster :
 HHT1 HHF1 HTA1 HHT2 HHF2 HTA2 HTB1 HTB2

Number of genes selected : 8

Total number of genes in annotation : 5932

6. There are a number of Web sites that facilitate the extraction of promoter sequences. A service for the extraction of human, mouse, and rat promoters is freely available at <http://bio.wulf.bu.edu/zlab/promoser/>
7. The A-GLAM package is currently available in source code and binary forms for the Linux operating system (see <ftp://ftp.ncbi.nih.gov/pub/spouge/papers/archive/AGLAM/>).

GO ID	P-value	Corrected P-value	Description
6333	4.9168E-15	1.2292E-13	Chromatin assembly or disassembly
6325	2.2510E-12	1.8758E-11	Establishment and/or maintenance of chromatin architecture
6323	2.2510E-12	1.8758E-11	DNA packaging
7001	2.0415E-10	1.2759E-9	Chromosome organization and biogenesis (sensu Eukaryota)
51276	2.5897E-10	1.2949E-9	Chromosome organization and biogenesis
6259	5.9413E-9	2.4756E-8	DNA metabolism
6996	6.9565E-7	2.4845E-6	Organelle organization and biogenesis

Installation of the Linux binary: Get the executable from the FTP site and set execute permissions.

```
$chmod +x aglam
```

Installation from source: Unpack the glam archive and compile A-GLAM.

```
$tar -zxvf aglam.tar.gz
```

```
$cd aglam
```

```
$make aglam
```

8. Possible scenarios and options to modify A-GLAM's behavior.

```
$aglam <myseqs.fa>
```

This command simply uses the standard Gibbs sampling procedure to find sequence motifs in “myseqs.fa”.

```
$aglam <myseqs.fa> -n 20000 -a 5 -b 15 -j
```

This tells the program to search only the given strand of the sequences to find motifs of length between 5 and 15 bp. The flag *n* specifies the number of iterations performed in each of the ten runs. Low values of *n* are adequate when the problem size is small, i.e., when the sequences are short and more importantly there are few of them, but high values of *n* are needed for large problems. In addition, smaller values of *n* are sufficient when there is a strong alignment to be found, but larger values are necessary when there is not, e.g., for finding the optimal alignment of random sequences. You will have to

choose n on a case-by-case basis. This parameter also controls the tradeoff between speed and accuracy.

```
$aglam <myseqs.fa> -i TATA
```

This important option sets the program to run in a “seed”-oriented mode. The above command restricts the search to sequences that are TATA-like. Unlike the procedure followed in the standard Gibbs sampling algorithm, however, A-GLAM continues to align one exact copy of the “seed” in all “seed sequences”. Therefore, A-GLAM uses the seed sequences to direct its search in the remaining non-seed “target sequences”. Using this option leads to the global optimum quickly.

```
$aglam <myseqs.fa> -f <positions.dat>
```

The above command uses an extra option that allows A-GLAM to take a set of positions from an input file “positions.dat”. Like with the “-” flag, this option provides “seeds” for the A-GLAM alignment. Using this command restricts the Gibbs sampling step to aligning the original list of windows specified by the positions in the file. The seed sequences then direct the search in the remaining non-seed sequences.

```
$aglam <myseqs.fa> -l -k 0.05 -g 2000
```

Usable only with version 1.1. This tells the program to find multiple motif instances in each input sequence, via the scanning step (described above). Those instances that receive an E -value less than 0.05 are included in the PSSM. The search for multiple motifs is carried on until either (a) no new motifs are present or (b) the user-specified number of iterations (in this case, it is 2,000) is attained, whichever comes first.

9. A-GLAM uses sequences in FASTA format as input. Cluster number 2, identified by MCODE, is composed of eight genes linked by 28 co-expression and GO connections. Interestingly, the intergenic regions of the same cluster are shared between the genes in the cluster:

```
>B:235796-236494, Chr 2 from 235796-236494,
between YBL003C and YBL002W
TATATATTAATTTGCTCTTGTCTGTACTTTCCCTAATCTTATGTA
AAAAGACAAGAAT
TTATGATACTATTTAATAACAAAAAACTACCTAAGAAAAGCATCATGCAG
TCGAAATTGA
AATCGAAAAGTAAAACTTTAACGGAACATGTTTGAAATCTAAGAAAAGC
ATACATCTTCA
TCCCTTATATATAGAGTTATGTTTGATATTAGTAGTCATGTTGTAATCT
CTGGCCTAAGT
ATACGTAACGAAAATGGTAGCACGTCGCGTTTATGGCCCCAGGTTAAT
GTGTTCTCTGA
AATTCGCATCACTTTGAGAAAATAATGGGAACACCTTACGCGTGAGCTGT
GCCACCGCTT
CGCCTAATAAAGCGGTGTTCTCAAATTTCTCCCCGTTTTTCAGGATCAC
GAGCGCCATCT
```

AGTTCTGGTAAAATCGCGCTTACAAGAACAAAGAAAAGAAACATCGCGT
AATGCAACAGT
GAGACTTGCCGTCATATATAAGGTTTGGATCAGTAAACCGTTATTTG
AGCATAACACA
GGTTTTAAATATATTATATATATCATGGTATATGTGTAATAATTTTT
TGCTGACTGGT
TTTGTTTATTTATTTAGCTTTTTAAAAATTTACTTTCTTCTTGTAAAT
TTTTTCTGATT
GCTCTATACTCAAACCAACAACAACCTTACTCTACAATA
>D:914709-915525, Chr 4 from 914709-915525, between
YDR224C and YDR225W
TGTATGTGTGTATGGTTTATTTGTGGTTGACTTGCTATATAGGATAA
ATTTAATATAA
CAATAATCGAAAATGCGGAAAGAGAAACGTCTTTAATAAATCTGACCAT
CTGAGATGATC
AAATCATGTTGTTTATATACATCAAGAAAACAGAGATGCCCCTTTCTTA
CCAATCGTTAC
AAGATAACCAACCAAGGTAGTATTTGCCACTACTAAGGCCAATCTCTTT
GATTTTAAATC
CATCGTTCTCATTTTTTCGCGGAAGAAAGGGTGAACGCGCGAAAAAGT
GAGAACAGCCT
TCCCTTTCGGGCGACATTGAGCGCTAACCATAGTTAACGACCCAACCG
CGTTTTCTTCA
AATTTGAACTCGCCGAGCTCACAAATAATTCATTAGCGCTGTTCCAAAA
TTTTCGCCTCA
CTGTGCGAAGCTATTGGAATGGAGTG
TATTTGGTGGCTCAAAAAAGAGCACAAATAGTTA
ACTCGTCGTTGTTGAAGAAACGCCCGTAGAGATATGTGGTTTCTCATGC
TGTATTTGTT
ATTGCCACTTTGTTGATTTCAAATCTTTTCTCACCCCCTTCCCCGTT
CACGAAGCCAG
CCAGTGGATCGTAAATACTAGCAATAAGTCTTGACCTAAAAAATATATA
AATAAGACTCC
TAATCAGCTTGTAGATTTTCTGGTCTTGTGAACCATCATCTATTTACT
TCCAATCTGTA
CTTCTCTTCTTGATACTACATCATCATAACGGATTTGGTTATTTCTCAGT
GAATAACAAC
TTCAAAACAACAATTTATACATATAAAATATAAA
>N:576052-576727, Chr 14 from 576052-576727, between
YNL031C and YNL030W
TGTGGAGTGTGCTTGGATCCTTTAGTAAAAGGGGAAGAACAGTTGGAA
GGCCAAAGT
GGAAGTCACAAAACAGTGGTCCATATAAAAAGAACAAGAAAAGATTATT
TATATAACAAC
TGCGGTCACAAGAAGCAACGCGAGAGAGACAAACACGCTGTTATCACGCA
AACTATGTTT
TGACACCGAGCCATAGCCGTGATTTGTGCGTCACATTGGGCGATAATGAAC
GCTAAATGAC
CAACTCCCATCCGTAGGAGCCCCTTAGGGCGTGCCAATAGTTTACGCGC
TTAATGCGAA
GTGCTCGGAACGGACAACCTGTGGTCGTTTGGCACCGGAAAGTGGTACTA
GACCGAGAGT
TTCGATTTGTATGGCAGGACGTTCTGGGAGCTTCGCGTCTCAAGCTTTT
TCGGGCGCGA

```

AATGCAGACCAGACCAGAACAAAACAACCTGACAAGAAGCGCTTTAATTTA
ATATGTTGTT
CACTCGCGCCTGGGCTGTTGTTATTCGGCTAGATACATACGTGTTTGTGC
GTATGTAGTT
ATATCATATATAAGTATATTAGGATGAGCGGTGAAAGAGATTTTTTTT
TTTTCGCTTAA
TTTATTCTTTTCTCTATCTTTTTTCTACATCTTGTTCAAAAGAGTAGC
AAAAACAACAA
TCAATACAATAAAAATA
>B:255683-256328, Chr 2 from 255683-256328, between
YBR009C and YBR010W
ATTTTACTATATTATATTTGTTGCTTGTTTTTGTTTGTGCTTTAGTAC
TATAGAGTACA
ATAATGCGACGGAAACCATCATATAGAAAAATATCTCGGTATTTATAG
GAAAAAGAATT
AGACCTTTTCCACAACCAATTTATCCATCAAATTGGTCTTTACCCAATG
AATGGGGAAGG
GGGGGTGGCAATTTACCACCGTATTCGCGGGCATTGCTAAAGTAAACA
ACTTCGGTTTT
TACCACTAACCATTTATGGGGAGAAGCGCTCGGAACAGTTTTACTATGTG
AAGATGCGAAG
TTTTCAGAACGCGGTTTCCAAATTCGGCGGGGAGATACAAAAAGATTT
TTGCTCTCGTT
CTCACATTTTTCGCATTGTCCCATACATTATCGTTCTCACAAATTTCTCAC
ATTTCTTGTGCT
CTGCACCTTTGCGATCCTGGCCGTAATATCTCTCCTTGACTTTTAGCGT
GGAAGATAACG
AAATGCCCGGGCGATTTTTCTTTTTTGGTACCCTCCACGGCTCCTTGTG
AAATACATATA
TAAAAGACTGTGTATTCTTCGGGATACATCTCTTCTCAACCTTTTAT
ATTCTTTCTTT
CTAGTTAATAAGAAAAACATCTAACATAAATATATAAACGCAAACA

```

A-GLAM has a number of useful command line options that can be adjusted to improve *ab initio* motif finding; in this example we have restricted the search to motifs no larger than 20 bp.

```
$aglam -b 20 -l 02.fa
```

```
A-GLAM: Anchored Gapless Local Alignment of
Multiple
```

```

Sequences Compiled on Jun 2 2006
Run 1... 11724 iterations
Run 2... 10879 iterations
Run 3... 10878 iterations
Run 4... 10336 iterations
Run 5... 10181 iterations
Run 6... 10637 iterations
Run 7... 10116 iterations
Run 8... 11534 iterations
Run 9... 10097 iterations
Run 10... 10239 iterations

```

```
! The sequence file was [ 02.fa]
! Reading the file took [ 0] secs
! Sequences in file [ 4]
! Maximum possible alignment width [ 1292]
! Score [ 243.4] bits
! Motif Width [ 20]
! Runs [ 10]

! Best possible alignment:
>B:235796-236494, Chr 2 from 235796-236494, between
  YBL003C and YBL002W
365 AGGCGAAGCGGTGGGCACAG 346 - (21.29360)
  (2.820982e-08)
394 GGGAGAAATTTTGAGAACAC 375 - (13.97930)
  (5.205043e-04)
309 ATGCGAATTCAGAGAACAC 290 - (11.12770)
  (5.771870e-03)
314 TTGAGAAATAATGGGAACAC 333 + (9.034960)
  (2.714569e-02)
>D:914709-915525, Chr 4 from 914709-915525, between
  YDR224C and YDR225W
423 GTGCGAAGCTATTGGAATGG 442 + (18.55810)
  (2.256236e-06)
278 GCGCGAAAAAGTGAGAACAG 297 + (13.90430)
  (6.495526e-04)
418 AGGCGAAAATTTTGGAACAG 399 - (12.51460)
  (2.007017e-03)
262 CCGCGAAAAAATGAGAACGA 243 - (9.499530)
  (2.299132e-02)
>N:576052-576727, Chr 14 from 576052-576727,
  between YNL031C and YNL030W
294 ATGCGAAGTGCTCGGAACGG 313 + (21.65330)
  (1.526033e-08)
367 ATGCGAAACTCTCGGTCTAG 348 - (11.95760)
  (2.781407e-03)
399 ACGCGAAGCTCCCAGAACGT 380 - (11.25120)
  (5.253971e-03)
288 GCGTGAAACTATTGGCACGC 269 - (8.853600)
  (3.961768e-02)
>B:255683-256328, Chr 2 from 255683-256328, between
  YBR009C and YBR010W
258 GGGAGAAGCGCTCGGAACAG 277 + (22.13350)
  (6.281785e-09)
293 ATGCGAAGTTTTCAGAACGC 312 + (11.81510)
  (3.041439e-03)
409 GTGAGAAATTGTGAGAACGA 390 - (8.852760)
  (3.780865e-02)
375 ATGCGAAAATGTGAGAACGA 356 - (8.564750)
  (4.774790e-02)
```

```

! 16 sequences in alignment
! Residue abundances:Pseudocounts
! A = 0.312544:0.468816 C = 0.187456:0.281184
! G = 0.187456:0.281184 T = 0.312544:0.468816
! Total Time to find best alignment [ 15.87] secs

```

Acknowledgments

The authors would like to thank King Jordan for important suggestions and helpful discussions and Alex Brick for his assistance in obtaining intergenic regions during his internship at NCBI. This research was supported by the Intramural Research Program of the NIH, NLM, NCBI.

References

1. Harbison CT, Gordon DB, Lee TI, Rinaldi NJ, Macisaac KD, Danford TW, Hannett NM, Tagne JB, Reynolds DB, Yoo J et al. Transcriptional regulatory code of a eukaryotic genome. *Nature* 2004, 431(7004): 99–104.
2. Bieda M, Xu X, Singer MA, Green R, Farnham PJ. Unbiased location analysis of E2F1-binding sites suggests a widespread role for E2F1 in the human genome. *Genome Res* 2006, 16(5):595–605.
3. Cawley S, Bekiranov S, Ng HH, Kapranov P, Sekinger EA, Kampa D, Piccolboni A, Sementchenko V, Cheng J, Williams AJ et al. Unbiased mapping of transcription factor binding sites along human chromosomes 21 and 22 points to widespread regulation of noncoding RNAs. *Cell* 2004, 116(4): 499–509.
4. Guccione E, Martinato F, Finocchiaro G, Luzi L, Tizzoni L, Dall' Olio V, Zardo G, Nervi C, Bernard L, Amati B. Myc-binding-site recognition in the human genome is determined by chromatin context. *Nat Cell Biol* 2006, 8(7):764–770.
5. Tompa M, Li N, Bailey TL, Church GM, De Moor B, Eskin E, Favorov AV, Frith MC, Fu Y, Kent WJ et al. Assessing computational tools for the discovery of transcription factor binding sites. *Nat Biotechnol* 2005, 23(1): 137–144.
6. Ohler U, Niemann H. Identification and analysis of eukaryotic promoters: recent computational approaches. *Trends Genet* 2001, 17(2): 56–60.
7. Marino-Ramirez L, Spouge JL, Kanga GC, Landsman D. Statistical analysis of over-represented words in human promoter sequences. *Nucleic Acids Res* 2004, 32(3): 949–958.
8. Tharakaraman K, Marino-Ramirez L, Sheettlin S, Landsman D, Spouge JL. Alignments anchored on genomic landmarks can aid in the identification of regulatory elements. *Bioinformatics* 2005, 21(Suppl 1): i440–448.
9. Cho RJ, Campbell MJ, Winzeler EA, Steinmetz L, Conway A, Wodicka L, Wolfsberg TG, Gabrielian AE, Landsman D, Lockhart DJ et al. A genome-wide transcriptional analysis of the mitotic cell cycle. *Mol Cell* 1998, 2(1):65–73.
10. Spellman PT, Sherlock G, Zhang MQ, Iyer VR, Anders K, Eisen MB, Brown PO, Botstein D, Futcher B. Comprehensive identification of cell cycle-regulated genes of the yeast *Saccharomyces cerevisiae* by microarray hybridization. *Mol Biol Cell* 1998, 9(12):3273–3297.
11. Lord PW, Stevens RD, Brass A, Goble CA. Investigating semantic similarity measures across the Gene Ontology: the relationship between sequence and annotation. *Bioinformatics* 2003, 19(10): 1275–1283.
12. Azuaje F, Wang H, Bodenreider O. Ontology-driven similarity approaches to supporting gene functional assessment. In: *Proceedings of the ISMB'2005 SIG Meeting on Bio-Ontologies*. Detroit, MI, 2005:9–10.

13. Shannon P, Markiel A, Ozier O, Baliga NS, Wang JT, Ramage D, Amin N, Schwikowski B, Ideker T. Cytoscape: a software environment for integrated models of biomolecular interaction networks. *Genome Res* 2003, 13(11):2498–2504.
14. Bader GD, Hogue CW. An automated method for finding molecular complexes in large protein interaction networks. *BMC Bioinformatics* 2003, 4:2.
15. Tsaparas P, Marino-Ramirez L, Bodenreider O, Koonin EV, Jordan IK. Global similarity and local divergence in human and mouse gene co-expression networks. *BMC Evol Biol* 2006, 6:70.
16. Babu MM. An introduction to microarray data analysis. In: *Computational Genomics: Theory and Application*, Edited by Grant RP. Cambridge, UK: Horizon Bioscience, 2004:225–249.
17. Jordan IK, Marino-Ramirez L, Koonin EV. Evolutionary significance of gene expression divergence. *Gene* 2005, 345(1): 119–126.
18. Jordan IK, Marino-Ramirez L, Wolf YI, Koonin EV. Conservation and coevolution in the scale-free human gene coexpression network. *Mol Biol Evol* 2004, 21(11): 2058–2070.
19. Lawrence CE, Altschul SF, Boguski MS, Liu JS, Neuwald AF, Wootton JC. Detecting subtle sequence signals: a Gibbs sampling strategy for multiple alignment. *Science* 1993, 262(5131):208–214.
20. Altschul SF, Madden TL, Schaffer AA, Zhang J, Zhang Z, Miller W, Lipman DJ. Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Res* 1997, 25(17): 3389–3402.
21. Staden R. Methods for calculating the probabilities of finding patterns in sequences. *Comput Appl Biosci* 1989, 5(2):89–96.
22. Tharakaraman K, Marino-Ramirez L, Sheetlin S, Landsman D, Spouge JL. Scanning sequences after Gibbs sampling to find multiple occurrences of functional elements. *BMC Bioinformatics* 2006, 7:408.
23. Orwant J, Hietaniemi J, Macdonald J. *Mastering Algorithms with Perl*. Sebastopol, CA: O'Reilly, 1999.
24. Maere S, Heymans K, Kuiper M. BiNGO: a Cytoscape plugin to assess overrepresentation of gene ontology categories in biological networks. *Bioinformatics* 2005, 21(16): 3448–3449.
25. Benjamini Y, Hochberg Y. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *J Roy Statist Soc Ser B* 1995, 57(1):289–300.
26. Marino-Ramirez L, Jordan IK, Landsman D. Multiple independent evolutionary solutions to core histone gene regulation. *Genome biology* 2006, 7(12):R122.
27. Eriksson PR, Mendiratta G, McLaughlin NB, Wolfsberg TG, Marino-Ramirez L, Pompa TA, Jainerin M, Landsman D, Shen CH, Clark DJ. Global regulation by the yeast Spt10 protein is mediated through chromatin structure and the histone upstream activating sequence elements. *Mol Cell Biol* 2005, 25(20):9127–9137.
28. Schneider TD, Stephens RM. Sequence logos: a new way to display consensus sequences. *Nucleic Acids Res* 1990, 18(20): 6097–6100.
29. Crooks GE, Hon G, Chandonia JM, Brenner SE. WebLogo: a sequence logo generator. *Genome Res* 2004, 14(6): 1188–1190.

Chapter 2

Structure-Based *Ab Initio* Prediction of Transcription Factor–Binding Sites

L. Angela Liu and Joel S. Bader

Abstract

We present an all-atom molecular modeling method that can predict the binding specificity of a transcription factor based on its 3D structure, with no further information required. We use molecular dynamics and free energy calculations to compute the relative binding free energies for a transcription factor with multiple possible DNA sequences. These sequences are then used to construct a position weight matrix to represent the transcription factor–binding sites. Free energy differences are calculated by morphing one base pair into another using a multi-copy representation in which multiple base pairs are superimposed at a single DNA position. Water-mediated hydrogen bonds between transcription factor side chains and DNA bases are known to contribute to binding specificity for certain transcription factors. To account for this important effect, the simulation protocol includes an explicit molecular water solvent and counter-ions. For computational efficiency, we use a standard additive approximation for the contribution of each DNA base pair to the total binding free energy. The additive approximation is not strictly necessary, and more detailed computations could be used to investigate non-additive effects.

Key words: Transcription factor–binding sites, molecular dynamics, free energy, position weight matrix (PWM), multi-copy, thermodynamic integration, protein–DNA binding.

1. Introduction

Transcription factors are DNA-binding proteins that control gene expression (*1*). They often recognize short DNA sequences (about six to eight base pairs long, roughly the number of base pairs exposed on the single face of a DNA major groove) that can be degenerate. Traditionally, binding sites have been obtained using

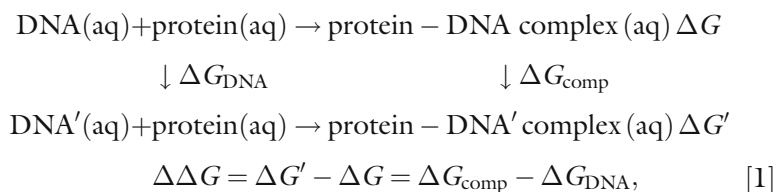
experimental methods, including SELEX (2), ChIP-chip (3), protein-binding microarrays (4), etc. These methods are often labor-intensive and expensive.

The binding sites of a transcription factor are intrinsically determined by the 3D structures of the protein and DNA and their structural complementarities. Binding sites of a transcription factor may also depend on its participation in a multi-protein complex. It is therefore desirable to predict these binding sites based on the 3D structures of transcription factors. This *ab initio* approach uses all-atom molecular simulation and remains a challenging problem. Several previous attempts (5–7) are limited to implicit solvent, enthalpic calculations of the free energy and frozen macromolecular backbones, all of which could lead to a bias in the binding site prediction.

In this chapter, we present an improved and, in principle, exact method (at least to the level of accuracy of molecular force fields) that can predict the transcription factor–binding sites using their structural information. There is no other required information, except for a well-chosen atomic force field for the representation of the protein–DNA complex.

The theoretical basis for structure-based binding site predictions for transcription factors is the binding free energy of the protein–DNA complex, calculated as the difference in free energy between the solvated complex and the solvated individual protein and DNA components. A transcription factor could possibly bind to multiple different DNA sequences with comparable binding affinity. This is because both DNA and protein are highly flexible molecules. Once a DNA base pair is changed to a different base pair and its prior favorable contacts with the protein are disrupted, protein and DNA can relax and change their geometries to achieve alternative favorable binding conformation. Typically, a specific DNA sequence and a non-specific DNA sequence to the same transcription factor differ only in binding energy on the magnitude of 10 kcal/mol. This is roughly equivalent to the energy of breaking two to five hydrogen bonds, as hydrogen bonds formed between oxygen and nitrogen atoms are typically 2–5 kcal/mol.

The relative binding free energy of a transcription factor with two different DNA sequences can be obtained using the following thermodynamic cycle:



where $\Delta\Delta G$ is the relative binding free energy of the protein with DNA and DNA'. The two horizontal reactions represent the association of the protein with two different DNA sequences. These binding free energies can be obtained from experimental measurement. The two vertical reactions represent mutations of changing the DNA sequence in the DNA duplex (ΔG_{DNA}) and the protein–DNA complex (ΔG_{comp}). In computation, it is the two vertical or “mutational” reactions that are calculated. There are two common methods for the calculation of such “mutational” free energies: free energy perturbation and thermodynamic integration. From our experience, we found that the latter method, thermodynamic integration, is easy to implement and provides more opportunities for extension such as free energy decomposition analyses. In this chapter, we use this method exclusively.

A prevalent representation of the transcription factor–binding site is a position weight matrix (PWM), which can be converted into a sequence logo for graphical representation. In order for the PWM representation to be valid, each base pair must contribute independently or additively to the total binding free energy, commonly called the “additive approximation”. In transcription factor – DNA complexes that have relatively small deformations in the DNA structure, this assumption has been observed to be a fairly good simplification (6). In this chapter, we will also use this additive approximation and point out ways to assess the non-additivity in **Note 1**.

At each base pair position along the DNA, there are four possible Watson-Crick base pairs. **Equation [1]** can be used to calculate the relative binding free energies among these four possible base pairs, which will result in a four-level energy diagram. The base pair with the lowest energy leads to the strongest binding, and is normally the base that appears in the experimental consensus binding sequence. These relative energies can be converted into probabilities using the Boltzmann factor, as in

$$\Pr(\text{bp} = \alpha, \alpha \in \{\text{A, C, G, T}\}) = \frac{\exp[-\beta(E_\alpha - E_0)]}{\sum_{\gamma \in \{\text{A, C, G, T}\}} \exp[-\beta(E_\gamma - E_0)]}, \quad [2]$$

where the four possible base pairs are labeled as A, C, G, or T; α and γ represent possible base pair identities; β is the inverse temperature (i.e., $1/k_{\text{B}}T$, k_{B} is the Boltzmann constant and T is the temperature); E_α and E_0 represent the free energy of the base pair α and the free energy of a reference base pair. Then $(E_\alpha - E_0)$ corresponds to the $\Delta\Delta G$ of **Eq. [1]** for changing the reference base pair into base pair α . For convenience, we choose the base pair leading to the lowest free energy (thus the strongest binder) as the reference point.

These probabilities can then be converted into a sequence logo (8) using the following formula,

$$IC(l) = 2 + \sum_{\alpha \in \{A,C,G,T\}} \Pr(\alpha, l) \log_2 \Pr(\alpha, l), \quad [3]$$

where $IC(l)$ represents the information content (in bits) at base pair position l ; $\Pr(\alpha, l)$ represents the probability (from Eq. [2]) of base pair α at position l . In the sequence logo, the letters A, C, G, and T (representing the corresponding base pair) are stacked on top of each other in the order of descending probability at each base pair position. The relative height of each base pair at a position is proportional to their corresponding probabilities. The maximum height of information content at each position is 2 bits, representing 100% conservation at the position; the minimum height is 0, representing equal probabilities for all four possible base pairs.

Taking the vertical reaction ΔG_{comp} as an example, the free energy simulation and analysis can be done as follows. We will use a single base pair change as an example, using the above-mentioned additive approximation. First, a protein–DNA complex structure is made. Then a base pair at a specific position is changed to another possible base pair. These two structures represent the reactant and the product of the reaction. Our job is to calculate the free energy change associated with the reaction. Because free energy is a state function, we can connect the reactant and the product using an arbitrary reaction path, and integrate the energy gradient along the path to obtain the total free energy change. This approach is called thermodynamic integration. The formal derivation of this method can be found in Leach’s introductory modeling book (9), as well as most of the theoretical background for this chapter. More advanced treatments are also available (10). Here we list the equations that are pertinent to the discussion. The energy function of the system is

$$H = H_0 + (1 - \lambda)H_{\text{reac}} + \lambda H_{\text{prod}}, \quad [4]$$

where H is the total Hamiltonian of the system that contains all the energetic terms; H_0 is the energy terms for the environmental atoms, comprising all those other than the reactant and product; H_{reac} and H_{prod} represent the energy terms associated with atoms in the reactant and the product, respectively; and λ represents the reaction coordinate (aka coupling parameter). Here the reactant refers to the original base pair; the product refers to the final base pair; and the environment refers to the atoms of the DNA backbone, other DNA base pairs, protein, and the solvent. From this equation we can see that the Hamiltonian becomes that of the reactant system when λ is 0, and becomes that of the product system when λ is 1. At intermediate λ values, the Hamiltonian corresponds to an artificial system that contains both the reactant

atoms and the product atoms. The reactant and the product, however, do not have any interaction terms, allowing them to occupy the same space.

Based on the linear coupling scheme of **Eq. [4]**, the free energy change for changing the base pair is

$$\Delta G = \int_0^1 d\lambda \left\langle \frac{\partial H}{\partial \lambda} \right\rangle_\lambda = \int_0^1 d\lambda \langle H_{\text{prod}} - H_{\text{reac}} \rangle_\lambda, \quad [5]$$

where the angular brackets “ $\langle \rangle_\lambda$ ” represent an ensemble average at a particular value of λ . In practice, the free energy simulation is done using traditional molecular dynamics methods, except that the energy function is now evaluated using **Eq. [4]**. After every 1 ps or so, the simulation trajectory will be saved. When the simulation is done, the saved trajectory will be analyzed using **Eq. [5]** to obtain the ensemble average of the Hamiltonian gradient. Typically, a numerical integration scheme is used to compute the free energy change for the reaction, such as the trapezoidal rule.

2. Materials

We list here the required computational resources for carrying out the computations discussed in the next section. The computational cost is listed in **Note 2**.

The majority of the calculations are done using a molecular modeling package called CHARMM (<http://www.charmm.org>) that requires a license. The version for wide distribution as of Jan. 2009 is c35b1. We have carried out all calculations using version c32b1. CHARMM requires FORTRAN90 compiler. On a Linux computer with Intel processors, the GNU FORTRAN compiler suffices. On Apple PowerPC computers with IBM processors, the IBM FORTRAN compiler is required. The benchmarks for these two architectures lead to similar running time for identical molecular test systems in serial mode, where the Intel processor is 3.0 GHz and the IBM processor is 2.2 GHz. CHARMM requires a moderate amount of memory at about 250 MB on the above two architectures for a system with about 25,000 atoms. The CHARMM executable is also available at public supercomputer sites, such as BigBen at the Pittsburgh Supercomputing Center (PSC), which has a parallel version of CHARMM installed (proof of license is required for usage). Benchmarks for additional systems are available from CHARMM’s website, which lists a wide range of supported architectures and compilers.

We chose CHARMM because we found it the easiest for implementing the calculations we desired (*see Note 3*). The CHARMM27 atomic force field has been well tested to be accurate for the description of proteins and nucleic acids. In tests on BigBen at PSC, we have found a drop-off in performance for running CHARMM on more than 16 compute nodes (32 CPUs) in parallel. This drop-off may be system-specific or even due to inexperience on our part. We did not investigate this issue because parallelization of the code is not particularly important for our calculations. Calculations may be trivially parallelized by simulating each nucleotide on a separate node (*see Note 4*).

3. Methods

3.1. Simulation Protocols for Native DNA Duplex or Native Protein–DNA Complex

The starting point of the simulation is the 3D structure of the protein–DNA complex of interest. The structure can be obtained from X-ray crystallography, NMR determination, or homology modeling. We outline the protocol in **Fig. 2.1** and explain the steps below. The same protocols are carried out for the protein–DNA complex as well as the DNA duplex in the complex. This is necessary according to the thermodynamic cycle in **Eq. [1]**.

3.1.1. Preparation of the Complete Structure File

CHARMM incorporates PDB (*11*) structural files to initiate the molecular modeling and simulation. The starting structure's PDB file must be edited to follow CHARMM's naming convention. This may be done manually. One can also write a computer program to do these modifications once they become familiar with the required changes for amino acids and nucleotides. If the starting structure is from crystallography, missing side chains will be added by CHARMM. If the structure file is obtained from NMR determination, the hydrogen atoms need to be removed first, and CHARMM's HBUILD module is used to add hydrogen atoms according to its own naming convention. Any water molecules that are resolved in the original structures are also removed.

A common practice for the assignment of charge state of titratable amino acid residues is to assign a +1 charge for all basic residues including lysine and arginine, assign a –1 charge to all acidic residues including glutamate and aspartate, and finally assign a +1 charge to histidine residues that are exposed at the protein surface. These assignments are appropriate for near-neutral pH values. If histidine is buried in the protein core, then more advanced studies are required to assign its proper protonation state. For the transcription factors we have simulated to date, all histidine residues are exposed at the surface.

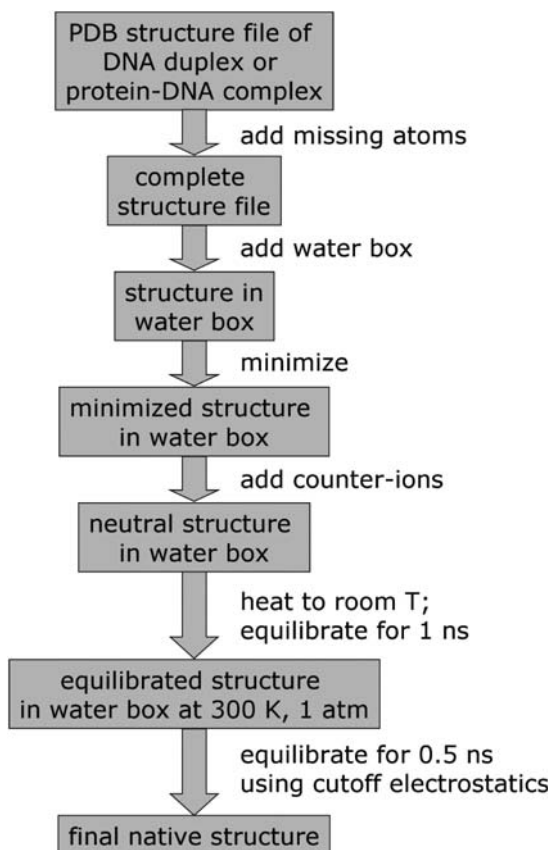


Fig. 2.1. Simulation protocol for generating a fully equilibrated native protein–DNA complex or DNA duplex structure in explicit solvent.

The ends of the protein contain a positively charged N-terminus and a negatively charged C-terminus. For the DNA section of the structure file, the 5' end phosphate groups of both strands are removed. Other possible end-cappings are also supported in CHARMM.

Once the initial PDB file is edited to conform to CHARMM's convention and missing atoms are added, we will have a dry protein–DNA complex or DNA duplex with no solvent atoms.

3.1.2. Introduction of Explicit Water Molecules and Preparation of Minimized Structure in Water Box

Since we consider explicitly the role of water in the binding of protein and DNA, we now add water molecules to the dry complex structure (*see Note 5*) to form a solvated system in a periodic boundary condition. Because the water model TIP3P was used during the development of the current CHARMM force field CHARMM27, we recommend its usage over other water models.

Once the water molecules are added, a series of minimizations are required to allow the water molecules to relax around the macromolecules. The recommended minimization algorithms

are Steepest Descent at the initial stage of the minimizations, and then Adopted Basis Newton-Raphson method for more refined minimizations. We use 1,000 steps of the former and 3,000 steps of the latter. The energy of the system should be decreasing steadily and reach stability. However, we do not advise running long minimizations to achieve convergence (to reach absolute zero K in temperature), as all our molecular dynamic simulation and free energy calculations need to be carried out at room temperature.

3.1.3. Introduction of Counter-ions

After the system in the water box is minimized, we use the CHARMM script file written by Rick Venable (available from CHARMM Discussion Forum Script Archive at <http://www.charmm.org/ubbthreads/ubbthreads.php?Cat=0>) to replace an appropriate number of water molecules with counter-ions. For the protein-DNA complexes we have studied, typically about ten sodium ions are required to neutralize the system. For a 10-base pair DNA duplex, 18 sodium ions are required. In Venable's script, the same number of water molecules as the desired counter-ions are selected at random and replaced by sodium ions. Then the system is minimized for 50 steps by Steepest Descent and by Adopted Basis Newton-Raphson. One hundred different sets of water selections are done. The lowest energy configuration among them is chosen to proceed to the next step.

3.1.4. Heating and Equilibration of the Structure

Since minimization freezes many degrees of freedom of the system, the solvent box is roughly about 50 K in temperature. Now we heat the system to room temperature and equilibrate it for 1.5 ns. We ramp up the temperature linearly from 50 to 300 K over 50 ps at a heating speed of 5 K per ps. During equilibration, constant temperature (300 K) and constant pressure (1 atm) are maintained using CHARMM's CPT keyword. This corresponds to the NPT ensemble. A time step of 1 fs is used. SHAKE is used to constrain all the bonds with hydrogen atoms to be at the equilibrium values. All other degrees of freedom are allowed.

The BLOCK module for free energy analysis has a limitation in that it requires the electrostatic interactions to be evaluated using non-Ewald methods, i.e., spherical cutoffs. Long, computationally expensive cutoffs are required to obtain an adequate representation of long-range electrostatic interactions. To reach a compromise between accuracy and computational saving, we carry out initial equilibration of the system for 1 ns using Ewald summation method Particle Mesh Ewald. Then we switch to spherical cutoff scheme using a cutoff value of 14 Å. Further equilibration of 0.5 ns is run at this condition.

After the 1.5 ns equilibration, the native protein-DNA complex structure is now considered well equilibrated. We need to note here that this equilibration time is still far too short for the

equilibration of the counter-ions, which typically requires much longer equilibrations on the scale of tens to hundreds of ns. Please see **Note 6** for strategies that avoid long equilibrations for ions.

3.2. Simulation Protocols for Free Energy Calculations

3.2.1. Multi-Copy Base Pairs

CHARMM supports dual-topology, which means that in the “mutational” reactions, the reactant and the product chemical groups co-exist in the structure. This is also known as “multi-copy” representation, where multiple functional groups occupy possibly the same space; their interactions with the rest of the system are scaled by a coupling parameter, but there are no interactions among the multiple copies. As we have discussed in the Introduction, thermodynamic integration is an established method for calculating the free energy change associated with changing one functional group in the multi-copy into another. In the simulations that are discussed in this chapter, we consider only the co-existence of two possible base pairs at any base pair position. **Figure 2.2** illustrates the construction of such structures. We call these 2-base multi-copy base pairs, or in short multi-copy base pairs. Details on how to create structures with multi-copy bases and how to enable CHARMM to evaluate their force and energy functions are in **Notes 7** and **8**.

3.2.2. Using BLOCK for Simulation and Free Energy Analysis

The BLOCK module in CHARMM allows straightforward force and energy evaluation of multi-copies. Here we use a simple example to illustrate its usage. Imagine a protein–DNA complex in which one base pair is a multi-copy base. Using **Eq. [4]**, the total Hamiltonian that contains the contributions from the environment, the reactant, and the product will be further separated into six contributions, as in **Eq. [6]** in BLOCK.

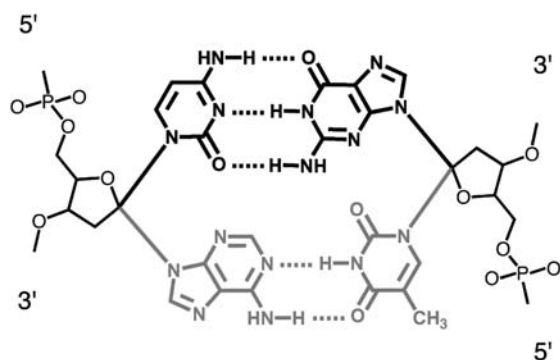


Fig. 2.2. Schematic diagram of multi-copy base pair. Single base pair change from A (gray, bottom base pair) to C (black, top base pair) is used as an example. The multi-copy base pair is referred to as A/C. The left strand is treated as the leading strand. The bases within each physical base pair interact normally, as evidenced by the hydrogen bonds (dotted lines) between complementary bases. The gray (reactant) atoms do not interact with the black (product) atoms.

$$\begin{array}{cccc}
& \text{Env} & \text{Reac} & \text{Prod} \\
\text{Env} & 1 & 1 - \lambda & \lambda \\
\text{Reac} & & 1 - \lambda & 0 \\
\text{Prod} & & & \lambda
\end{array} \tag{6}$$

This matrix is used for force and energy evaluation. The labels “Env”, “Reac”, and “Prod” represent the environmental atoms, the atoms in the reactant, and the atoms in the product, respectively. The matrix is symmetric so the lower half is not shown. Each element in the matrix represents the interaction term between the atoms in the corresponding row and the atoms in the corresponding column. For example, the term H_0 in **Eq. [4]** is 1 in the matrix and represents the interactions between atoms in the environment. Note that the interactions that involve reactant atoms are scaled by the $(1-\lambda)$ coupling parameter, whereas those involving product atoms are scaled by λ , just like in **Eq. [4]**. Finally, there are no interactions between the atoms of the reactant and the atoms of the product, hence the zero in the matrix.

BLOCK uses a different matrix to calculate the Hamiltonian gradient in **Eq. [5]** for the free energy analysis. The matrix is listed in **Eq. [7]**.

$$\begin{array}{cccc}
& \text{Env} & \text{Reac} & \text{Prod} \\
\text{Env} & 0 & -1 & 1 \\
\text{Reac} & & -1 & 0 \\
\text{Prod} & & & 1
\end{array} \tag{7}$$

Note that at all values of λ , the analysis matrix is of the same form.

Because of the flexibility of BLOCK, multiple multi-copy bases can be studied at the same time, and the dynamics and analysis matrices need to be adjusted correspondingly. The environmental atoms can also be further partitioned so that their contributions to the free energy can be calculated separately.

3.2.3. Simulation of Multi-copy Structures

For each multi-copy structure we create in **Section 3.2.1**, the following simulation protocol is used.

A short minimization is needed in order to resolve the potential bad contacts caused by the introduction of the multi-copy base pair. We use 100 steps of Steepest Descent and 100 steps of Adopted Basis Newton-Raphson for this purpose. Then the system is heated from 50 to 350 K over a linear ramp for 15 ps at a speed of 20 K per ps. Then the system is equilibrated at 350 K for 15 ps. After that, a linear ramp is used to cool the system down to 300 K at a speed of -10 K per ps for 5 ps. The system is then equilibrated at 300 K for 65 ps. This heat-cool-cycle is similar to

the annealing process, except that we only heat the system up to 50 K above room temperature. The current force field is still expected to be reasonable in describing the system. The purpose of this heat-cool-cycle is to help the new multi-copy structure overcome the energy barriers that could trap the structure in the conformation favorable only to the native structure (*see Note 9*). Finally, a 100 ps production run is done (*see Note 10*), during which the system configuration is saved at every 0.5 ps. A time step of 1 fs is used. SHAKE is again applied to constrain bonds involving hydrogen atoms.

During the simulation, we use `IMAGE` to describe the cubic-shaped periodic boundary condition. The `BLOCK` matrix of [Eq. \[6\]](#) is used for force and energy evaluation. We assume that the density of the system is well-equilibrated over 1.5 ns simulation ([Section 3.1.4](#)). So the box size is fixed here using the final box size from the 1.5 ns equilibration, and the NVT ensemble is run. After the production run is finished, we examine all the saved configurations to calculate the free energy change using the `BLOCK` analysis matrix of [Eq. \[7\]](#).

The saved configurations in the trajectory might be correlated among adjacent frames. To correct for this effect, we use [Eq. \[8\]](#) in estimating the sampling error.

$$E = \sigma \left[\left(\frac{1}{N} \right) \frac{1 + \epsilon}{1 - \epsilon} \right]^{1/2} = \left[\sum_{f=1}^N \frac{(x^2 - \bar{x}^2)}{N(N-1)} \frac{1 + \epsilon}{1 - \epsilon} \right]^{1/2}, \quad [8]$$

where E is the estimated error of the free energy change ΔG , x represents ΔG value at each frame, ϵ is the correlation between adjacent frames, f is the frame number from 1 to N (total number of frames), and σ is the standard deviation of ΔG for all frames. Systematic and statistical errors that could exist in the simulation and free energy calculations are summarized in [Notes 9](#) and [10](#).

3.2.4. Tournament Approach

According to [Eq. \[1\]](#), two free energy calculations (one for the complex and one for the DNA duplex) are required to obtain the relative binding free energy for a single base pair change as in $\Delta\Delta G = \Delta G_{\text{comp}} - \Delta G_{\text{DNA}}$. At each base pair position, we evaluate the free energy changes for three multi-copy structures for both the DNA duplex and the protein–DNA complex. We carry out three $\Delta\Delta G$ calculations as a tournament, which contains three ΔG_{DNA} and three ΔG_{comp} calculations. Two competitions for multi-copy A/T and C/G are carried out first to obtain $\Delta\Delta G_{\text{A/T}}$ and $\Delta\Delta G_{\text{C/G}}$. The two winners then compete in the second round, e.g., $\Delta\Delta G_{\text{A/C}}$ when A and C are the two winners. These three relative free energies are sufficient to describe the energy diagram of all four possible base pairs. These energies are then converted into probability and sequence logo representation using [Eqs. \[2\]](#) and [\[3\]](#).

4. Notes



1. *Correlation between adjacent base pairs.* The “additive approximation” in Section 1 might not always be valid depending on the transcription factor. We can estimate the correlation between adjacent base pairs by the following test. It is analogous to comparing the energy change caused by two separate single mutations of the DNA and the energy change caused by a double mutation of the DNA. For example, one might be interested in the correlation between positions five and six. The user will first do two separate free energy evaluations for position five and position six. Only one base pair is changed at any time. Then, the user calculates the free energy change caused by changing positions five and six simultaneously. Taking multi-copy base pair A/C as an example, we use A5C and A6C to describe the base pair change at these two positions. The non-additivity can be estimated by $\Delta\Delta G_{A5C,A6C} - (\Delta\Delta G_{A5C} + \Delta\Delta G_{A6C})$. These calculations can help quantify the non-additivity as well as the correlation between adjacent base pairs.
2. *Total computational cost and monetary equivalent.* The computational cost for obtaining the binding sites as a PWM for a transcription factor is about 400 CPU-days on a single Intel 3.0 GHz processor. The calculations in **Sections 3.1** and **3.2** are both included. We also list in **Table 2.1** the computational cost on the supercomputer BigBen at PSC. The total computational cost for the prediction of one transcription factor is about 1.2 CPU-years, or \$1,200 if we assume one CPU-year is about \$1,000.
3. *Force field and multi-copy implementations.* We compare four popular molecular modeling packages here, CHARMM, AMBER, NAMD, and GROMACS, and explain the reasons based on which we choose CHARMM in our simulations (**Section 2**).

CHARMM was the first package to be developed and has the most capabilities and functions. CHARMM and AMBER are written in FORTRAN, and the GROMACS is written in C. NAMD is developed using similar philosophy of CHARMM, but is written in C++/C. All four packages can carry out traditional molecular dynamics simulations, and lead to similar results when the same force field is used.

Many packages allow the user to choose a specific force field. The CHARMM27 force field is currently recommended for use in CHARMM. It can accurately characterize proteins and nucleic acids, and has overcome problems associated with the older versions. AMBER parm99 and parm03 are force fields

Table 2.1
Computational cost for the prediction of transcription factor–binding sites
on supercomputer BigBen at Pittsburgh Supercomputing Center

	Counter	CPU hour	CPU days
Native structures			
Protein–DNA complex, 1.5 ns equilibration		1,200	50
DNA duplex, 1.5 ns equilibration		1,200	50
Multi-copy structures			
Number of free energy evaluations per ΔG	1	160	
Number of ΔG 's per $\Delta\Delta G$	2		
Number of $\Delta\Delta G$'s per base pair in tournament	3		
Number of base pair positions	8		
Number of free energy evaluations per protein	48	7,680	320
Total cost per protein		10,080	420

recommended for use in AMBER. However, the A-DNA form tends to be over-stabilized in these force fields (13, 14). GROMACS uses OPLS force field for all-atom simulations that leads to good results for proteins but is less characterized for nucleic acids. NAMD allows the user to choose whether they want to use CHARMM, AMBER, or GROMACS force fields.

All these force fields use pairwise additive energy functions, typically including the bond length, bond angle, dihedral angle, van der Waals, and electrostatic interaction terms. Two library files are used for the implementation of the force field. The topology library file contains the list of these terms, whereas the parameter library file contains the force constants and other relevant constants.

The most important factor that leads us to choose CHARMM is its “dual-topology” implementation. AMBER and GROMACS support only “single-topology”, which means that if a “mutational” free energy perturbation is to be carried out, the two end points (reactant and product) must be similar in structure and number of atoms. In practice, they typically differ in only a small functional group (15). This poses serious challenges for the perturbations of two groups of varying number of atoms. For instance, one might be interested in finding the free energy change associated with morphing an A = T base pair into a T = A base pair along a linear coupling path. For this mutation, the total numbers of atoms in the two end

states are the same. However, because the atom types and parameters are very different for bases adenine and thymine, this morphing and free energy calculation was difficult for us to implement in AMBER and, presumably, GROMACS. One possible solution is to introduce a common intermediate topology, calculate two free energy changes of the two end states morphing into the intermediate, and then calculate the sum of the two to obtain the total free energy change. As we have already mentioned in **Note 9**, free energy calculations have a large innate systematic error, and we decided against using two “single-topology” simulations to mimic a single “dual-topology” calculation.

In contrast, both CHARMM and NAMD support “dual-topology”. However, NAMD only supports free energy perturbation for “mutational” reactions, which is generally less accurate than thermodynamic integration. This is because in the free energy perturbation formula (9, 10), the free energy change is obtained as the ensemble average of the exponential of the energy function of the system. If we assume these energy function evaluations are Gaussian-distributed, which is often true, then only one of the tails of the Gaussian curve will contribute to the final free energy change, since all the other energy values nearly contribute nothing to the ensemble average of the exponentials. However, given the same trajectory, if we use thermodynamic integration, then all these configurations will contribute to the final free energy change. A second factor is that, to our knowledge, NAMD only permits single mutations. In CHARMM, the BLOCK module allows us to carry out simulations of multiple mutations at the same time, which could lead to significant computational saving.

4. *Parallelization.* CHARMM, as pointed out in **Section 2**, does not scale very well in parallel. However, inefficient parallelization is at best a minor concern for our study, because the calculations we have described in **Section 3.2** are trivially parallelizable by running a free energy calculation at each base pair on a different node. For a binding site of length eight, 48 free energy evaluations are required to obtain the relative binding free energies at all eight base pair positions (see **Table 2.1** second column). The only exception to the trivial parallelization is the initial long equilibration (for 1.5 ns, **Section 3.1.4**) for generating configurations of the native protein–DNA complex and DNA duplex. If parallel runs are to be planned, we advise a short benchmark to be done first to establish the optimal number of processors for each system of different size. For the protein–DNA complexes we have studied (about 25,000 atoms in total), we

found that the optimal number of processors was eight on the supercomputer BigBen at PSC, and the 1.5 ns equilibrations typically take about 6 days.

5. *Addition of water box as solvent.* In **Section 3.1.2**, water box is added to the dry protein–DNA complex or DNA duplex. A CHARMM script file written by Lennart Nilsson can be used to add a small box of water with a maximal number of water molecules of 9,999. A modified script file written by Davit Hakobyan can be used to add larger boxes of water exceeding 10,000 water molecules. Both script files assume periodic boundary condition. These script files can be downloaded from the “Script Archive” on the “CHARMM Discussion Forum” (<http://www.charmm.org/ubbthreads/ubbthreads.php?Cat=0>). The TIP3P water model is used in these scripts.

For periodic boundary conditions, there are a variety of available box shapes to choose when adding water molecules using the above-mentioned scripts. Since we rely on the BLOCK module, which in turn requires the IMAGE module of CHARMM, to carry out free energy simulation and analysis, we use the cubic box shape, which is supported by IMAGE. It is also possible to use other more spherical-like box shapes, such as truncated octahedron, but it requires the creation of the corresponding IMAGE file by the user.

6. *Other treatments of counter-ions.* Two simple strategies are listed here, which avoid running long equilibrations for the ions in the system (**Section 3.1.4**). First, the system can be studied without counter-ions as a non-neutral system. This means that **Section 3.1.3** can be bypassed. The Ewald summation and spherical cutoff methods for electrostatic interactions are still valid in non-neutral systems. However, for certain molecular systems, salt concentration is an important factor for structural stability. In this case, both positive and negative ions should be added in order to obtain the desired salt concentration. Second, one can use a simple uniform neutralizing background to achieve neutral system. This is typically achieved by setting the $k=0$ term in the Ewald sum to zero (this term is automatically zero for a charge-neutral system). Simulations with a uniform neutralizing background may require modifications to be made to the standard CHARMM source code.
7. *Generation of structures with multi-copy bases.* There are two types of files that must be created for the study of multi-copy structures in **Section 3.2**: PDB and an extended topology library file. We explain the method for creating PDB files with multi-copy bases in this section. The extended topology file is explained in **Note 8**.

First of all, a library of all 2-base multi-copy PDB files is made. There are several ways of doing this. We use the standard base geometry in Ref. (12) to create PDB files for each base. These base geometries do not contain the backbone geometry or hydrogen atoms. One can then use CHARMM to read this PDB file and use “IC BUILD” and HBUILD routines to create a complete PDB file for each DNA nucleotide. Note that CHARMM’s default nucleotides are for RNA, so patches need to be applied to convert them into DNA nucleotides. After the set of PDB files are prepared for the four DNA nucleotides, the atomic entries for the base atoms are concatenated to form the 2-base multi-copy PDB files. We use the following shorthand for multi-copy bases, e.g., A/C represents the multi-copy base of changing adenine to cytosine in the leading (1st) strand of the DNA (C/A is not needed as it is simply the reverse reaction of A/C). There are six files needed for describing all possible 2-base multi-copies that constitute the library: A/C, A/G, A/T, C/G, C/T, and G/T.

Second, a fully equilibrated native DNA duplex or protein–DNA complex structure is modified to create all possible multi-copy structures for each base pair position. For a 10-base pair DNA, there are 60 multi-copy structures. We developed a C++ program to replace the original base pair by one multi-copy base pair from the above-mentioned library. Three rotations are required to align the *N*-glycosidic bond, then align the base atoms to preserve Watson-Crick base-pairing arrangement, and finally align the original plane of the base with the new multi-copy plane. For the complementary strand, the complementary multi-copy base is used so that proper base pairing is achieved.

8. *Topology files for multi-copy bases.* The multi-copy bases of the previous section are not yet integrated in the CHARMM27 topology files (“top_all27_prot_na.rtf”). The user needs to create topology entries for the six multi-copy bases (**Note 7**) and append them to the original library file. The interested users can consult CHARMM27’s topology library file, “top_all27_prot_na.rtf”, which is distributed with the package, to learn the proper naming conventions CHARMM uses for protein and nucleic acids.

The lines starting with “ATOM” in the PDB file are used by CHARMM to define the 3D coordinate of each atom. However, PDB files do not specify which atom is bonded with which one. The topology library file contains the information of the bonding arrangement and connectivity of each monomer unit (amino acids for proteins and nucleotides for DNAs), so that all the bonds, angles, and dihedral angles can be included in the evaluations of the

force and energy. Therefore, it is of paramount importance that the topology of the molecular system is properly built.

For each nucleotide in the topology library file, “top_all27_prot_na.rtf”, there are the following sections of information: the atom types, the atomic charges, the bonds that connect the atoms, the hydrogen bond donor and acceptor atoms, and the internal coordinates required for adding missing hydrogen atoms and side chains for “IC BUILD” and HBUILD. Since all the entries of the nucleic acid nucleotides share identical backbone section (phosphate and sugar group), only the entries corresponding to the base atoms need to be combined to form the multi-copy base section. All the sections that correspond to base atoms need to be combined. The hydrogen bond sections are necessary if the HBOND module of CHARMM is to be used for hydrogen bond analysis.

An important addition to the multi-copy topology library file is the non-bonded exclusion section between atoms of the two bases in the multi-copy. For example, if A/C multi-copy is made, the atom section of the topology file must specify that the base atoms of the cytosine do not have any non-bonded (including electrostatic and van der Waals) interactions with the adenine base atoms.

As bond angles and dihedral angles are not explicitly listed in the topology files, the keyword “SETUP” is needed for generating them in CHARMM. This step will add one unwanted bond angle and four unwanted dihedral angles between the two bases in the multi-copy. So the keyword “DISCONNECT” should be used for these two bases, which will remove the unwanted angles from future force and energy evaluation. Using this method, the user will also need to append a few fictitious force field parameters to the standard parameter file (“par_all27_prot_na.prm”) for the unwanted angles. The force constant values do not matter, as they are removed from the force and energy evaluation by the “DISCONNECT” step.

9. *Systematic error.* As we can see from the Introduction, the relative binding free energy of a protein with two different DNA sequences is usually small. This creates a problem if the systematic and statistical errors of the calculation are larger than the relative energy difference we want to calculate. Statistical errors can be overcome by running longer simulations to collect independent data values for analysis. Systematic error is still a hard problem and there is no sound solution for its complete removal. Systematic error in molecular dynamics

simulation and free energy calculation is typically a result of poor sampling of the entire conformational space. It may also be due to biases in the molecular force field. Sufficient sampling of alternative favorable conformations of the protein and DNA is necessary. However, because these macromolecular systems contain tens of thousands of atoms and huge number of degrees of freedom, the entire conformational space is combinatorially large. This rugged energy surface often presents energy barriers between adjacent local minima, possibly limiting the sampling space. The heat-cool-cycle step we use in **Section 3.2.3** is an attempt to overcome local energy barriers.

For protein–DNA complexes, a problem that could cause insufficient sampling is the long-lived hydrogen bonds between protein and DNA bases. The hydrogen bonds formed with the DNA backbone generally do not contribute to the binding specificity, unless the backbone geometry is highly dependent on the base identity. If there is a particular hydrogen bond that exists between a protein residue and a DNA base pair throughout the simulation of the native complex, one must closely examine what is the fate of this hydrogen bond in the multi-copy complex structures. Since the multi-copy base pair is larger and needs more space, a prior stable hydrogen bond might become unstable due to strong van der Waals repulsion, and that part of the configurational space will no longer be sampled, leading to a bias in the calculations. This can also be true if there is a persistent and stable water-mediated hydrogen bond between the protein and the DNA. For such cases, other force field choices might need to be explored, such as the “soft core potential” that tones down van der Waals repulsion to allow bulky groups in a crowded space.

10. *Statistical error.* The duration of the production run during which trajectory frames are saved for future free energy analysis is important. Good statistics can in general be achieved by running a sufficiently long production. However, the ensemble average we want to calculate **Eq. [5]** converges at about 100 ps (**Section 3.2.3**), indicating that longer productions than that will lead to the same free energy results. This production duration might be different for different systems. Therefore, it is important that the users examine the convergence of the ensemble average to reach a good compromise of convergence and statistical significance.

Acknowledgments

LAL acknowledges funding from the Department of Energy (DE-FG0204ER25626). JSB acknowledges funding from NSF CAREER 0546446, NIH/NCRR U54RR020839, and the Whitaker foundation. We acknowledge a starter grant and an MRAC grant of computer time from the Pittsburgh Supercomputer Center, MCB060010P, MCB060033P, and MCB060056N.

References

1. Pabo CO, Sauer RT. Transcription factors: structural families and principles of DNA recognition. *Annu Rev Biochem* 1992, 61:1053–1095.
2. Tuerk C, Gold L. Systematic evolution of ligands by exponential enrichment: RNA ligands to bacteriophage T4 DNA polymerase. *Science* 1990, 249(4968):505–510.
3. Ren B, Robert F, Wyrick JJ, et al. Genome-wide location and function of DNA binding proteins. *Science* 2000, 290(5500):2306–2309.
4. Mukherjee S, Berger MF, Jona G, et al. Rapid analysis of the DNA-binding specificities of transcription factors with DNA microarrays. *Nat Genet* 2004, 36(12):1331–1339.
5. Morozov AV, Havranek JJ, Baker D, Siggia ED. Protein-DNA binding specificity predictions with structural models. *Nucleic Acids Res* 2005, 33(18):5781–5798.
6. Paillard G, Lavery R. Analyzing protein-DNA recognition mechanisms. *Structure (Camb)* 2004, 12(1):113–122.
7. Endres RG, Schulthess TC, Wingreen NS. Toward an atomistic model for predicting transcription-factor binding sites. *Proteins* 2004, 57(2):262–268.
8. Schneider TD, Stephens RM. Sequence logos: a new way to display consensus sequences. *Nucleic Acids Res* 1990, 18(20):6097–6100.
9. Leach A. *Molecular Modelling: Principles and Applications*, 2nd ed. Prentice Hall, Harlow, England; New York, 2001.
10. Frenkel D, Smit B. *Understanding Molecular Simulations: From Algorithms to Applications*, 2nd ed. San Diego: Academic Press, 2002.
11. Berman HM, Westbrook J, Feng Z, et al. The protein data bank. *Nucleic Acids Res* 2000, 28(1):235–242.
12. Olson WK, Bansal M, Burley SK, et al. A standard reference frame for the description of nucleic acid base-pair geometry. *J Mol Biol* 2001, 313(1):229–237.
13. Cheatham TE, III, Young MA. Molecular dynamics simulation of nucleic acids: successes, limitations, and promise. *Biopolymers* 2000, 56(4):232–256.
14. Mackerell AD, Jr. Empirical force fields for biological macromolecules: overview and issues. *J Comput Chem* 2004, 25(13):1584–1604.
15. Kollman P. Free energy calculations: applications to chemical and biochemical phenomena. *Chem Rev* 1993, 93:2395–2417.

Chapter 3

Inferring Protein–Protein Interactions from Multiple Protein Domain Combinations

Simon P. Kanaan, Chengbang Huang, Stefan Wuchty, Danny Z. Chen, and Jesús A. Izaguirre

Abstract

The ever accumulating wealth of knowledge about protein interactions and the domain architecture of involved proteins in different organisms offers ways to understand the intricate interplay between interactome and proteome. Ultimately, the combination of these sources of information will allow the prediction of interactions among proteins where only domain composition is known. Based on the currently available protein–protein interaction and domain data of *Saccharomyces cerevisiae* and *Drosophila melanogaster* we introduce a novel method, Maximum Specificity Set Cover (MSSC), to predict potential protein–protein interactions. Utilizing interactions and domain architectures of domains as training sets, this algorithm employs a set cover approach to partition domain pairs, which allows the explanation of the underlying protein interaction to the largest degree of specificity. While MSSC in its basic version only considers domain pairs as the driving force between interactions, we also modified the algorithm to account for combinations of more than two domains that govern a protein–protein interaction. This approach allows us to predict the previously unknown protein–protein interactions in *S. cerevisiae* and *D. melanogaster*, with a degree of sensitivity and specificity that clearly outscores other approaches. As a proof of concept we also observe high levels of co-expression and decreasing GO distances between interacting proteins. Although our results are very encouraging, we observe that the quality of predictions significantly depends on the quality of interactions, which were utilized as the training set of the algorithm. The algorithm is part of a Web portal available at <http://ppi.cse.nd.edu>.

Key words: Domain combinations, set cover, protein interaction prediction.

1. Introduction

Contemporary proteome research attempts to elucidate the structure, interactions, and functions of the proteins that constitute cells and organisms. Large-scale methods determine the molecular

interactions and unravel the complex web of protein–protein interactions in single-cellular organisms such as *Helicobacter pylori* (1) and *Saccharomyces cerevisiae* (2–7). Most recently, attention focused on the first protein–protein interaction maps of complex multicellular organisms such as *Caenorhabditis elegans* (8, 9) and *Drosophila melanogaster* (10).

Such experimental results provide the basis for theoretical considerations that focus on the prediction of potential protein–protein interactions. Pioneering methods drew on the observation that interacting protein domains tend to combine into a fusion protein (11, 12) in higher organisms. Another method utilizes the observation that proteins having matching phylogenetic profiles strongly tend to be functionally linked (13, 12). The domain architecture of interacting proteins offers a framework (14) for assessing the potential presence of a particular interaction by clustering protein domains, depending on sequence and connectivity similarities. Another approach estimates the maximum likelihood of domain interaction (15, 16). Further ideas include overrepresented domain signatures (17), graph-theoretical methods (18), and other probabilistic approaches (19). Support vector machines also were employed to predict potential interactions by modeling network motifs that exhibit higher reliability of the underlying protein–protein interactions (20).

2. Materials

2.1. Protein–Protein Interactions

The first comprehensive, albeit weakly overlapping protein–protein interaction maps of *S. cerevisiae* have been provided with the yeast-two-hybrid method (2, 4). Currently, there exists a variety of yeast-specific protein–protein interaction databases. Most of them, such as MINT (21), MIPS (22), and BIND (23), collect experimentally determined protein–protein interactions. These databases lack an assessment of the data’s quality. In contrast, the GRID database, a compilation of BIND, MIPS, and other data sets, as well as the DIP database (24), provides sets of manually curated protein–protein interactions in *S. cerevisiae*. The majority of DIP entries are obtained from combined, non-overlapping data mostly obtained by systematic two-hybrid analyses. Here, we used a compilation of yeast interactions that have been evaluated by a logistic regression method, allowing the assessment of 47,773 interactions among 4,627 proteins (25). Similarly, experimentally determined interactions in *D. melanogaster* were evaluated, allowing for 20,047 interactions among 6,996 proteins (10).

2.2. Protein Domain Data

The advent of fully sequenced genomes of various organisms has facilitated the investigation of proteomes. The *Integr8* database (<http://www.ebi.ac.uk/integr8>) has been set up to provide comprehensive statistical and comparative analyses of complete proteomes of fully sequenced organisms. The initial version of the application contains data for the genomes and proteomes of 182 sequenced organisms (including 19 archae, 150 bacteria, and 13 eukaryotes) and proteome analysis derived through the integration of UniProt (26), InterPro (27), CluSTR (28), GO/GOA (29), EMSD, Genome Reviews, and IPI (30). In particular, we utilized IPI (International Protein Index) files of Yeast, which provide full annotations of each protein with its corresponding domains. In particular, we elucidated the domain architecture of the corresponding proteins by focusing on PFAM domain information as of the corresponding IPI files (31).

2.3. Microarray Data and Co-expression Correlation Coefficients

Genes with similar expression profiles are likely to encode interacting proteins (32, 33). We assess MSSC's ability to predict pairs of potentially interacting yeast proteins (Section 3.5), by utilizing the gene expression data of *S. cerevisiae* and *D. melanogaster*. By downloading 1,051 expression profiles of Yeast and 157 of fly from the Stanford Microarray Database (SMD, <http://genome-www5.stanford.edu>), we calculated the Pearson's correlation coefficient r_P for each pair of interacting proteins. Provided there exist data for both proteins over m time points, the Pearson correlation coefficient is calculated by

$$r_P = \frac{\frac{1}{m} \sum_{i=1}^m x_i y_i - \bar{x} \bar{y}}{\sigma_i \sigma_j}, \quad [1]$$

where \bar{x} and \bar{y} are the sample means, and σ_i and σ_j are the standard deviations of i and j .

2.4. GO Annotation Data and GO Distance

For any two interacting proteins, we calculate an annotation-based distance between proteins, taking into account all Gene Ontology terms (29) (GO, <http://www.geneontology.org>) that are common to the pair and terms which are specific to each protein. Any two proteins can have several shared GO terms (common terms) and a variable number of terms specific for each protein (specific terms). This distance between interacting proteins i and j is based on the Czekanowski-Dice formula (34):

$$d_{i,j} = \frac{|T_{GO}(i) \Delta T_{GO}(j)|}{|T_{GO}(i) \cup T_{GO}(j)| + |T_{GO}(i) \cap T_{GO}(j)|}. \quad [2]$$

In this formula, T_{GO} are the sets of the proteins with associated GO terms while $|T_{GO}|$ stands for their number of elements, and Δ is the symmetrical difference between two sets. This distance

formula emphasizes the importance of the shared GO terms by giving more weight to similarities than to differences. Consequently, for two genes that do not share any GO terms the distance value is 1, while for two proteins sharing exactly the same set of GO terms the distance value is 0.

3. Methods

3.1. General Outline of the Protein Cover Problem

Investigations of the three-dimensional protein structure suggest that the fundamental unit of protein structure is a domain. Independent of neighboring sequences, this region of a polypeptide chain folds into a distinct structure and mediates the proteins' biological functionality. A domain can also be defined as an amino acid sequence motif with an associated function. Largely, proteins contain only one domain (35) while the majority of sequences from multicellular eukaryotes appear as multidomain proteins of up to 130 domains (36).

We identify proteins in the proteome that give rise to protein interactions through the selection of domain combinations that explain the known protein interaction network. In the simplest case, we use only a selected set of domain pairs in a training set of protein interactions $R = (P_R, E_R)$, where P_R is the set of proteins, and E_R defines a set of edges between proteins if and only if they interact with each other. The protein interactions R induce a set of domain pairs $D_R = \{(d_i, d_j)\}$ where the domains d_i and d_j belong to the proteins involved in the interactions E_R . Schematically, we show these relations in **Fig. 3.1a**. The protein-protein cover problem thus arising is to choose an "optimal" subset of domain pairs $D_R, D \subseteq D_R$, such that D covers all the interactions in R (**Section 3.5**).

3.2. Domain Combinations

We conceptualize domains as the driving force behind the formation of protein interactions. Since the vast majority of proteins in single cellular organisms carry a single domain, domain pairs are sufficient to explain the presence of a protein interaction. However, in more complex organisms the number of multidomain proteins increases. Indicating that protein interactions might be also facilitated by multidomain interactions, we also allow D_R to include combinations of interacting domains that are potentially involved in a given protein interaction. As such, we handle domain combinations in our framework as new "domains." Assuming that P_1 has domains d_1, d_2 , and d_3 , we label d_1d_2, d_1d_3, d_2d_3 , and $d_1d_2d_3$ as "new" domains (**Fig. 3.1b**). Depending on the complexity of the proteins, we might only want to look at combinations up to a certain number of domains.

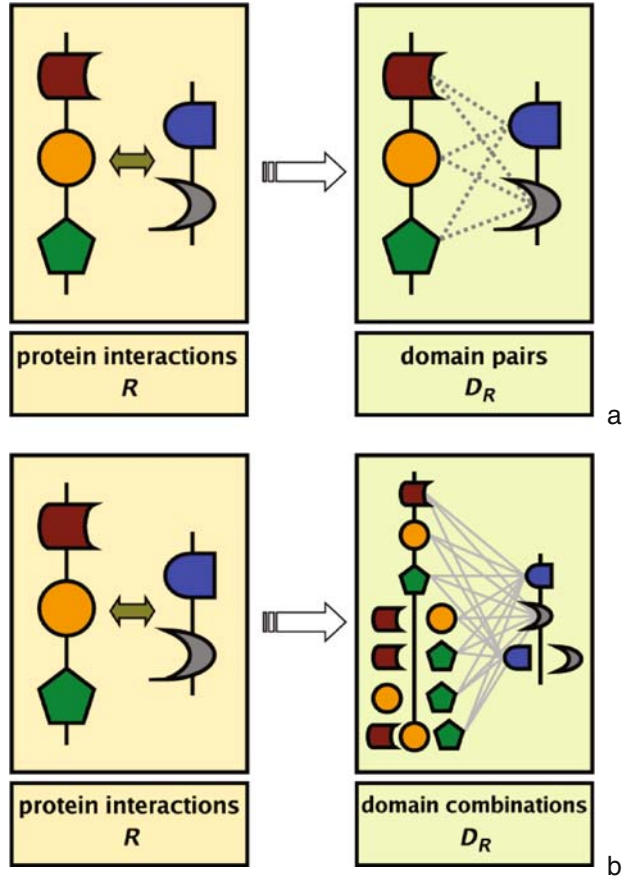


Fig. 3.1. **Combinatorics of protein interactions.** (a) In the simplest case, we consider protein domain pairs as the driving force behind the formation of protein interactions. (b) In a more sophisticated way, we also consider combination of domain pairs that potentially can give rise to observed protein interactions.

3.3. Previous Prediction Methods

3.3.1. Association Method

The set of domain pairs D obtained from a training set of interactions among proteins for which the domain architectures are known can be utilized in different ways to predict protein–protein interactions. The association method (17) assigns an interaction probability $P(d_m, d_n)$ to each domain pair (d_m, d_n) in D_R (such that $D_R = D$) by

$$P(d_i, d_j) = \frac{I_{ij}}{N_{ij}}, \tag{3}$$

where I_{ij} is the number of interacting protein pairs that contain (d_i, d_j) , and N_{ij} is the total number of protein pairs that contain (d_i, d_j) . The interaction probability for each putative interaction between pairs of proteins is calculated using

$$P(P_i, P_j) = 1 - \prod_{(d_m, d_n) \in (P_i, P_j)} (1 - P(d_m, d_n)). \tag{4}$$

3.3.2. Maximum Likelihood Estimation (MLE)

The maximum likelihood estimation method (15) assumes that two proteins interact if at least one pair of domains of the two proteins interacts. Under the above assumption, any protein pair (P_i, P_j) is the same as the one used in our protein–protein cover problem, Eq. [4]. So, the maximum likelihood is

$$L = \prod P(O_{ij} = 1)^{O_{ij}}(1 - P(O_{ij} = 1))^{1-O_{ij}} \quad [5]$$

where

$$O_{ij} = \begin{cases} 1 & \text{if } (P_i, P_j) \in E_R, \\ 0 & \text{otherwise.} \end{cases} \quad [6]$$

The likelihood L is a function of $\theta(P(d_i, d_j), f_p, f_n)$, where $P(d_i, d_j)$ represents the probability that domains d_i and d_j interact while f_p and f_n indicate fixed rates of false positive and false negative interactions in the underlying network. Note that in both the Association Method (AM) and the Maximum-Likelihood-Estimation (MLE), domain pairs were utilized to predict potential protein interactions.

3.4. Transformation of Protein Network to Set Cover Problem

Suppose X is a finite set and \mathcal{F} is a family of subsets of X that can cover X , i.e., $X \subseteq \bigcup_{S \in \mathcal{F}} S$. The set-cover problem is to find a subset C of \mathcal{F} to cover X ,

$$X \subseteq \bigcup_{S \in C} S, \quad [7]$$

where C is also required to satisfy certain conditions according to different specific problems. For example, the minimum exact set-cover problem requires that $\sum_{S \in C} |S|$ is minimized, allowing for a C with minimum cardinality $|C|$ (37, 38). The minimum set-cover problem is NP-complete. The set-cover problem can be generalized for our purposes by putting X into a bigger set \mathcal{Y} (Fig. 3.2a). Suppose \mathcal{Y} is a finite set, $X \subseteq \mathcal{Y}$ and F is a family of subsets of \mathcal{Y} that can cover X , i.e., $X \subseteq \bigcup_{S \in F} S$. Thus, the generalized set-cover problem is to find a subset C of F to cover X ,

$$X \subseteq \bigcup_{S \in C} S, \quad [8]$$

where C is again constrained to certain problem-specific conditions. This theoretical framework allows us to conceive protein interactions as a set-cover problem. As already mentioned, protein–protein interactions can be modeled as a graph $P_R = (P, E)$, where P is the set of proteins and E is the set of edges between two proteins if and only if they interact with each other. A set-cover problem is set up from the protein–protein interaction network P_R by taking

$$\mathcal{Y} = \{\text{all protein pairs } (P_i, P_j) | P_i, P_j \in P_R\},$$

$$X = \{\text{protein pairs } (P_i, P_j) | (P_i, P_j) \in E_R\},$$

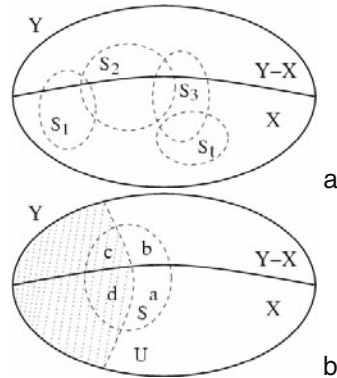


Fig. 3.2. **Schematic representations of set cover problems.** (a) Here, we show the generalized set cover problem: X is a subset of Y , and $F = \{S_i, 1 \leq i \leq t\}$ is a family of subsets of Y . (b) MSSC chooses a set S with the minimum $\frac{b+c}{a}$. The greedy algorithm for MSSC allows overlapping of subset inside X . The overlap actually increases the interaction probability for a protein pair.

and F to be the set of all domain pairs (d_m, d_n) , where (d_m, d_n) is contained by at least one element of X . A domain pair (d_m, d_n) is considered as a subset of \mathcal{Y} . Specifically, if a protein pair (P_i, P_j) (an element in X) contains (d_m, d_n) , then (P_i, P_j) belongs to the subset (d_m, d_n) .

Suppose we find a subset C of F to cover every element (P_i, P_j) in X where an element in C corresponds to a domain pair (d_m, d_n) . If (d_m, d_n) covers (P_i, P_j) , then the two proteins P_i and P_j contain d_m and d_n , respectively. Since (d_m, d_n) can be used to cover the interaction between P_i and P_j , we also have a set of domain pairs to cover the protein network P_R . Suppose there is a set D of domain pairs to cover the network P_R . For every element (P_i, P_j) in X , there is a domain pair (d_m, d_n) from D to cover the interaction between P_i and P_j . Since (d_m, d_n) can be viewed as an element in F , the collection C of all the domain pairs from D is a subset of F , and C covers X .

In this transformation, the set of protein–protein interactions P_R corresponds to the set X that needs to be covered, and a domain pair corresponds to an element in F (a subset of \mathcal{Y}).

Once a set cover that fulfills these criteria is found, sets of protein domain interactions allow a description and explanation of the underlying protein interactions to the best extent. Such pairs of proteins can be scanned if an interaction among their domains is actually present in the set cover and therefore is a potential candidate for a putative protein interaction.

3.5. MSSC Approach

Many ways exist that allow choices of domain pairs which cover the protein–protein interactions in a training set. AM simply uses all the possible domain pairs to explain the protein–protein

interaction network, i.e., it uses F to cover X , so the resulting specificity is very low (15). Sometimes, we are only interested in using a subset of domain pairs to cover the protein–protein interaction network, and hopefully the subset is chosen so that both specificity and sensitivity are maximized. So, the MSSC problem is to find a subset C of F to cover X such that

$$m(C) := \sum_{S \in C} |S - X| \quad [9]$$

is minimized.

MSSC allows the subset C to cover the overlap with X , but the overlap with $\mathcal{Y} - X$ (outside X) is minimized. MSSC chooses a cover in this way to maximize the specificity because the false positives appear only in $\mathcal{Y} - X$. In developing a greedy algorithm for MSSC (**Fig. 3.3**) at each step, it chooses a subset whose ratio between the part outside X and the part inside U , $(|S - X|)/(|S \cap U|)$, is minimized (**Fig. 3.2b**).

The number of iterations of the while loop is bounded by $\min(|X|, |F|)$ where each single iteration takes $(O|X||F|)$ time; so the time complexity of this greedy algorithm is $\mathcal{O}(|X||F| \min(|X|, |F|))$. If we apply appropriate data structures, it can be realized in $O(\log |F| \sum_{S \in F} |S|)$ time. In particular, we maintain a bipartite graph between the elements in \mathcal{Y} and the elements in F . If the former is contained by the latter, we add an

```

GREEDY_MSSC( $Y, X, \mathcal{F}$ )
   $U \leftarrow X$ 
   $\mathcal{E} \leftarrow \mathcal{F}$ 
   $\mathcal{C} \leftarrow \emptyset$ 
  while  $U \neq \emptyset$ 
    do pick  $S \in \mathcal{E}$  with the minimum  $\frac{|S-X|}{|S \cap U|}$ 
      (a tie is broken by  $|S \cap U|$ )
       $U \leftarrow U - \{S\}$ 
       $\mathcal{E} \leftarrow \mathcal{E} - \{S\}$ 
       $\mathcal{C} \leftarrow \mathcal{C} \cup \{S\}$ 
  return  $\mathcal{C}$ 

```

Fig. 3.3. Pseudocode of the greedy algorithm for solving the Maximum Specificity Set Cover (MSSC) problem. In this representation, we describe the basic steps of the greedy algorithm, which allows us to find a set cover with maximum specificity. In particular, the routine *GREEDY_MSSC*(Y, X, F) receives X , the set of actual interactions, Y the set of non existing interactions, and F , a family of possible families of protein domain interactions that cover X . In every step of the algorithm a family S from E is chosen, which minimizes the ratio between the cover between S and X and the intersection between S and the already covered set of interactions U . This optimization procedure allows us to obtain an optimal collection of families of domain interactions that allow the largest possible cover of interactions.

edge between them, so there are $\sum_{S \in F} |S|$ edges. Furthermore, we store all elements in F into a heap ordered by $\frac{|S-X|}{|S \cap U|}$. When a subset S is selected, it is excluded from our problem. We update the bipartite graph and the heap accordingly. The bipartite graph will not be updated more than $\sum_{S \in F} |S|$ times totally. For a single S , the updating of the heap takes $|S| \log |F|$. Therefore, the total time is $O(\sum_{S \in F} |S| + \sum_{S \in F} |S| \log |F|)$, which is $O(\log |F| \sum_{S \in F} |S|)$. If $|F|$ is very large, we use an array of $|X|^2$ instead of a heap to store F , resulting in a time which is $O(|X|^2 + \sum_{S \in F} |S|)$.

The greedy algorithm only allows an approximation. Its solution has the following relationship with the optimal solution of MSSC

Theorem Suppose C_a is the approximation of MSSC found by the above greedy algorithm, and C_o is an optimal subset for MSSC. Let $k = \max_{S \in F} |S|$. If $m(C_o) = 0$, then $m(C_a) = 0$; otherwise, we have

$$\frac{m(C_a)}{m(C_o)} \leq \lceil \ln(k-1) + 1 \rceil. \quad [10]$$

The proof for this theorem can be found in (39). Since k is the maximum number of elements a subset can have, it corresponds to the maximum number of protein pairs that contain a domain pair in the protein network. Therefore, this theorem indicates that the difference between the approximated and the exact solution remains small, if the maximal number of protein interactions covered by a domain pair is kept small too (i.e., we want to get rid of “promiscuous” domain pairs).

The MSSC procedure, which also accounts for multidomain interactions, allows us to predict putative protein–protein interactions in *S. cerevisiae* and *D. melanogaster*. We observe that our algorithm clearly outscores previous methods such as the Association method (AM) and Maximum Likelihood Estimation (MLE) in terms of sensitivity and specificity. We also observe that our algorithm increases the quality of predictions by using a carefully selected training set of protein interactions. As such we observe that a curated set of yeast and fly protein interactions, which aims to evaluate each interaction with a confidence score, can increase the quality of predictions drastically. Indeed, we observe that our predictions correlate significantly with elevated levels of co-expression and low GO distances, a strong indication for the quality of our predictions.

3.6. Comparison of the Performance of MSSC to Other Algorithms

Since it was used in the original paper (15), we use the combined data set of Uetz et al. (4) and Ito et al. (2), which allows a direct performance comparison of AM, MLE, MSC, MSSC, and MSSC₂. We choose MSSC₂, the MSSC version that accounts for up to two domain combinations, since we did not find a significant

improvement when going to greater than two domain combinations with the current data. However, this might not be the case for more complex data sets.

We use all interactions in the aforementioned dataset as training and test sets. As for information about the domain architecture of yeast proteins, we utilized PFAM domain data (31) as laid down in the Integr8 database. This induces an overfitting, but results in disjoint training and testing sets are qualitatively similar. We measure the prediction accuracy by specificity, the ratio of the number of matched interactions between the predicted interaction set, I , and the testing set, T , over the total number of predicted interactions, $SP = \frac{|I \cap T|}{|I|}$. As quality parameters, we define sensitivity as the ratio of the number of matched interactions between the predicted set, I , and the testing set, T , over the total number of observed interactions, $SN = \frac{|I \cap T|}{|T|}$.

In **Fig. 3.4**, we observe that MSSC – the implementation of our algorithm that accounts for domain pairs only – outperforms AM in terms of both specificity and sensitivity drastically. While MSSC in general allows for results that are very similar to MLE, we observe that MSSC generates significantly more results in areas of high specificity.

Apart from MSSC, we also tried the minimum set cover (MSC), utilizing domain pairs. MSC uses different criteria to choose the subset C from F so that C has minimum cardinality $|C|$ (38, 39). Compared to MSSC, MSC chooses fewer domain

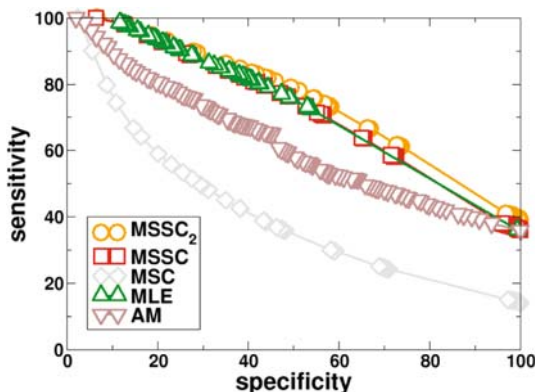


Fig. 3.4. Performance comparison of different algorithms. We compare the performance of the Association Method (AM) (17), Maximum Likelihood Estimation (MLE) (15), Minimum Set Cover (MSC) (38), Maximum Specificity Set cover (MSSC), and its version that uses pairwise domain combinations (MSSC₂). As training and testing set we utilize a combined set of interactions retrieved from yeast-two-hybrid compilations of Uetz et al. (4) and Ito et al. (2). We observe that MLE and MSSC share the same prediction characteristics while MSSC₂ allows the best predictions. On the other hand, MSSC and MSSC₂ clearly outscore AM and MSC.

pairs to cover the protein–protein interaction network, but actually covers more false positives, as indicated by the comparably low values of sensitivity and specificity. In **Fig. 3.4**, we also observe that an implementation of MSSC that accounts for interactions between up to two domain combinations (MSSC₂) slightly outperforms MSSC. Note that the solution for MSSC is a subset of MSSC₂ since both use the same algorithm, while MSSC₂ gives the algorithm more choices in order to obtain more accurate predictions.

3.7. Results with High-Quality Interactions

Currently available sets of protein–protein interactions contain startling rates of false positives (~50%) and false negatives (~90%) (40). However, there exist a variety of ways to circumvent this problem. One of the most reliable ways to assess the quality of interactions is to integrate different sources interactions to increase their reliability. In particular, training a logistic regression model with parameters such as co-expression, topological and protein interaction related data allows a prediction of an interaction’s reliability quantified by a confidence value (25). Here, we utilize such evaluated interaction data sets of the organisms *S. cerevisiae* (25) and *D. melanogaster* (10), combining 4,627 yeast proteins and 47,783 interactions and 6,996 fly protein, which are involved in 20,047 interactions. All of these interactions are evaluated by a confidence value.

In general, proteomes are composed by a majority of proteins that carry one domain. However, in more complex organisms, proteins carry more than one domain, suggesting that domain combinations are putatively important for protein interactions. In **Fig. 3.5**, we present the sensitivity/specificity curves we obtained by applying MSSC and MSSC₂ to the curated sets of yeast and fly protein interactions. In particular, we utilized sets of increasing reliability (as measured by the threshold of the confidence value c) of yeast (25) and fly (10), allowing us to obtain sensitivity/sensitivity curves of predictions by considering these sets as trainings as well as testing set. In general, we observe that both yeast and fly protein interaction sets of increasing reliability outscore the corresponding curves obtained with sets of lower quality. In particular, our results suggest that predicting interactions with MSSC₂ slightly outperforms the results obtained with MSSC, indicating the role of protein domain combinations for the underlying interactome. In the following, we predict protein interactions in yeast and fly by the application of the MSSC₂ algorithm.

To evaluate the quality of our predictions, we analyze the distributions of co-expression correlations. In **Fig. 3.6**, we observe that training sets containing high confidence interactions indeed allow a significant shift of distributions of co-expression correlation coefficients. In both cases, yeast and fly predictions show significant different means of their distributions when

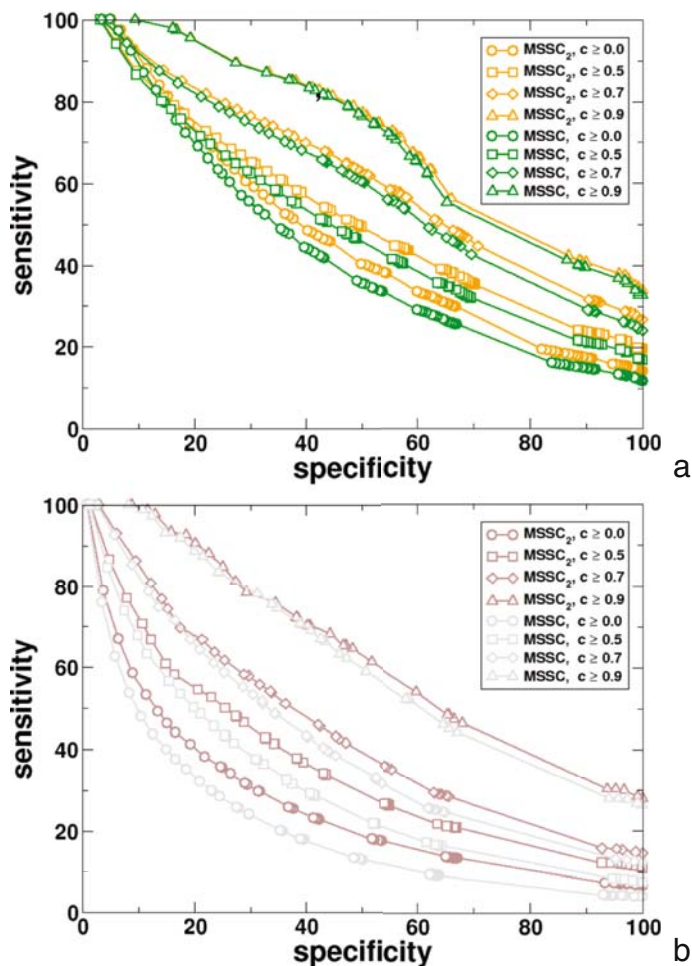


Fig. 3.5. **Sensitivity/specificity curves of predictions in yeast and fly protein interactions sets.** We predicted interactions by feeding the MSSC and MSSC₂ algorithms with protein interactions sets of increasing reliability (c). We obtained sensitivity/specificity curves by considering these sets as trainings as well as testing sets. **(a)** In general, we observe that both **(a)** yeast and **(b)** fly protein interaction sets of increasing reliability allow us to obtain sensitivity and specificity values that outscore the corresponding curves obtained with sets of lower quality. In particular, our results suggest that predicting interactions with MSSC₂ slightly outscore the results obtained with MSSC, indicating the dominating role of domain combinations for the underlying interactome.

compared to a background distribution (which is defined as the full set of uncurated yeast (25) and fly interactions (10)). Significant Student's t -test scores support our conclusion that high-quality interactions allow a higher degree of quality predictions.

As a different measure for the existence of a predicted interaction, we utilize the empirical observation that interacting proteins show a significantly elevated tendency to share similar functions.

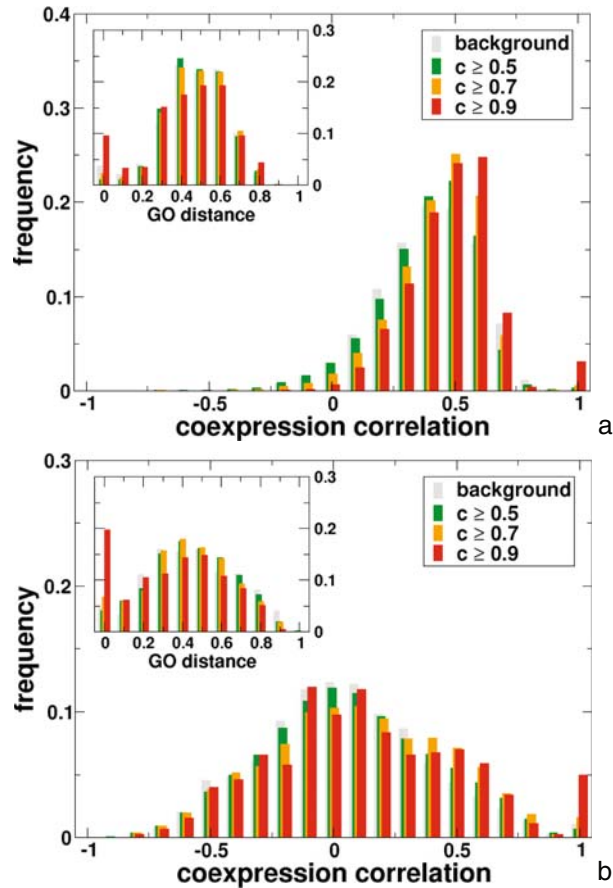


Fig. 3.6. **Co-expression and GO distance analysis of the predictions in yeast and fly.** Compared to a comprehensive set of co-expression correlations values of protein links (background), we observe that in **(a)** yeast and **(b)** fly the quality of the underlying protein interaction network increases the quality of our predictions. The shift toward higher coexpression correlation values is further supported by significant Student's *t*-test scores, when testing the curves of the predictions to a background distribution being the full set of uncurated yeast and fly interactions. [$c \geq 0.5$: 2.61 ($P = 9.1 \times 10^{-3}$), $c \geq 0.7$: 41.66 ($P < 10^{-10}$), $c \geq 0.9$: 36.26 ($P < 10^{-10}$)] and fly [$c \geq 0.5$: 4.20 ($P = 2.6 \times 10^{-5}$), $c \geq 0.7$: 10.53 ($P < 10^{-10}$), $c \geq 0.9$: 8.81 ($P < 10^{-10}$)]. As a different indicator of the existence of a potential interaction we show the GO distance for yeast **(a)**, inset) and fly **(b)**, inset). Similarly to the distributions of co-expression coefficients we find that an increasing quality of the training sets allows qualitatively better predictions as exemplified by the shifts toward lower values. These results are further supported by significant Student's *t*-test scores when compared to the background distributions of yeast [$c \geq 0.5$: 3.85 ($P = 1.2 \times 10^{-4}$), $c \geq 0.7$: 3.10 ($P = 2.0 \times 10^{-3}$), $c \geq 0.9$: 5.94 ($P = 2.9 \times 10^{-9}$)] and fly [$c \geq 0.5$: 1.02 ($P = 0.31$), $c \geq 0.7$: 2.88 ($P = 4.0 \times 10^{-3}$), $c \geq 0.9$: 6.27 ($P = 5.9 \times 10^{-10}$)].

In turn, a measure that represents functional similarity might be used as an indicator of an interaction's existence. Utilizing GO annotations (29), we calculate a GO distance (see Materials) for every predicted interaction. If two proteins do not share any GO

terms, the distance value is 1, while the opposite holds for proteins sharing exactly the same set of GO terms. Indeed, in the insets of **Fig. 3.6**, we observe that both yeast and fly predictions, that were obtained with high-quality training sets show a stronger functional similarity. Again, in both cases distributions have significantly different means as exemplified by significant Student's *t*-test scores, indicating that high-quality interactions indeed influence the quality of the predictions.

4. Conclusions

In this paper, we present a novel algorithm that allows the selection of a set of domain pairs, which covers the experimental observations and maximizes the specificity in the training set of protein interactions. Compared to previous methods, MSSC is able to improve the specificity for a given sensitivity. As a refinement of this algorithm we also introduced the opportunity to model and predict interactions as the consequence of interactions among many combinations of domains. In particular, we observe that the relatively small amount of multidomain proteins in yeast compared to fly already have a significant impact on the interactions of the underlying interactome. As such, we observe that we obtain better results by applying MSSC₂, our algorithmic extension that accounts for domain combinations.

5. Notes



1. *Dependence from training data.* Our results also suggest that the quality of predicted protein–protein interactions strongly depends on the utilized training sets. Although we showed that our algorithm by design reduces the amount of false positives, it allows only high-quality interactions if the training set reflects an elevated degree of quality.

The dependence on high-quality interaction sets also poses a sometimes intricate problem. The increase in quality always is accompanied by a decrease of protein interactions and therefore limits the number of interacting proteins involved. Thus, the number of protein domains that allow these interactions is diminished as well. Since protein interactions and domains are the major data sources of our algorithm, the choice of an

appropriate training set that balances quality and a reasonable number of domains, that still allows predictions on a large scale is a crucial step.

2. *Outlook.* The proposed algorithm can be used to predict protein interactions for every organism for which data of protein interactions and the corresponding protein domain architectures are available. Encouraged by the high quality of our results, a next step is the prediction of potential interactions between proteins in organisms where high-quality interaction data are available to train MSSC₂. Furthermore, proteins that participate in many interactions are preferentially conserved and change their sequence only to a small extent (41, 42). The observation that high clustering and co-expressed protein–protein interaction sets show preferential evolutionary conservation (43, 44), and increased reliability (18) allows us to not only obtain a high-quality set of interactions potentially serving as the basis for predictions in a reference organism. In fact, such sets of interactions may also indicate evolutionary cores that have been conserved more generally among different organisms. As such they not only allow the evaluation of protein–protein interactions but also could serve as a training set for the predictions of protein interactions in target organisms.

Acknowledgments

Danny Chen was supported in part by the NSF under Grant CCF-0515203. Jesús Izaguirre was supported by partial funding from NSF grants IOB-0313730, CCR-0135195, and DBI- 0450067. Stefan Wuchty was supported by the Northwestern Institute of Complexity (NICO).

References

1. Rain JC, Selig L, DeReuse H, Battaglia V, Reverdy C, Simon S, Lenzen G, Petel F, Wojcik J, Schächter V, Chemama Y, Labigne A, Legrain P. The protein–protein interaction map of *Helicobacter pylori*. *Nature* 2001, 409, 211–215.
2. Ito T, Tashiro K, Muta S, Ozawa R, Chiba T, Nishizawa M, Yamamoto K, Kuhara S, Sakaki Y. Towards a protein–protein interaction map of the budding yeast: A comprehensive system to examine two-hybrid interactions in all possible combinations between the yeast proteins. *Proc Nat Acad Sci USA* 2000, 97, 1143–1147.
3. Ito T, Chiba T, Ozawa R, Yoshida M, Hattori M, Sakaki Y. A comprehensive two-hybrid analysis to explore the yeast protein interactome. *Proc Nat Acad Sci USA* 2001, 98, 4569–4574.
4. Uetz P, Giot L, Cagney G, Mansfield T, Judson R, Knight J, Lockshorn D, Narayan V, Srinivasan M, Pochart P, Qureshi-Emili A, Li Y, Godwin B, Conover D, Kalbfleisch T, Vijayadamodar G, Yang M, Johnston M,

- Fields S, Rothberg J. A comprehensive analysis of protein–protein interactions of *Saccharomyces cerevisiae*. *Nature* 2000, 403, 623–627.
5. Gavin A, Bösch M, Krause R, Grandi P, Marzioch M, Bauer A, Schultz J, Rick J, Michon AM, Cruciat CM, Remor M, Böfert C, Schelder M, Brajenovic M, Ruffner H, Merino A, Klein K, Hudak M, Dickson D, Rudi T, Gnau V, Bauch A, Bastuck S, Huhse B, Leutwein C, Heurtier MA, Copley R, Edelmann A, Querfurth E, Rybin V, Drewes G, Raida M, Bouwmeester T, Bork P, Seraphin B, Kuster B, Neubauer G, Superti-Furga G. Functional organization of the yeast proteome by systematic analysis of protein complexes. *Nature* 2002, 415, 141–147.
 6. Ho Y, Gruhler A, Heilbut A, Bader G, Moore L, Adams SL, Millar A, Taylor P, Bennett K, Boutillier K, coauthors. Systematic identification of protein complexes in *Saccharomyces cerevisiae* by mass spectrometry. *Nature* 2002, 415, 180–183.
 7. Jeong H, Mason S, Barabási AL, Oltvai Z. Lethality and centrality in protein networks. *Nature* 2001, 411, 41–42.
 8. Walhout A, Sordella R, Lu X, Hartley J, Temple G, Brasch M, Thierry-Mieg N, Vidal M. Protein interaction mapping in *C. elegans* using proteins involved in vulval development. *Science* 2000, 287, 116–122.
 9. Li S, Armstrong C, Bertin N, Ge H, Milstein S, Boxem M, Vidalain PO, Han JD, Chesneau A, Ha T, et al. A map of the interactome network of the metazoan *C. elegans*. *Science* 2004, 303, 540–543.
 10. Giot L, Bader J, Brouwer C, Chaudhuri A, Kuang B, Li Y, Hao Y, Ooi C, Godwin B, Vitols E, Vijayadamar G, Pochart P, Machineni H, Welsh M, Kong Y, Zerhusen B, Malcolm R, Varrone Z, Collis A, Minto M, Burgess S, McDaniel L, Stimpson E, Spriggs F, Williams J, Neurath K, Ioime N, Agce M, Voss E, Furtak K, Renzulli R, Aanensen N, Carroll S, Bickelhaupt E, Lazovatsky Y, DaSilva A, Zhong J, Stanyon C, Finley R Jr, White K, Braverman M, Jarvie T, Gold S, Leach M, Knight J, Shinkets R, McKenna M, Chant J, Rothberg J. A protein interaction map of *Drosophila melanogaster*. *Science* 2004, 302, 1727–1736.
 11. Enright A, Iliopoulos I, Kyrpides N, Ouzounis C. Protein interaction maps for complete genomes based on gene fusion events. *Nature* 1999, 402, 86–90.
 12. Marcotte E, Pellegrini M, Thompson M, Yeates T, Eisenberg D. A combined algorithm for genomewide prediction of protein function. *Nature* 1999, 402, 83–86.
 13. Pellegrini M, Marcotte E, Thompson M, Eisenberg D, Yeates T. Assigning protein functions by comparative genome analysis: Protein phylogenetic profiles. *Proc Natl Acad Sci USA* 1999, 96, 4285–4288.
 14. Wojcik J, Schächter V. protein–protein interaction map inference using interacting domain profile pairs. *Bioinformatics* 2001, 17, 296S–305S.
 15. Deng M, Mehta S, Sun F, Cheng T. Inferring domain–domain interactions from protein–protein interactions. *Genome Res* 2002, 12, 1540–1548.
 16. Iossifov I, Krauthammer M, Friedman C, Hatzivassiloglou V, Bader J, White K, Rzhetsky A. Probabilistic inference of molecular networks from noisy data sources. *Bioinformatics* 2004, 20, 1205–1213.
 17. Sprinzak E, Margalit H. Correlated sequence–signature as markers of protein–protein interaction. *J Mol Biol* 2001, 311, 681–692.
 18. Goldberg D, Roth F. Assessing experimentally derived interactions in a small world. *Proc Natl Acad Sci USA* 2003, 100, 4372–4376.
 19. Tong A, Drees B, Nardelli G, Bader G, Brannetti B, Castagnoli L, Evangelista M, Ferracuti S, Nelson B, Apoluzzi S, et al. A combined experimental and computational strategy to define protein interaction networks for peptide recognition modules. *Science* 2002, 295, 321–324.
 20. Albert I, Albert R. Conserved network motifs allow protein–protein interaction prediction. *Bioinformatics* 2004, 20, 3346–3352.
 21. Zanzoni A, Montecchi-Palazzi L, Quondam M, Ausiello G, Helmer-Citterich M, Cesari G. Mint – a molecular interaction database. *FEBS Lett.* 513, 2002, 135–140.
 22. Mewes HW, D Frishman UB, Mannhaupt G, Mayer K, Mokrejs M, Morgenstern B, Munsterkotter M, Rudd S, Weil B. MIPS: A database for genomes and protein sequences. *Nucl Acids Res* 2002, 30, 31–34.
 23. Bader G, Donaldson I, Wolting C, Ouellette B, Pawson T, Hogue C. BIND – The biomolecular interaction network database. *Nucl Acids Res* 2001, 29, 242–245.
 24. Xenarios I, Salwinski L, Duan X, Higney P, Kim SM, Eisenberg D. Dip, the database of interacting proteins: A research tool for studying cellular networks of protein interactions. *Nucl Acids Res* 2002, 30, 303–305.

25. Bader J, Chaudhuri D, Rothberg J, Chant J. Gaining confidence in high-throughput protein interaction networks. *Nature Biotech* 2004, 22, 78–85.
26. Apweiler R, Bairoch A, Wu C, Barker W, Boeckmann B, Ferro S, Gasteiger E, Huang H, Lopez R, Magrane M, Martin M, Natale D, O'Donovan C, Redaschi N, Yeh L. UniProt: The universal protein knowledgebase. *Nucl Acids Res* 2004, 32, D115–D119.
27. Mulder N, Apweiler R, Attwood T, Bairoch A, Barrell D, Bateman A, Binns D, Biswas M, Bradley P, Bork P, Bucher P, Copley R, Courcelle E, Das U, Durbin R, Falquet L, Fleischmann W, Griffiths-Jones S, Haft D, Harte N, Hulo N, Kahn D, Kanapin A, Krestyaninova M, Lopez R, Letunic I, Lonsdale D, Silventoinen V, Orchard S, Pagni M, Peyruc D, Ponting C, Selengut J, Servant F, Sigrist C, Vaughan R, Zdobnov E. The InterPro database, 2003 brings increased coverage and new features. *Nucl Acids Res* 2003, 31, 315–318.
28. Kriventseva E, Fleischmann W, Zdobnov E, Apweiler R. CluSTR: A database of clusters of SWISS-PROT+TrEMBL proteins. *Nucl Acids Res* 2001, 29, 33–36.
29. Consortium G. The gene ontology (go) database and information resource. *Nucl Acids Res* 2004, 32, D258–D261.
30. Kersey P, Duarte J, Williams A, Apweiler R, Karavidopoulou Y, Birney E. The international protein index: An integrated database for proteomics experiments. *Proteomics* 2004, 4, 1985–1988.
31. Bateman A, Coin L, Durbin R, Finn R, Hollich V, Griffiths-Jones S, Khanna A, Marshall M, Moxon S, Sonnhammer E, Studholme D, Yeats C, Eddy S. The PFAM protein families database. *Nucl Acids Res* 2004, 32, D138–D141.
32. Grigoriev A. A relationship between gene expression and protein interactions on the proteome scale: Analysis of the bacteriophage t7 and the yeast *Saccharomyces cerevisiae*. *Nucl Acids Res* 2001, 29: 3513–3519.
33. Ge H, Ziu L, Church G, Vidal M. Correlation between transcriptome and interactome mapping data from *Saccharomyces cerevisiae*. *Nat Genet* 2001, 29, 482–486.
34. Martin D, Brun C, Remy E, Mouren P, Thieffry D, Jacq B. GOToolBox: Functional analysis of gene datasets based on gene ontology. *Genome Biol.* 2004, 5, R101.
35. Doolittle R. The multiplicity of domains in proteins. *Ann Rev Biochem* 1995, 64, 287–314.
36. Li WH, Gu Z, Wang H. Evolutionary analyses of the human genome. *Nature* 2001, 409, 847–849.
37. Johnson DS. Approximation algorithms for combinatorial problems. *J Comput System Sci* 1974, 9, 256–278.
38. Cormen TH, Leiserson CE, Rivest RL, Stein C. *Introduction to Algorithms*, Second Edition. McGraw Hill Boston, MA, 2001.
39. Huang C, Morcos F, Kanaan S, Wuchty S, Chen D, Izaguirre J. Predicting protein–protein interactions from protein domains using a set cover approach. *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 2007, 4, 78–87.
40. von Mering C, Krause R, Snel B, Cornell M, Oliver S, Fields S, Bork P. Comparative assessment of large-scale data sets of protein–protein interactions. *Nature* 2003, 31, 399–403.
41. Wuchty S. Topology and evolution in the yeast protein interaction network. *Genome Res* 2004, 14, 1310–1314.
42. Fraser H, Hirsh A, Steinmetz L, Scharf C, Feldman M. Evolutionary rate in the protein interaction network. *Science* 2002, 296, 750–752.
43. Wuchty S, Oltvai Z, Barabási AL. Evolutionary conservation of motif constituents within the yeast protein interaction network. *Nat Genet* 2003, 35, 176–179.
44. Wuchty S, Barabási AL, Ferdig M. Stable evolutionary signal in a yeast protein interaction network. *BMC Evol Biol.* 2006, 6, pp. 8.

Chapter 4

Prediction of Protein–Protein Interactions: A Study of the Co-evolution Model

Itai Sharon, Jason V. Davis, and Golan Yona

Abstract

The concept of molecular co-evolution drew attention in recent years as the basis for several algorithms for the prediction of protein–protein interactions. While being successful on specific data, the concept has never been tested on a large set of proteins. In this chapter we analyze the feasibility of the co-evolution principle for protein–protein interaction prediction through one of its derivatives, the correlated divergence model. Given two proteins, the model compares the patterns of divergence of their families and assigns a score based on the correlation between the two. The working hypothesis of the model postulates that the stronger the correlation the more likely is that the two proteins interact. Several novel variants of this model are considered, including algorithms that attempt to identify the subset of the database proteins (the homologs of the query proteins) that are more likely to interact. We test the models over a large set of protein interactions extracted from several sources, including BIND, DIP, and HPRD.

Key words: Protein–protein interactions, co-evolution, mirror-tree.

1. Introduction

Protein–protein interactions are at the core of numerous basic reactions that make up complex biological processes. The detection of these interactions can help to better understand the molecular machinery of the cell and expose biological processes and pathways that have not been characterized so far. Existing technologies enable researchers to detect interactions on a genomic scale and have triggered studies that explore large networks of known protein interactions in search of interesting subnetworks, complexes, and regular patterns (1–4).

Traditionally, protein–protein interactions have been deduced via conventional wet-lab experimental methods, such as the yeast two-hybrid (Y2H) system (5) and mass spectrometry (2). While these are high-throughput technologies, they tend to be very expensive and time-consuming. Moreover, their error rate is high, especially with the Y2H method, with many false negatives (due to the cellular localization of the proteins, post-translational modifications, misfolding of the recruited proteins, or steric hindrances) and high percentage of false positives (6). Other experimental techniques (such as affinity chromatography and co-immunoprecipitation) are more accurate but are low-throughput methods.

In this view, there is a strong interest in tools that can reliably predict the existence of protein–protein interactions. This problem has received a considerable attention in the past several years, and many methods to predict interactions were developed (see next section for a survey). Here we focus on the co-evolution model and test it extensively. As opposed to other methods that look for recurring patterns in proteins that are known to interact, the co-evolution model is based on a concrete biological hypothesis, namely interacting proteins evolve in coordination. The goal of our computational experiments is to test the validity and extent of this hypothesis and to determine how successful it is in discerning interacting from non-interacting proteins.

We start with a review of the computational methods that are used to *predict* interactions. We then proceed to discuss the co-evolution model in detail. To assess co-evolution and predict protein–protein interactions, we test several variants of the mirror-tree method and different strategies to identify the subset of proteins within two given protein families that are more likely to interact.

1.1. Prediction of Protein–Protein Interactions: Survey

The field of protein–protein interaction prediction via computational methods is relatively new but very active. The methods can be mainly classified into four different approaches: studies that use structural information, relational data mining inference, co-evolution analysis, and hybrid approaches.

1.1.1. Protein Structure– Based Approaches

Methods using protein tertiary structure have had some success (7, 8), but are relatively slow. Protein docking methods use known tertiary structures to predict the most probable binding site between two structures. As this problem is NP hard, conventional methods rely on local search heuristics. For example, the method of Espadaler et al. (9) uses residue patches that characterize protein interfaces to search for proteins containing similar patches. The interface patches are determined by analysis of residue contacts in complexes whose structure is known. Other methods use known hydrogen bond potentials, charge potentials, and

interaction site energy minimization techniques. Most of these approaches, however, do not take into consideration possible protein deformations that can change the binding properties at the time of interaction. Methods that do allow such flexible docking structures are sometimes computationally infeasible (10). Furthermore, structure-based approaches fail to distinguish between two proteins that have the biochemical potential of interaction, and two proteins that physiologically interact. Lastly, these methods are limited to proteins of known structures, which account for less than 5% of all known proteins and 30% of the known protein families.

Protein threading methods can extend predictions to sequences of unknown structures. Lu et al. use a multi-threading approach in their “Multiprospector” algorithm (11). The method threads sequences in a library of monomer structures that are known to participate also as part of dimer structures. If two sequences have significant signal with respect to two chains that are part of the same complex, the sequences are re-aligned (considering the other sequence in the template structure), the energy between the interacting residues is computed, and if the z-score of the dimer is significant compared to that of the monomers, the sequences are predicted to interact. One of the main advantages of the method is that beyond predicting the interaction itself, it also predicts the interaction site. However, it is limited to solved complexes of interacting proteins, of which only a few are known in the protein data bank.

1.1.2. Relational Data Mining Inference

These approaches are among the most successful in the field of computational protein–protein interaction inference. Kini and Evans’ earlier work showed a correlation between interaction binding sites and proline residues: these residues are 2.5 times more likely in these areas (12). Aytuna et al. (13) utilized information about protein interfaces and hotspot residues in interacting pairs (residues that contribute most of the binding energy for the interaction (14)), for deducing possible interactions in other pairs that share similarity with the interacting protein interfaces. It has been suggested that the presence of a few hotspots may be a characteristic of most interactions (15). Extending this analysis, Sprinzak and Margalit (16) provided a framework for identifying protein–protein interactions through the analysis of over-represented sequence signatures. Utilizing also the information on known three-dimensional structures of interacting proteins, InterPreTS characterizes possible interaction sites between two proteins (17). Other algorithms attempt to detect protein–protein interactions through the interactions between domains (18, 19). In the algorithm described in (20), each protein is represented as a vector in the domain space. Using a machine learning technique known as random forests (21), in which many randomly generated

decision trees are combined through majority voting, the authors classify each pair of proteins as either interacting or non-interacting. Han et al. (22) propose an algorithm that uses information from multiple domains to predict interactions.

Other relational methods trace evolutionary events that might hint at the existence of interaction. Marcotte et al. (23) note the relation between protein interaction and gene locality through a technique termed “domain fusion prediction”. Enright et al. present a refinement over typical domain fusion prediction methods, suggesting that interacting proteins can also be the result of a “composite” protein evolving into “component” proteins (24).

Relational methods have had some success, but mostly for interactions of similar nature, either with the same signatures or when there is an evidence of gene fusion/decomposition events in sequence databases. Clearly, such evolutionary circumstances are not the case for all protein–protein interactions.

1.1.3. Co-evolution Analysis

Co-evolution approaches include gene preservation correlation, phylogenetic tree topology comparison, and correlated mutation approaches. Gene preservation approaches are very simple: if two proteins interact to perform a vital biological function, then both proteins will be passed on during speciation (24). Many interactions are conserved across species, in particular interactions with functions such as protein translation, ribosomal structure, DNA binding, and ATP metabolism (25). This approach was used for the prediction of potential protein–protein interactions in the POINT database (26). Sun et al. (27) proposed the phylogenetic-profile method that is based on the assumption that interacting proteins are inherited together during speciation events due to strong selective pressure. Thus, these proteins are expected to have similar phylogenetic profiles composed of those organisms from a reference set in which their homologs are present. Other gene preservation approaches consider locality constraints among protein subdomains: subdomains will tend to have the same relative position in interacting proteins (28).

Phylogenetic tree topology methods compare homologs of interacting proteins (i.e., protein families) and their phylogenetic trees: if the two trees are very similar (termed **mirror trees**), then it is assumed the proteins have co-evolved and possibly interact (29, 30). Mirror-tree-based methods gained popularity in recent years. The reported results seem to be promising, and already led to the development of tools and web servers (31, 32). However, while there are several examples of interactions that follow this model, it is unclear how successful it is in predicting interactions and distinguishing correlation due to interaction and co-evolution, from correlation due to similar evolutionary trees in general. Moreover, the approach may suffer from several drawbacks. One problem occurs when the tested families contain proteins from close

organisms. The distance matrices in that case are expected to contain many small values, regardless of the mutation rates of the tested families. When the Pearson correlation (or any other correlation measure) is used in that case, the true signal is likely to be disturbed by noise or a few outliers, which may produce meaningless results. Another drawback lies in the assumption that all members of both families interact, which may not be true in the general case, in particular when considering several paralogs from the same organism. In addition, the model depends on correct construction of the phylogenetic tree, which is hard to guarantee. These issues and others are addressed in this chapter.

A more localized position-specific approach is the correlated mutations method introduced by Pazos and Valencia (33). This approach searches for correlated, compensating mutations in specific positions between two candidate interacting proteins. Their results presented on structural subdomains seem fairly strong (34). They later extended this algorithm to detect interacting partners based on the ratio between intra-protein correlations and inter-protein correlations (35).

1.1.4. Hybrid Approaches

A natural progression over the previous approaches is of hybrid methods that combine different sources of information. Jansen et al. (36) propose an integrative approach that uses a Bayesian network for deciding whether two proteins are interacting, based on information from several sources including GO (37) and MIPS (38) databases. Another integrative algorithm that is based on Support Vector Machines (SVMs) has been proposed in (39). Information sources for this method include sequence similarity, homology to other interacting pairs, and relation in the GO database. The model uses several kernel functions that are constructed for the different information sources and used in conjunction with each other, to train a classifier that separates interacting from non-interacting proteins.

2. Methods I – The Basic Model

In this chapter we test the co-evolution model for inferring protein–protein interactions and present several variants of this model. Our basic assumption is similar to the co-evolution principle that was originally introduced in (40). Specifically, two proteins that interact will tend to co-evolve in a coordinated manner, resulting in a higher evolutionary correlation between their corresponding homologs. The intuition behind this premise is fairly simple; if one partner in a protein interaction pair mutates, then its

counterpart will have to adapt in order to preserve the interaction. Therefore, given two query proteins and their homologs, one can theoretically predict an interaction if there is an evidence that the groups co-evolve.

Correlated evolution approaches (such as mirror-tree) offer several advantages over other methods. First, there are several algorithms that can approximate a correlated evolution score and also run in polynomial time. Second, the idea of correlated evolution is a priori and fits in with basic biological principles. Sequence signatures represent an a posteriori approach; consequently, they can identify only the protein interactions similar to those found in a training data set. Finally, co-evolution can detect proteins that physiologically interact.

It is important to note that co-evolution does not necessarily entail physical interaction. For example, two proteins can be part of the same complex without interacting directly. However, even if the proteins do not interact with each other directly, they might still co-evolve to preserve the structural stability and functionality of the complex. To determine if the proteins actually interact, it is necessary to inspect the structure of the complex, which is usually unknown. However, from biological (functional) standpoint, proteins that are part of the same complex are often considered as interacting proteins. Here too, we do not make a distinction.

Any co-evolution method for protein–protein interaction prediction relies on the knowledge of phylogenetic trees. However, in practice, the exact evolutionary path of a specific protein is unknown; therefore, one must infer a protein’s phylogeny via careful examination of the differences between protein homologs found in different organisms. An ideal solution to this problem would be to reconstruct phylogenetic trees for each interacting protein partner and its homologs (herein referred to as a protein family) and then to compare their similarities. Unfortunately, phylogenetic tree reconstruction is provably an NP-complete problem, and existing measures for assessing co-evolution (as the Pearson correlation coefficient) attempt to avoid this problem by considering all pairs of protein homologs. However, this clearly affects the sensitivity of the method. That is one of the issues we address in this study.

Mirror-tree co-evolution-based algorithms are usually composed of the following three steps: Given two query proteins (i) identify their homologs (family members) in a common set of n organisms, (ii) construct distance matrices for the two families, and (iii) measure the similarity between the two matrices using the Pearson correlation. We start by presenting the general case (of multiple paralogs in each organism) and then discuss methods to reduce this set to a single protein from each organism, and eliminate non-interacting pairs.

2.1. Correlated Divergence

2.1.1. Definitions

We are given two query proteins q_1, q_2 and a multiple alignment of each with other, related, database sequences. The groups of database sequences are referred to as protein families F_1 and F_2 . The families consist of n_1 (n_2) sequences originated from N_1 (N_2) organisms (note that usually $N_i < n_i$). Denote the organism sets by O_1 and O_2 , respectively. We hypothesize that the query sequences interact and therefore there is an evidence of co-evolution among the two protein families.

The first step in the co-evolution algorithm is to select the subset of organisms that are common to both protein families, $O_{\text{common}} = O_1 \cap O_2$. In each organism $o \in O_{\text{common}}$, there exist at least one protein in F_1 and one protein in F_2 . Each such pair is a candidate interacting pair. Initially, we assume that of all possible candidate pairs in an organism, at least one interacts in a similar way as the two query proteins; therefore, the core set of interacting proteins consists of at least O_{common} interacting pairs. Assuming that all proteins from a family are involved in the hypothesized interaction, the maximal size of the set may be larger. Both these assumptions are revised later. With a little abuse of notation we revise our definitions of F_i , n_i and N_i to the groups of proteins that originate from the (smaller) organism set O_{common} . Denote $N = |O_{\text{common}}|$. The set of proteins from family F_i that is found in organism o is denoted by $F_{i,o}$.

2.1.2. Computing Correlated Divergence

The assumption of co-evolution for interacting proteins holds only for the interaction site; however, this information is clearly an unknown parameter when the two query proteins are only hypothesized to interact. Even for known interactions, the binding box is usually unknown (**Note 1**). However, one would expect interacting proteins to have similar divergence patterns overall. Although not necessarily constrained in a correlated manner outside of the binding box, it is assumed that both interacting proteins have similar rates of mutation. This is not true for different protein families in general, as different families have different molecular *clocks*, some exhibiting faster mutation rates than others, where rapidly changing protein might undergo significant changes in conformation. However, since the general structure of the protein has to be preserved to maintain a structurally and functionally active interaction site, it is less likely that the two interacting proteins will evolve in significantly different paces.

Our first measure attempts to detect signals of **correlated divergence** and is similar to the mirror-tree approach (29). To estimate the divergence rate, we compute the total number of amino acid mutations between all protein pairs within a given family. If two protein families co-evolve, these mutation levels should correlate, indicating similar protein clocks. For each

organism pair $o_1, o_2 \in O_{\text{common}}$ within a protein family F_i , we define the mutation level $D_i(o_1, o_2)$ to be the average number of mutations over all protein pairs in the cross product of F_{i,o_1} and F_{i,o_2}

$$D_i(o_1, o_2) = \frac{\sum_{p_1 \in F_{i,o_1}, p_2 \in F_{i,o_2}} d(p_1, p_2)}{|F_{i,o_1}| |F_{i,o_2}|},$$

where $d(p_1, p_2)$ is the average number of mutations between proteins p_1 and p_2 , normalized per 100 residues to prevent biases due to different lengths. The mutations are computed from sequence alignments. The conservation level of p_1 and p_2 is defined as $100 - d(p_1, p_2)$.

This measure was chosen as a starting point due to its simplicity and relative ease of implementation and computational speed. Moreover, this measure qualifies as a distance metric, as it is symmetric, non-negative, and satisfies the triangle inequality. This property is necessary to accurately estimate the distances between entities that are evolutionarily more distant. It is also a necessary condition for constructing a phylogenetic topology.

The correlated divergence is estimated by computing the Pearson correlation coefficient r between the mutation levels of organism pairs in family 1 and family 2:

$$r = \frac{\sum_{i=1}^N \sum_{j=i+1}^N [D_1(o_i, o_j) - \mu_1] [D_2(o_i, o_j) - \mu_2]}{\sqrt{\sum_{i=1}^N \sum_{j=i+1}^N [D_1(o_i, o_j) - \mu_1]^2} \sqrt{\sum_{i=1}^N \sum_{j=i+1}^N [D_2(o_i, o_j) - \mu_2]^2}}, [1]$$

where $\mu_k = 2 \sum_{i=1}^N \sum_{j=i+1}^N D_k(o_i, o_j) / N(N-1)$. The correlation coefficient is ranging in value from -1 (anti-correlation) to 1 (perfect correlation). Value of 0 indicates no correlation. In practice, this raw correlation score is not very robust and suffers from several drawbacks as discussed in **Section 2.1.5**. An even more stressing problem is that the method does not scale well, as the number of pairs grows quadratically with the number of organisms. If the actual subset of organisms with interacting pairs O_{interact} is much smaller than the total number of organisms (i.e., $O_{\text{interact}} \ll O_{\text{common}}$), then the ratio of interacting pairs to all pairs, $\frac{|O_{\text{interact}}|^2}{|O_{\text{common}}|^2}$, will be close to zero, making it almost impossible to detect correlation signals. This problem will be addressed in **Section 3.1**.

2.1.3. Data preparation

Naturally, any two sets of homologous proteins over the same set of organisms have the same true phylogenetic tree structure with similar distance matrices. To test if the co-evolution signal (as approximated by the correlated divergence score) is causally related to protein-protein interaction and not the result of correlation due to similar evolutionary trees in general, we compute the distribution of correlated divergence scores for a set of interacting proteins and a second set of non-interacting protein pairs and compare the two distributions. Preparation of these data sets

should be handled with care, in order to prevent statistical biases. Here we took special measures to make sure that our data sets of interacting and non-interacting proteins have similar properties and are clean from certain statistical biases that may be caused by erroneous construction.

Interacting proteins: Our data set of interacting proteins was derived from the Biozon database (41) and processed to eliminate redundancy and ensure high quality of data. We started with a set of 59,624 unique protein–protein interactions that were available as of October 2004, gathered from BIND (42), DIP (43), and HPRD. Many interactions were associated with multiple evidence codes (e.g., yeast two-hybrid and immunoprecipitation). After excluding interactions that were determined exclusively with the yeast two-hybrid test, we were left with 13,767 unique interactions that we consider to be of high quality (the break-up by method is given in Table 4.1). The data set was further pruned by excluding the following interactions from the initial set:

Table 4.1
Break-up of interactions by evidence codes. First column is the number of interactions that were verified by each method. The second is the number of interactions that were determined *only* by that method

Method	#interactions	#interactions (unique)
Immunoprecipitation	7717	5572
Tandem affinity purification (tap)	4108	3375
Affinity chromatography	1645	842
X-ray diffraction	688	394
In vitro binding	646	238
Cross linking	363	159
Gel filtration chromatography	351	69
Copurification	327	112
Biochemical	217	95
Competition binding	214	87
In vivo kinase activity	198	176
Immunoblotting	189	51
Biophysical	188	79
Gel retardation assays	144	52

(continued)

Table 4.1 (continued)

Method	#interactions	#interactions (unique)
Cosedimentation	124	33
Alanine scanning	119	44
Three-dimensional structure	118	67
Native gel electrophoresis	114	78
Genetic	113	50
Other	99	72
Electron microscopy	86	18
Experimental	71	54
Density gradient sedimentation	66	12
Surface plasmon resonance	62	22
Interaction adhesion assay	61	33
Filter overlay assay	58	11
Calcium mobilization assay	55	15
Autoradiography	51	4
Split ubiquitin system	48	38
Lambda fusion	46	26
Immunofluorescence	33	9
Transcription assay	30	21
Elisa	24	4
Monoclonal antibody blockade	22	11
Immunostaining	19	6
Phage display	18	5
Fret analysis	18	11
Transient coexpression	17	9
NMR	15	3
Fluorescence spectroscopy	9	2
Nuclear translocation assay	9	1
Chemotaxis	8	7
Not specified	7	5

(continued)

Table 4.1 (continued)

Method	#interactions	#interactions (unique)
Immunolocalization	7	0
X-ray scattering	6	0
Isothermal titration calorimetry	6	0
Mass spectrometry	3	1
Peptide spot assay	3	0
Microtiter plate binding assay	3	0
Photon correlation spectroscopy	3	0
Ion exchange chromatography	2	0
Sucrose gradient sedimentation	2	2
Neutron scattering	1	0

- Interactions in which one or both proteins have less than 20 homologs. This is necessary to ensure that enough information is available for the computation of the correlated divergence score. We also excluded proteins with more than 700 homologs.
- Interactions between homologous proteins (including self-interaction), since these bias the correlated divergence score.
- Interactions involving proteins whose homologs from other organisms are identical. In this case the correlated divergence score cannot be computed.
- Interactions between proteins whose families have less than eight organisms in common.

Finally, to minimize redundancy we picked only one pair from all pairs of homologous interactions. After applying these filtering criteria we were left with a set of 3192 interactions.

Non-interacting proteins: The construction of a non-interacting data set is a little more tricky, since no database of such proteins exists. In fact, even if such database would exist, it would have represented only a fraction of the space of protein pairs, which is not necessarily similar to the subspace represented by the data set of interacting proteins. In such cases it is impossible to know whether the results obtained are reflective of a true signal (correlated divergence in our case) or some other property that has nothing to do with it (e.g., high frequency of outliers or different number of entries in the distance matrices). Therefore, it is desirable to neutralize such irrelevant properties by choosing a data set

composed of ingredients similar to those of the data set of interacting pairs. In order to achieve this goal we constructed the data set of non-interacting proteins so that the connectivity of proteins in this set, namely the number of pairs in which each protein is involved, will be similar to the connectivity of the interacting data set as much as possible. Specifically, the pairs were constructed by pairing proteins (from the same organism) that were chosen randomly from the pool of interacting proteins according to their distribution in this data set, and after applying the same filtering criteria that were applied to the interacting set (**Note 2**). While this does not guarantee similar properties at the interaction/protein-pair level, it significantly reduces statistical biases that might affect the results. This has been verified by comparing some key properties of the interacting and non-interacting data sets that can influence the Pearson correlation, such as the distribution of $|O_{\text{common}}|$ and the distribution of the average number of paralogs for each organism in O_{common} (see **Fig. 4.1**). This procedure left us with 3117 pairs of proteins that are likely to be non-interacting (**Note 3**).

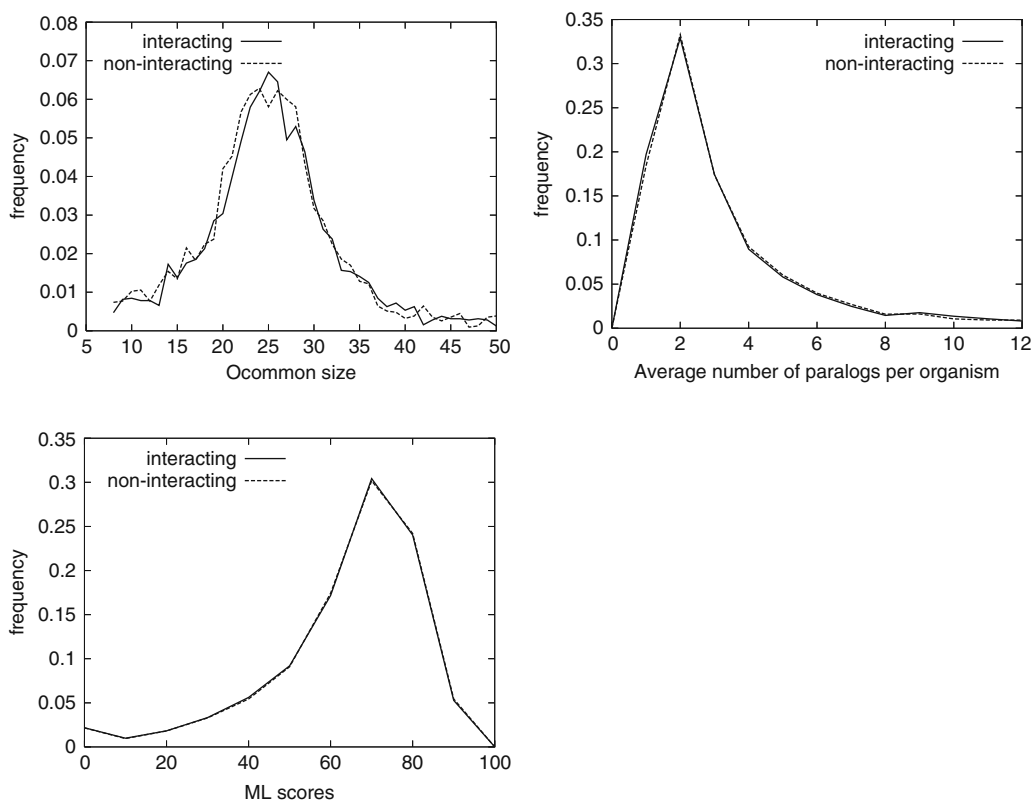


Fig. 4.1 Comparing statistics over interacting and non-interacting data sets. Size of O_{common} in interacting and non-interacting pairs (*left*), average number of paralogs per organism in families (*middle*), and distribution of ML-scores between pairs of organisms in interacting and non-interacting pairs (*right*). The distributions are almost identical, indicating that the data sets indeed bear similar statistical characteristics.

Family construction and multiple alignments: For each pair the analysis starts by generating multiple sequence alignments. To generate the alignments we collected a set of homologs for each protein from Biozon (these were generated using BLAST with e-value threshold of 0.01). Multiple sequence alignments were then constructed using MAFFT (44, 45) version 5 with the default parameters (Note 4).

2.1.4. Signals of Correlated Divergence

Our first experiment tests the plain correlated divergence model, using all organisms in O_{common} and all proteins from each organism. This setup is similar to the one usually used in other mirror-tree studies (49, 50). Figure 4.2 plots the distribution of correlated divergence scores for our data sets. Both the density and the cumulative functions are rather close to each other, with the interacting pairs being assigned a slightly higher correlated divergence scores than the non-interacting pairs, on average (see Table 4.2). Figure 4.3 displays typical scatterplots of high- and low-scoring pairs. The correlation (or lack thereof) can be easily seen in these examples.

Clearly, the distributions of Fig. 4.2 cannot be used for a reliable separation of interacting pairs from non-interacting pairs. In the next sections we discuss some of the model's drawbacks and suggest possible solutions.

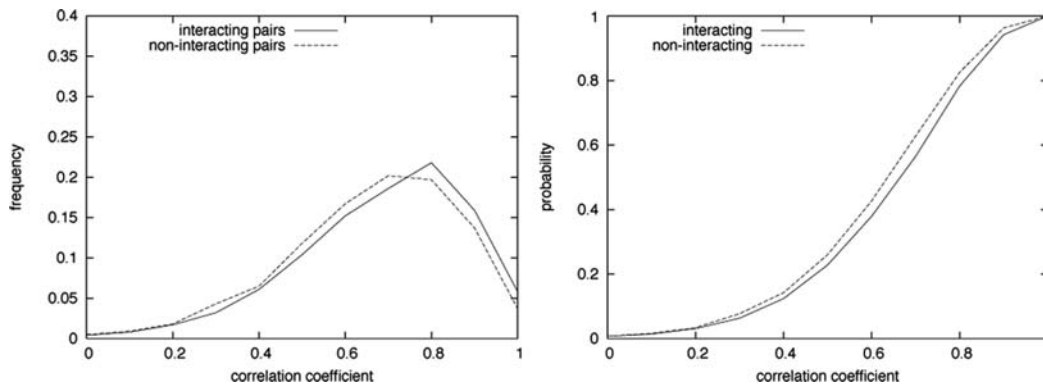


Fig. 4.2 Distribution of correlated divergence scores for interacting vs. non-interacting pairs. All organisms in O_{common} with all their proteins are used. Behavior of both data sets seems to be similar, with a slight advantage for the interacting over the non-interacting set.

Table 4.2
Simple correlated divergence statistical results

	μ	σ	Score < 0.3	Score > 0.7
Interacting pairs	0.636	0.199	0.063	0.435
Non-interacting pairs	0.612	0.199	0.077	0.372

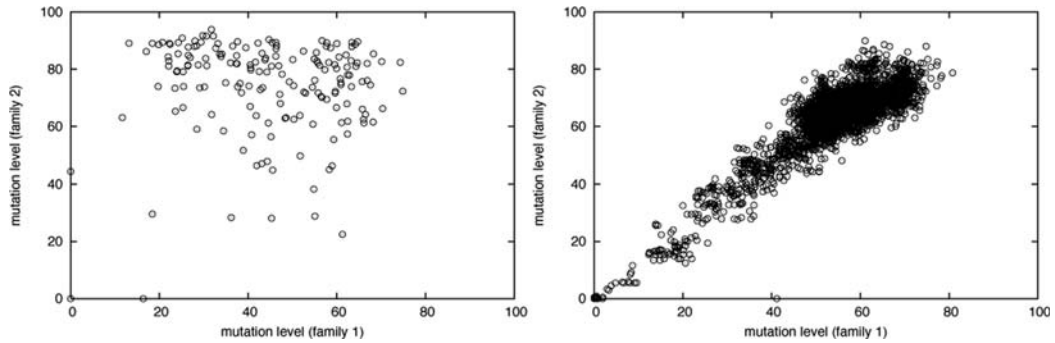


Fig. 4.3 **Correlation plots for interacting pairs with strong and weak correlated divergence scores.** Low correlated divergence score ($r = 0.03$, *left*) is assigned to the interaction between *nr1003180002190* (TATA box binding protein) and *nr112300000009* (TBP-interacting protein 120A), while the interaction between *nr1002690000168* (60S ribosomal protein L9, mitochondrial precursor YmL9) and *nr1002860000147* (60S ribosomal protein YmL6, mitochondrial precursor) shows much stronger signs of correlated divergence ($r = 0.90$, *right*). To view the Biozon profile page of a protein with *nr* identifier *nr1x*, follow the URL biozon.org/Biozon/Profile/x.

2.1.5. Drawbacks of the Pearson Correlation Measure

Although widely used in mirror-tree-based algorithms, the Pearson correlation coefficient is not a very robust test statistic and has several properties that should be taken into consideration when applying it for the specific task of detecting correlated divergence.

Uneven divergence rates. The Pearson correlation coefficient assigns high scores for data points with linear correlation, regardless of the slope of the line. This may lead to undesirable situations as presented in Fig. 4.4, in which two families are assigned high correlated

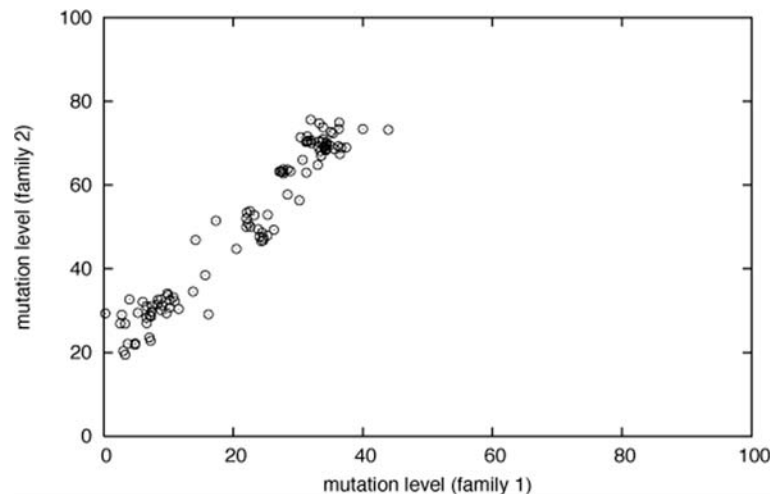


Fig. 4.4 **High Pearson correlation coefficient does not necessarily indicate correlated divergence.** The interaction between *nr1002140001532* (ribosomal protein L10) and *nr1004670000093* (presenilin 1) is assigned a high Pearson score ($r = 0.97$), but the divergence rates are different in the two families, with the second diverging twice as fast as the first one. We would expect the data points of two families whose divergence rates are in correlation due to the interaction (and not due to similar evolutionary trees in general) to be concentrated around the $y = x$ diagonal.

divergence score despite the fact that the divergence rate of one family is about twice as fast as the divergence rate of the other family. One possible solution is to measure how well the data fits the line $y = x$.

Insensitivity to the size of the data set. The Pearson score is insensitive to the number of data points used to compute the correlation. In general, we would consider correlation detected in a larger set of data points more reliable than the same correlation patterns found in a smaller set. However, the Pearson correlation measure assigns the same values for different sets showing similar signs of correlation, regardless of their size.

To address this problem we normalized the Pearson score with respect to a background distribution of correlation scores of non-corresponding proteins. This is done by randomly permuting the organism order of one of the families and recomputing r as defined above. From this distribution, the mean Pearson correlation score, \bar{r} , and standard deviation, σ , are calculated. Along with the true correlation score, r^* , the normalized z -score is calculated as:

$$z = \frac{r^* - \bar{r}}{\sigma}.$$

The z -score-based normalization mitigates the aforementioned problem, as it assigns higher scores to larger sets. However, on the other hand, we found that the size of the data set tends to dominate this measure, thus creating a bias that can mask differences between signals of correlation, either positive or negative, between the two distance matrices. Therefore, we did not use this normalization in the subsequent experiments.

The effect of outliers. Another issue with the Pearson score is that it can be heavily influenced by a few outliers. Consider the scatterplots of artificial datapoints given in **Fig. 4.5**. The plot on the left shows signs of correlation, as the datapoints fit the linear regression model quite well. The plot on the right, however, is heavily influenced by five outliers marked by black squares. When computing the Pearson correlation coefficient for both examples the results are similar ($r = 0.69$ and $r = 0.67$ for the left and right

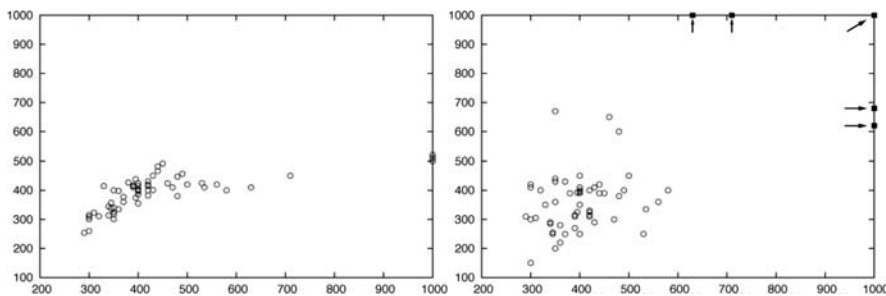


Fig. 4.5 **Instability of the Pearson correlation coefficient. Scatterplots of artificial data.** *Left:* a data set showing signs of a real correlation ($r = 0.69$). *Right:* a data set heavily influenced by a few outliers, marked with arrows ($r = 0.67$ with outliers, $r = 0.17$ without outliers).

plots, respectively). When the five outliers are removed from the right plot, however, the correlation coefficient decreases to an insignificant score of $r = 0.17$. The case of a few outliers is common when using the correlated divergence model, and is usually caused by proteins that are either very similar (almost identical) to the query proteins or by remote homologs that are weakly similar to the query proteins.

There are several approaches one can take to handle these kind of situations. One possibility is to weigh the entries based on their relative similarity, to decrease the contributions of highly similar or highly dissimilar proteins. There are quite a few weighting schemes that are commonly used when constructing profiles or HMMs from MSAs, for example (e.g., (51)). However, these methods underweight either the most similar or the most dissimilar proteins, but none of them is designed to underweight both. Therefore, we left the Pearson correlation coefficient as is.

Another possibility is to exclude outliers (above a certain threshold) from the distance matrices. Very high values in the distance matrix of one family are usually the result of comparing two distant organisms. In such cases the real co-evolution signal, if exists, is likely to be masked by noise due to the large number of mutations overall, making it almost impossible to detect signs of correlated divergence. The opposite situation in which mutation levels are too low is undesirable as well: the proteins of two close organisms probably did not diverge much, again making it difficult to detect co-evolution. In order to improve the signal, we tested a variant of the correlated diverge algorithm where all proteins that are either more than 90% or less than 30% identical to the query proteins are excluded. However, this approach did not improve the separation between the interacting and the non-interacting pairs.

3. Methods II – Improvements over the Basic Model

Although the conceptual framework presented in the previous section is clean and simple, in practice, many assumptions made are in need of revision. For example, it is not uncommon to find in a genome multiple genes that belong to the same protein family. Even if one of them interacts with another protein, there is no reason to assume that all its paralogs also interact with that protein (or its paralogs, if they exist). Moreover, the interaction might become inactive in some organisms, due to mutations after speciation; such organisms may add far more noise than real data as will be later explained. In an attempt to overcome these problems, we considered two variations of the correlated divergence algorithm that employ: (i) protein subset selection and (ii) organism subset

selection. The first approach focuses on selecting a subset of proteins from each family, which are more likely to interact, and the second searches for those organisms in which the interaction is more likely to be preserved. Next we discuss both approaches and draw conclusions regarding the effectiveness of each one.

3.1. Protein Subset Selection

There are two evolutionary phenomena that can explain the presence of multiple “paralogous pairs”. In both cases the phenomena are driven by duplication events. The difference lies in the timing:

1. At the time of speciation of o , only one protein from each family existed. Duplication events then took place in o , possibly in coordination (e.g., when the interacting genes are located physically close to each other), forming multiple paralogous genes and possibly multiple interacting pairs.
2. These multiple paralogous genes were in existence before speciation occurs for o . When speciation occurred, these paralogs were passed on to o .

Given two sets of proteins, $F_{1,o}$ and $F_{2,o}$ from families F_1 and F_2 in organism o , the basic method of **Section 2.1.2** takes the average over the $|F_{1,o}| \times |F_{2,o}|$ protein pairs. However, even if one of these gene pairs is known to interact, it is unclear if all homologous gene pairs can form an interaction. Actually, it is unlikely that all paralogs from one family interact with all paralogs of the other family. Rather, it is more likely that each paralog is “tuned” to perform different functions (3, 52). Moreover, after speciation an interacting pair might mutate and become non-interacting. However, without experimentation it is hard to determine in advance which pairs interact and which are not.

We contend that this protein multiplicity will only weaken our co-evolutionary signal as many of the pairs considered are not interacting, and therefore are likely to evolve without explicit co-evolution constraints. It should be noted that there is an overwhelming evidence that biological systems employ fail-safe, redundancy-based mechanisms, thus suggesting that many of these pairs are actually interacting (53, 54). Nevertheless, the maximal number of expected interactions is of the order of $O(n)$ while the actual number of pairs considered in this analysis is of the order of $O(n^2)$. Thus the majority of pairs is only indirectly constrained. In this view it is clear that one should consider only the truly interacting pairs in the analysis, excluding all other homologs, even those that are significantly similar.

The problem has been addressed to some extent in previous studies. The algorithm presented in (30) works on families composed of exactly one protein from each organism in the common set (picking the closest protein to *Escherichia coli* proteins when paralogs were available). Two studies (49, 50) independently proposed algorithms in which more than one protein from a single organism may be considered. This problem is harder, since it is necessary to decide which pairs of proteins from the same organism are the most

likely to interact. The problem is tackled by looking for the pair of distance matrices that yields the highest correlation score. Assuming that the distance matrix of the first family is set, these algorithms explore the search space of all distance matrices which may be constructed for the second family. The number of different matrices is $m!$, where m is the number of proteins in each family and $m!$ is the number of permutations over these m proteins.

Since the search space becomes too big for large m , the algorithm employs sampling algorithms to find a locally maximal solution. The problem of a huge search space was partially resolved in (55), where only isomorphic permutations, namely permutations that keep the tree topology, are considered. The method can reduce the size of the search space significantly, but in the worst case the search space remains of the order $m!$.

In this section we test several methods that are primarily concerned with determining the true interacting subset. The first attempts to minimize the total distance between the selected proteins, the second more restrictive approach attempts to identify the set of orthologous proteins, and the third attempts to identify the subset that maximizes the correlated divergence score.

3.1.1. The Minimum Distance Method

The minimum distance method reduces the set of proteins from each organism in O_{common} to a single protein. This protein is chosen such that the overall distance (approximated using the sum of pairs (SOP) function (56)) between all protein pairs in the family is minimal. The assumption behind this method is that interacting proteins are likely to mutate less than their paralogs which are not involved in the interaction, in order to preserve the interaction. This method can be computationally intensive for large families, containing many proteins or organisms, because of the large number of possible combinations that have to be considered. To test this method we used smaller subsets of about 500 interacting and 500 non-interacting proteins that were constructed as described in **Section 2.1.3**, with the additional criterion that the number of combinations does not exceed 10^7 (we refer to these data sets as the **reduced data sets**). As before, we compute the distributions of correlated divergence scores over the interacting and the non-interacting sets, where the formula of **Eq. [1]** is revised to consider only a single protein from each organism selected using the minimum-distance criterion. The results are presented in **Fig. 4.6**.

As the graphs show, the average score with the minimum distance method increases (both for the interacting and the non-interacting set) compared to the basic model, which suggests that the method is effective in picking the relevant paralogs. The minimum distance method also improves the separation between the two sets. For example, 24.4% of the interacting set and 12.7% of the non-interacting set are assigned a score > 0.9 when using all

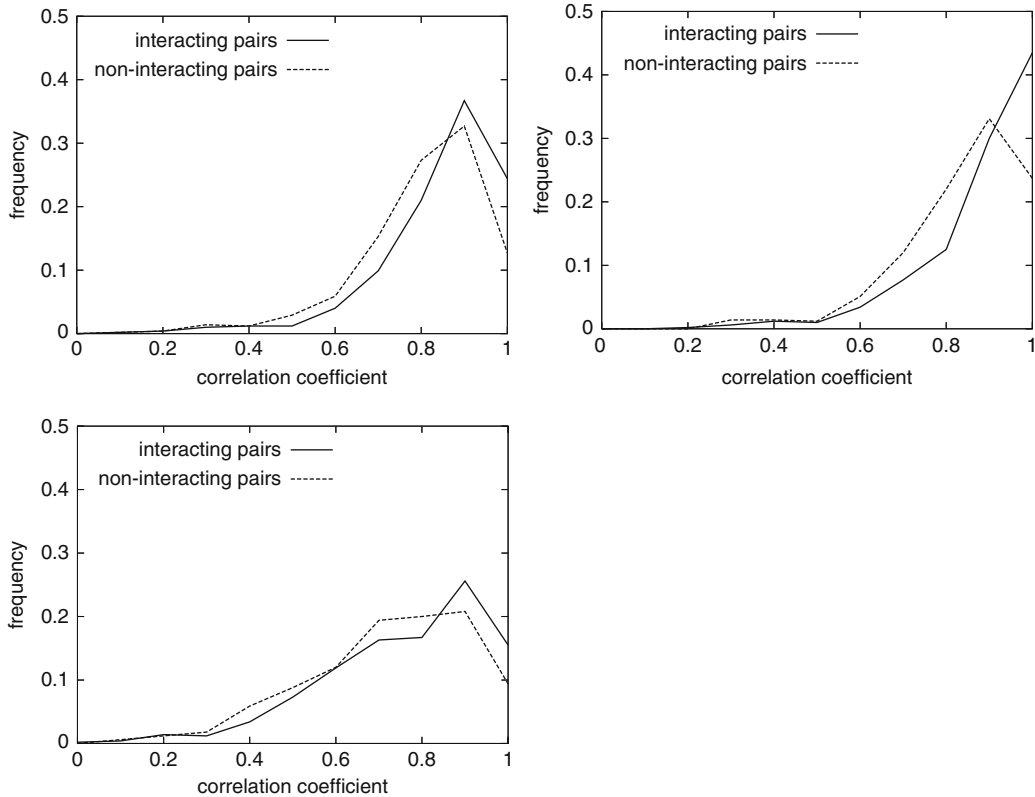


Fig. 4.6 **Distribution of correlated divergence scores for interacting vs. non-interacting pairs over the reduced data sets.** Density functions generated when all proteins in O_{common} are used (*left*), one protein from each organism in O_{common} is chosen by the minimum distance criterion (*middle*) and one protein from each organism is chosen at random (*right*).

paralogs. With the minimum-distance method, more than 43.5% of the interacting set are assigned a score >0.9 while only 23.7% of the non-interacting set exceed this threshold, thus decreasing the overlap between the two distributions at the high end of correlation scores. Indeed, the Jensen-Shannon (57) distance between the two distributions increases from $D_{\text{all}}^{\text{JS}} = 0.064$ to $D_{\text{min-dist}}^{\text{JS}} = 0.095$. Note that when one protein is selected at random from each organism (the *Random test*), the correlated divergence scores decrease and the distributions get closer, with a Jensen-Shannon distance of $D_{\text{random}}^{\text{JS}} = 0.035$ (see right panel of Fig. 4.6).

Interestingly, the basic correlated divergence model performs better over the reduced set and the scores are usually higher compared to the scores computed over the larger data sets (compare the left panel of Fig. 4.6 to left panel of Fig. 4.2). This might be attributed to more accurate assessment of the phylogenetic distances rather than co-evolution, since the reduced data sets contain fewer organisms and less homologs for each organism. However, it seems that the co-evolution signal also improves when the distant homologs are eliminated.

Optimizing the minimum-distance method for large families. In order to apply the minimum distance method to large families with large search spaces, one can apply the following optimization algorithm:

- **Input:** Protein family F over a set of organisms O . The maximal number of iterations $MaxIteration$.
The set F_o denotes the projection of F on o (the subset of proteins in F from organism $o \in O$).
- **Initialize:** The subset of minimum-distance proteins $S = \phi$ and $Iteration = 0$.
For each organism $o \in O$
 Pick one protein p at random from F_o
 $S = S \cup p$ (such that $S_o = p$)
- **Loop:** $Iteration = Iteration + 1$
For each organism o in O
 Set $p = S_o$ (the projection of the current set S on o)
 Set $S' = S \setminus p$
 Pick one protein p' at random from F_o s.t. $p' \neq p$
 If $SOP(p', S') > SOP(p, S)$ then $S = S \cup p'$
- **Until:** $Iteration = MaxIteration$ or if S converged.
- **Output:** The subset S

The algorithm attempts to improve the assignment of proteins by iterating over all organisms, trying each of their proteins in combination with the current best assignment of proteins from other organisms, and picking the one that minimizes the distance. The quality of the assignments is measured via the SOP score. While this algorithm is much more computationally efficient than exhaustive search over all possible assignments, it can still be computationally demanding for large families and it is suggested to stop it after a preset maximal number of iterations has reached. It should be noted that the application of this algorithm to larger data sets did not change the trends we observed with the smaller sets.

3.1.2. The Orthologs-Based Approach

Another variation we considered was to search for the group of orthologous proteins, using the Reciprocal Best BLAST Hit (RBH) algorithm (58, 59). This method was previously used to construct the InParanoid database of eukaryotic orthologs (60, 61). The RBH algorithm is stricter than the minimum distance algorithm; given two proteomes A and B , the two proteins $a \in A$ and $b \in B$ are considered orthologs if b is the first hit in a BLAST search in which a is the query and B is the database, and a is the first hit in the search of b against A . The method suffers from a high rate of false negatives since both proteins must be first in each other's list in order to be

considered as orthologs. While trying the method on our data we were left with too few proteins, which did not allow us to apply the correlated divergence model.

3.1.3. The Expectation Maximization approach

We also tested an EM-like algorithm for protein subset selection. In our case the hidden variables are the indicator variables that specify if a gene interacts with another gene or not. The maximization step selects from one organism at a time the pair of proteins that maximizes the co-evolution signal, given the existing set. The expectation step is essentially the averaging over all pairs in all other genomes (**Note 5**).

The algorithm consists of reducing the set of interacting proteins in each organism to a single interaction, i.e., $|F_{i,o}|=1$ for both families and all organisms o . As discussed above, it is very unlikely that all possible pairs interact; however, we anticipate that at least one pair interacts, and we target that gene pair. Our algorithm (shown below) first iterates through all protein organism sets and then continues these epochs until the subset remains unchanged and the correlation scores stop improving.

The EM Subset Selection Algorithm:

- **Input:** Two protein families F_1, F_2 over a set of common organisms O_{common} . The maximal number of iterations $MaxIteration$.
- **Initialize:** The subsets of proteins that maximize the correlated divergence score $S_1 = F_1, S_2 = F_2$. Set $Iteration = 0$
- **Loop:** $Iteration = Iteration + 1$

Foreach organism o in O_{common}

Set $S'_1 = S_1 \setminus S_{1,o}$

Set $S'_2 = S_2 \setminus S_{2,o}$

$MaxCorrelation = 0$

Foreach $p_1 \in F_{1,o}$

Foreach $p_2 \in F_{2,o}$

$Correlation = CorrelatedDivergence(S'_1 \cup p_1, S'_2 \cup p_2)$

If $Correlation > MaxCorrelation$ then

$S_1 = S'_1 \cup p_1$ and $S_2 = S'_2 \cup p_2$

$MaxCorrelation = Correlation$

Until: S_1 and S_2 converged or if $Iteration = MaxIteration$

- **Output:** The final subsets S_1 and S_2 .

The algorithm was implemented and tested over the interacting and non-interacting data sets. However, the resulting distributions were very similar (results not shown). One deficiency of the algorithm lies in its tendency to choose remote proteins that maximize the Pearson correlation score (see discussion of outliers

in **Section 2.1.5**). That is, the algorithm chooses members of the two families such that their distances from the other organisms are most alike, even if this means choosing distant members when closer alternatives exist. This is clearly undesirable as the selection bears no biological significance in the context of correlated divergence. This phenomenon emphasizes once again one of the major deficiencies with the Pearson correlation coefficient.

3.2. Organism Subset Selection

Our assumption that $O_{\text{common}} = O_{\text{interact}}$ is, in practice, not always true. After or during speciation, a pair of interacting proteins might lose its ability to interact. In **Section 2.1**, we conclude that if $O_{\text{interact}} \ll O_{\text{common}}$, then co-evolution signals from interacting proteins can quite easily be overwhelmed by data from the numerous non-interacting pairs; therefore, we would like to minimize this set to the smallest possible one, O_{interact} . Here we propose two simple methods for organism subset selection.

3.2.1. Closest Organism Criterion

Assuming that the pair of tested proteins indeed interacts, it is more likely that the interaction exists in organisms that are closer to the query proteins' organism rather than in distant organisms. Under this assumption we remove distant organisms in O_{common} . Given the pair of query proteins $p_{1,q}, p_{2,q} \in o_q$, the distance between o_q and any other organism $o_d \in O_{\text{common}}$ is estimated by

$$\text{dist}(o_q, o_d) = \sqrt{\text{evaluate}(p_{1,d})^* \text{evaluate}(p_{2,d})}, \quad [2]$$

where $\text{evaluate}(p_{i,d})$ is the minimal e-value assigned by BLAST to the most similar protein in o_d , with $p_{i,q}$ as the query.

Once all distances were estimated, we choose the closest N_r organisms and use these as our reduced set of common organisms, O_{common}^r , to compute the correlated divergence score as in **Eq. [1]**. We tested this method for $N_r = 8$. Our tests suggest that the exclusion of remote organisms does not affect the separation. To the contrary, the removal of the distant organisms resulted with almost identical graphs for both the interacting and the non-interacting data sets.

3.2.2. The Reduced Distance Matrices for Correlated Divergence

In the standard model of correlated mutations (as in **Section 2.1**), all protein pairs are considered in **Eq. [1]** when evaluating co-evolution. However, even with the minimal set of organisms and proteins (as in **Section 3.1**), this model is contradictory to the fundamental protein co-evolution assumptions. Given a phylogenetic tree topology representing O_{interact} , each organism o in this set is closely related only to a small subset of organisms. Although indirect relationships to all other organisms can be established by transitivity, the correlated mutations signal decreases quickly with the evolutionary distance between species, to practically undetectable levels for even relatively small distances. Therefore, when analyzing co-evolution within a protein family, one has to adjust

the model and compare only proteins whose organisms are phylogenetically closely related. Sato et al. (62) suggested a variant of the mirror-tree method, which reduces the high rate of false positives by subtracting information about the phylogenetic relations of the proteins represented in the distance matrix, so as to isolate and amplify the co-evolution signal. However, this approach presents another challenge. As discussed in the introduction, phylogenetic tree reconstruction is an extremely difficult problem, and complete phylogenetic species trees that span across higher-order domains (i.e., Eubacteria, Eukaryote, and Archea) are largely unavailable. To approximately define the set of relationships that are induced by the true phylogenetic tree, we use the given set of proteins from the two interacting families. However, we do not try to recover the complete topology of the tree, since we are only interested in organism pairs that are closely related. To assess co-evolution we consider only the k most similar relations. Specifically, for each organism we compile the list of neighboring organisms based on the average SOP score of the two homologous proteins and pick the k most similar ones. Denote by $kNN(i)$ as the set of k closest organisms to organism i , then the correlated mutation score under this reduced model is defined as

$$r = \frac{\sum_{i=1}^N \sum_{j \in kNN(i)} [D_1(o_i, o_j) - \mu_1][D_2(o_i, o_j) - \mu_2]}{\sqrt{\sum_{i=1}^N \sum_{j \in kNN(i)} [D_1(o_i, o_j) - \mu_1]^2} \sqrt{\sum_{i=1}^N \sum_{j \in kNN(i)} [D_2(o_i, o_j) - \mu_2]^2}} \quad [3]$$

We refer to this algorithm as the kNN algorithm. We ran the kNN algorithm on the data sets described in **Section 2.1.3**, with $k=8$. Our tests indicate that this method improves only slightly over the basic correlated divergence algorithm and the minimum-distance protein subset selection algorithm, suggesting that most interacting protein pairs do not co-evolve more strongly than what is expected in general for two sets of homologous proteins over the same set of organisms; and if certain positions exhibit correlated mutations, they are probably confined to the interaction site.

4. Conclusions

In this chapter we study methods to predict protein–protein interactions based on the co-evolution model. The premise of co-evolution methods was originally revealed in (29, 36, 40); however, without extensive assessment. Here we expand this model and test different variants.

The underlying co-evolution model is very appealing at first: if two proteins interact, one would expect to find some patterns of constrained, correlated mutations. Detecting such patterns in

protein pairs that have not been tested experimentally can suggest that the proteins interact. As opposed to docking and binding site identification methods, this method does not require knowledge of the structure. Similar to the Rosetta stone method, it attempts to predict interactions from sequence. Indeed, previous studies that explored the potential of this method suggested that the co-evolution model might be effective for the detection of novel interactions. However, these studies focused only on a few specific examples that appear to behave according to this model.

Motivated by this premise we tested this model and the variants that attempt to improve the signal-to-noise ratio. However, despite the extensions and enhancements we implemented, we were disappointed to find out that it is difficult to discern co-evolution signals due to interaction from the background evolutionary correlation which exists between any two sets of homologous proteins over the same set of organisms. A relatively small number of interactions seem to follow the co-evolution model (under the “mirror-tree” assumption), and only very few interactions can be predicted with the correlated divergence measure or its variants. There are several possible explanations. One possible reason could be that the data set is biased or skewed. Actually, the choice of the data set can greatly affect the results, and early experiments that we ran with other data sets looked promising at first. However, we soon realized that the too-good-to-be-true results were due to uneven sampling and statistical differences between the interacting and the non-interacting data sets. The final data sets we used in our tests are fairly large, were obtained from high-quality databases, and have similar statistical properties (*see* **Section 2.1.3**). Therefore, we think it is unlikely that the conclusions are the result of biased data sets. Another possible reason might be that the protein sets have too few homologs, or the chosen organisms are too close or too far. However, we believe that the main reason is that the correlated divergence measure is simply dominated by the background evolutionary correlation. Furthermore, the Pearson correlation measure is insensitive to weak correlation signals and has other drawbacks as discussed in **Section 2.1.5**. Other correlation measures (such as the Spearman rank correlation) might be more effective. However, we believe that if there was a signal of co-evolution, we would have detected it with one of the many variations we tried. Therefore, even if the signal exists, it ought to be very weak.

Clearly, if co-evolution occurs, it will be most pronounced in the interaction site and the correlation might be limited to the very few residues which are located in that interaction site. The correlated mutations between these sites, even if exist, might be overshadowed by other sites that are not constraint to the same extent. Moreover, the selective pressure might have eliminated any other possible variations that could have supported this hypothesis. The assumption of the mirror-tree

approach is that the whole protein evolves in a similar pace as its interacting partner. However, it seems that this hypothesis holds only in a very few cases, while in all other cases it is essentially impossible to discern interactions from non-interactions using the mirror-tree method and there is little evidence of co-evolution pressure. Refinements of the correlated divergence measure based on the minimum-distance method and the EM algorithms improved the signal slightly. However, even with these improvements the signal is too weak.

Our final conclusion is that the mirror-tree co-evolution model is not powerful enough to predict protein–protein interactions effectively in itself. Further enhancements, integration with other sources of information and with other techniques, and larger data sets with refined information about the binding site may improve the performance in the future.

5. Notes



1. For example, out of over 10,000 protein interactions that were available in the BIND database as of December 2003, only about 100 have detailed binding site information.
2. In terms of graphs, the proteins are viewed as nodes and each pair that is chosen determines an edge in the graph. The degree of each node is the number of pairs this node participates in. Our procedure results in two graphs (of interacting and of non-interacting proteins) over the same set of nodes, where the degree of each node is almost identical in both graphs.
3. It should be noted that even with a random choice of pairs, there is a chance that some of the pairs selected are actually interacting. If information on the subcellular locations of proteins is available, the pairs can be chosen from different locations to reduce this chance. However, subcellular location is available for a relatively small number of proteins and the probability to pick an interaction by chance is small to begin with; therefore, we do not apply additional filters to the negative set.
4. Early experiments were done with ClustalW (46) and iterative PSIBLAST (47). The correlated divergence measure can be sensitive to the choice of the MSA algorithm and therefore we opted for a more accurate MSA algorithm. MAFFT has been shown to outperform ClustalW and PSIBLAST, and yet it is relatively computationally efficient (48).
5. Formally, this procedure is not exactly an Expectation Maximization algorithm; however, it is inspired by the EM algorithm and is therefore referred to as an EM-like procedure.

Acknowledgments

This work is supported by the National Science Foundation under Grant No. 0133311 to Golan Yona, and by the National Science Foundation under Grant No. 0218521, as part of the NSF/NIH Collaborative Research in Computational Neuroscience Program.

References

- Schwikowski, B., Uetz, P. and Fields, S. A network of protein-protein interactions in yeast. *Nat Biotechnol.* 2000, 18:1257–61.
- Ho, Y., Gruhler, A., Heilbut, A., et al. Systematic identification of protein complexes in *Saccharomyces cerevisiae* by mass spectrometry. *Nature* 2003, 415:180–83.
- Ihmels, J., Levy, R. and Barkai, N. Principles of ranscriptional control in the metabolic network of *Saccharomyces cerevisiae*. *Nat Biotechnol.* 2004, 22:86–92.
- Milo, R., Shen-Orr, S., Itzkovitz, S., Kashtan, N., Chklovskii, D. and Alon, U. Network motifs: simple building blocks of complex networks. *Science* 2002, 298:824–27.
- Fields, S. and Song, O. A novel genetic system to detect protein-protein interactions. *Nature* 1989, 340:245–46.
- Uetz, P., Giot, L., Cagney, G., et al. A comprehensive analysis of protein-protein interactions in *Saccharomyces cerevisiae*. *Nature* 2000, 403:623–27.
- Sobolev, V., Sorokine, A., Prilusky, J., Abola, E. E. and Edelman, M. Automated analysis of interatomic contacts in proteins. *Bioinformatics* 1999, 4:327–32.
- Gallet, X., Charlotiaux, B., Thomas, A. and Brasseur, R. A fast method to predict protein interaction sites from sequences. *J. Mol. Biol.* 2000, 302:917–26.
- Espadaler, J., Romero-Isart, O., Jackson, R. M. and Oliva, B. Prediction of protein-protein interactions using distant conservation of sequence patterns and structure relationships. *Bioinformatics* 2005, 21(16): 3360–68.
- Teodoro, M., Phillips, G. and Kavradi, L. Molecular docking: A problem with thousands of degrees of freedom. *IEEE International Conference on Robotics and Automation (ICRA 2001)*, 2001 May, Seoul, Korea, pp. 960–966.
- Lu, L., Lu, H. and Skolnick, J. MULTI-PROSPECTOR: An Algorithm for the prediction of protein-protein interactions by multimeric threading. *Proteins* 2002, 49:350–64.
- Kini, R. M. and Evans, H. J. A hypothetical structural role for proline residues in the flanking segments of protein-protein interaction sites. *Biochem. Biophys. Res. Commun.* 1995, 212:1115–24.
- Aytuna, A. S., Gursoy, A. and Keskin, O. Prediction of protein-protein interactions by combining structure and sequence conservation in protein interfaces. *Bioinformatics* 2005, 21(12):2850–55.
- Clackson, T. and Wells, J. A. A hot spot of binding energy in a hormone receptor interface. *Science* 1995, 267:383–86.
- Thorn, K. S. and Bogan, A. A. ASEdb: A database of Alanine mutations and their effect on the free energy of binding in protein interactions. *Bioinformatics* 2001, 1:284–85.
- Sprinzak, E. and Margalit, H. Correlated sequence-signatures as markers of protein-protein interaction. *J. Mol. Biol.* 2001, 311:681–92.
- Aloy, P. and Russell, R. InterPreTS: Protein interaction prediction through tertiary structure. *Bioinformatics* 2003, 19:161–62.
- Deng, M., Mehta, S., Sun, F. and Chen, T. Inferring domain-domain interactions from protein-protein interactions. *Genome Res.* 2002, 12:1540–48.
- Liu, Y., Liu, N. and Zhao, H. Inferring protein-protein interactions through high-throughput interaction data from diverse organisms. *Bioinformatics* 2005, 21(15): 3279–85.
- Chen, X. W. and Liu, M. Prediction of protein-protein interactions using random decision forest framework. *Bioinformatics* 2005, 21(24):4394–400.
- Breiman, L. Random forests. *Mach. Learn.* 2001, 45:5–32.

22. Han, D., Kim, H., Jang, W., Lee, S. and Suh, J. PreSPI: A domain combination based prediction system for protein-protein interaction. *Nucl. Acids Res.* 2004, 32(21):6312-20.
23. Marcotte, E. M., Pellegrini, M., Ng, H. L., Rice, D. W., Yeates, T. O. and Eisenberg, D. Detecting protein function and protein-protein interactions from genome sequences. *Science* 1999, 285:751-53.
24. Enright, A. J., Iliopoulos, I., Kyripides, N. C. and Ouzounis, C. A. Protein interaction maps for complete genomes based on gene fusion events. *Nature* 1999, 402:86-90.
25. Park, D., Lee, S., Bolser, D., Schroeder, M., Lappe, M., Oh, D. and Bhak, J. Comparative interactomics analysis of protein family interaction networks using PSIMAP (protein structural interactome map). *Bioinformatics* 2005, 21(15):3234-40.
26. Huang, T., Tien, A., Huang, W., Lee, Y. G., Peng, C., Tseng, H., Kao, C. and Huang, C. F. POINT: A database for the prediction of protein-protein interactions based on the orthologous interactome. *Bioinformatics* 2004, 20(17):3273-76.
27. Sun, J., Xu, J., Liu, Z., Liu, Q., Zhao, A., Shi, T. and Li, Y. Refined phylogenetic profiles method for predicting protein-protein interactions. *Bioinformatics* 2005, 21(16):3409-15.
28. Dandekar, T., Snel, B., Huynen, M. and Bork, P. Conservation of gene order: A fingerprint of proteins that physically interact. *Trends Biochem. Sci.* 1998, 23:324-28.
29. Goh, C., Bogan, A., Joachimiak, M., Walther, D. and Cohen, F. Co-evolution of proteins with their interaction partners. *J. Mol. Biol.* 2000, 299:283-93.
30. Pazos, F. and Valencia, A. Similarity of phylogenetic trees as indicator of protein-protein interaction. *Protein Eng.* 2001, 14:609-14.
31. Tan, S., Zhang, Z. and Ng, S. ADVICE: Automated detection and validation of interaction by co-evolution. *Nucl. Acids Res.* 2004, 32:W69-W72.
32. Izarzugaza, J. M. G., Juan, D., Pons, C., Ranea, J. A. G., Valencia, A. and Pazos, F. TSEMA: Interactive prediction of protein pairings between interacting families. *Nucl. Acids Res.* 2006, 34:W315-W319.
33. Pazos, F., Helmer-Citterich, M., Ausiello, G. and Valencia, A. Correlated mutations contain information about protein-protein interaction. *J. Mol. Biol.* 1997, 271:511-23.
34. Valencia, A. and Pazos, F. Computational methods for the prediction of protein interactions. *Curr. Opin. Struct. Biol.* 2002, 12:368-73.
35. Pazos, F. and Valencia, A. In silico two-hybrid system for the selection of physically interacting protein pairs. *Proteins* 2002, 47:219-27.
36. Jansen, R., Yu, H., Greenbaum, D., Kluger, Y., Krogan, N. J., Chung, S., Emili, A., Snyder, M., Greenblatt, J. F. and Gerstein, M. A Bayesian networks approach for predicting protein-protein interactions from genomic data. *Science* 2003, 302(17):449-53.
37. Ashburner, M., Ball, C. A., Blake, J. A., Botstein, D., Butler, H., Cherry, J. M., Davis, A. P., Dolinski, K., Dwight, S. S., Eppig, J. T., Harris, M. A., Hill, D. P., Issel-Tarver, L., Kasarskis, A., Lewis, S., Matese, J. C., Richardson, J. E., Ringwald, M., Rubin, G. M. and Sherlock, G. Gene ontology: Tool for the unification of biology. The gene ontology consortium. *Nat. Genet.* 2000, 25(1):25-29.
38. Mewes, H. W., Frishman, D., Guldener, U., Mannhaupt, G., Mayer, K., Mokrejs, M., Morgenstern, B., Munsterkotter, M., Rudd, S. and Weil B. MIPS: A database for genomes and protein sequences. *Nucl. Acids Res.* 2002, 30(1):31-34.
39. Ben-Hur, A. and Noble, W. S. Kernel methods for predicting protein-protein interactions. *Bioinformatics* 2005, 21(Suppl. 1):i38-i46.
40. Gobel, U., Sander, C., Schneider, R. and Valencia, A. Correlated mutations and residue contacts in proteins. *Proteins* 1994, 18:309-17.
41. Birkland, A. and Yona, G. The BIOZON database: A hub of heterogeneous biological data. *Nucl. Acids Res.* 2006, 34:D235-D242.
42. Bader, G. D., Donaldson, I., Wolting, C., Ouellette, B. F., Pawson, T. and Hogue, C. W. BIND - The biomolecular interaction network database. *Nucl. Acids Res.* 2001, 29:242-45.
43. Xenarios, I., Fernandez, E., Salwinski, L., Duan, X. J., Thompson, M. J., Marcotte, E. M. and Eisenberg, D. DIP: The database of interacting proteins: 2001 update. *Nucl. Acids Res.* 2001, 29:239-241.
44. Katoh, K., Misawa, K., Kuma, K. and Miyata, T. MAFFT: A novel method for rapid multiple sequence alignment based on fast Fourier transform. *Nucl. Acids Res.* 2002, 30(14):3059-66.
45. Katoh, K., Kuma, K., Toh, H. and Miyata, T. MAFFT version 5: Improvement in accuracy of multiple sequence alignment. *Nucl. Acids Res.* 2005, 33(2):511-18.

46. Higgins, D. G., Thompson, J. D. and Gibson, T. J. Using CLUSTAL for multiple sequence alignments. *Methods Enzymol.* 1996, 266:383–402.
47. Altschul, S. F., Madden, T. L., Schaffer, A. A., Zhang, J., Zhang, Z., Miller, W. and Lipman, D. J. Gapped BLAST and PSI-BLAST: A new generation of protein database search programs. *Nucl. Acids Res.* 1997, 25:3389–402.
48. Do, C. B., Mahabhashyam, M. S. P., Brudno, M., and Batzoglou, S. PROBCONS: Probabilistic consistency-based multiple sequence alignment. *Genome Res.* 2005, 15:330–40.
49. Ramani, A. K. and Marcotte, E. M. Exploiting the co-evolution of interacting proteins to discover interaction specificity. *J. Mol. Biol.* 2003, 327:273–84.
50. Gertz, J., Elfond, G., Shustrova, A., Weisinger, M., Pellegrini, M., Cokus, S. and Rothschild, B. Inferring protein interactions from phylogenetic distance matrices. *Bioinformatics* 2003, 19(16):2039–45.
51. Henikoff, S. and Henikoff, J. G. Position-based sequence weights. *J. Mol. Biol.* 1994, 243:574–78.
52. Popescu, L. and Yona, G. Automation of gene assignments to metabolic pathways using high-throughput expression data. *BMC Bioinformatics* 2005, 6:217.
53. Miklos, G. and Rubin, G. The role of the genome project in determining gene function: Insights from model organisms. *Cell* 1996, 86:521–29.
54. Yona, G., Dirks, W., Rahman, R. and Lin, M. Effective similarity measures for expression profiles. *Bioinformatics* 2006, 22:1616–22.
55. Jothi, R., Kann, M. G. and Przytycka, T. M. Predicting protein–protein interaction by searching evolutionary tree automorphism space. *Bioinformatics* 2005, 21(Suppl. 1):i241–i250.
56. Carillo, H. and Lipman, D. The multiple sequence alignment problem in biology. *SIAM J. Appl. Math.* 1988, 48(5): 1073–82.
57. Lin, J. Divergence measures based on the Shannon entropy. *IEEE Trans. Info. Theory* 1991, 37(1):145–51.
58. Hirsh, A. E. and Fraser, H. B. Protein dispensability and rate of evolution. *Nature* 2001, 411(6841):1046–49.
59. Jordan, I. K., Rogozin, I. B., Wolf, Y. I. and Koonin, E. V. Essential genes are more evolutionarily conserved than are nonessential genes in bacteria. *Genome Res.* 2002, 12(6):962–68.
60. Remm, M., Storm, C. E. V. and Sonnhammer, E. L. L. Automatic clustering of orthologs and in-paralogs from pairwise species. *J. Mol. Biol.* 2001, 314:1041–52.
61. O’Brien, K. P., Remm, M. and Sonnhammer, E. L. L. Inparanoid: A comprehensive database of eukaryotic orthologs. *Nucl. Acids Res.* 2005, 33:D476–D480.
62. Sato, T., Yamanishi, Y., Kanehisa, M. and Toh, H. The inference of protein–protein interactions by co-evolutionary analysis is improved by excluding the information about the phylogenetic relationships. *Bioinformatics* 2005, 21(17):3482–89.

Chapter 5

Computational Reconstruction of Protein–Protein Interaction Networks: Algorithms and Issues

Eric Franzosa, Bolan Linghu, and Yu Xia

Abstract

Accurate mapping of protein–protein interaction networks in model organisms is a crucial first step toward subsequent quantitative study of the organization and evolution of biological systems. Data quality of experimental interactome maps can be assessed and improved by integrating multiple sources of evidence using machine learning methods. Here we describe the commonly used algorithms for predicting protein–protein interaction by genome data integration, and discuss several important yet often overlooked issues in computational reconstruction of protein–protein interaction networks.

Key words: Protein–protein interaction, machine learning, protein network, data integration, Naïve Bayes, logistic regression.

1. Introduction

In the past few years, significant progress has been made in genome-wide identification of protein–protein interactions, especially in model organisms such as *Saccharomyces cerevisiae* (1–6) and *Caenorhabditis elegans* (7), and also recently in human (8). With the availability of these experimental interactome maps, it is now possible for the first time to quantitatively study the organization and evolution of biological systems at the level of protein–protein interaction networks, and develop theoretical models that account for the observed statistical trends (9–11). This line of research depends crucially on the quality of the reconstructed protein–protein interaction networks, as measured by accuracy, completeness, and possible bias. The dependence of

derived organizational and evolutionary hypotheses on data quality is not always obvious; an excellent recent example is the observation that power-law topology of the interactome map depends on its completeness (12). Such studies underlie the importance of rigorous assessment and subsequent improvement of the quality of interactome maps by a combination of experimental and computational methods.

Here we focus on computational reconstruction of protein–protein interaction networks by integrating multiple sources of evidence (13–15). Such sources of evidence can be the interactome maps produced by different labs, other binary maps such as genetic interaction maps, or other genomic features suggestive of protein–protein interaction. The basic premise is simple: if multiple reliable sources of evidence all suggest that two proteins interact, then the probability that these two proteins interact is high. To make this intuition precise, we need to quantify the reliability of each source of evidence, taking into account data quality (as mentioned above), as well as redundancy and similarity among different sources of evidence. Machine learning methods provide a straightforward solution to this issue. In machine learning, we specify the simplest possible model that, we believe, captures the dominant structure in the data. In our case, the model relates multiple sources of evidence to whether or not two proteins interact. We then fit the model to a training set (selected from a small gold-standard data set), adjusting the model parameters so as to maximize the agreement between the model and the data. The performance of the learned model on unseen data can be evaluated using a separate testing set, again selected from the gold-standard data set. Finally, we apply the model genome-wide to generate predictions. Here, the complexity of the data is captured by the choice of the model. Linear models and their variants have been widely used, because: (1) these models often capture the dominant structure in the data: noise, incompleteness, redundancy, and correlation; (2) many nonlinear structures in the data can become linear after appropriate data transformation; (3) these models are simple: efficient optimization methods exist to fit such models to the data, and over-fitting problem is usually minimal.

In **Section 2**, we describe the choice of gold-standard positive and negative interaction data sets, genomic features for predicting protein–protein interaction, machine learning methods for predicting protein–protein interaction, and ways to transform nonlinear structure in continuous and graph-based data into linear structure. In **Section 3**, we describe additional important issues in reconstructing the interactome: the choice of the size of positive and negative examples, dealing with features whose predictive power is difficult to quantify, and the effect of size and bias in the experimental interactome maps.

2. Methods

2.1. Gold-Standard Positive and Negative Interaction Data Sets

There are two different ways of defining protein–protein interactions. The first definition is more specific: two proteins interact when they share a physical binding interface. This is also called binary interaction, and can be detected with yeast two-hybrid experiments. The second definition is broader: two proteins interact when they are subunits of the same complex. This is also called co-complex memberships, and can be detected with pull-down experiments. Here we focus on the prediction of co-complex memberships in yeast, but the same framework also applies to the prediction of binary interactions.

The gold-standard positive data set, a set of protein pairs that are known to interact, is usually constructed from known protein complexes annotated in MIPS (14, 16). Gold-standard positive data sets constructed in this way, although highly useful, are not perfect: they are biased toward important, well-behaved proteins and protein complexes associated with pronounced phenotypes or diseases. Unless explicitly modeled, standard machine learning methods are not able to correct such biases.

The gold-standard negative data set, a set of protein pairs that are known not to interact, is much harder to construct (17). This is because negative results are typically neither published nor stored in any database. One way to solve this problem is to assume that proteins that localize in different cellular compartments do not interact (14). An alternative approach is to construct an approximate gold-standard negative data set as all protein pairs that do not belong to the gold-standard positive data set and to use co-localization information as one of the many features (18, 19). There are several advantages of this approach. First, co-localization information is treated in the same way as all other features. Second, a protein is estimated to interact on average with at most 10–20 proteins out of $\approx 6,000$ proteins in yeast. As a result, the vast majority ($>99.5\%$) of the approximate gold-standard negative data sets are in fact true negatives. Third, gold-standard data sets do not need to be 100% accurate. A small amount of noise can be tolerated as long as the gold-standard data sets contain strong enough signals to guide the parameterization of the classifier.

2.2. Compiling a List of Genomic Features

Many protein pair features correlate with interaction. Such genomic features can be collected for each of the ≈ 18 million yeast protein pairs. Here we list a representative subset of these features: (1) experimental physical and genetic interaction maps from different labs; (2) the mapping of interologs (20), i.e., conserved interactions between two proteins or domains, from another organism to yeast; (3) features based on comparative genomic evidence, such as

similarity of phylogenetic profiles (21) and gene neighborhood (22), co-evolution (23), belonging to the same gene cluster (24), and the existence of domain fusion events in another organism (25, 26); (4) pair protein features that are derived from single protein features such as function (14), localization (14), mRNA expression (27, 28), abundance (15, 18), regulation (29), and phenotype (14, 15, 30); (5) features based on 3D structural analysis, such as multimeric threading (31).

Missing data is a serious problem and needs to be treated differently depending on the missing data mechanism. However, in many cases, if feature X contains missing data, simply creating a new binary variable “ X -is-missing” will work well in practice.

2.3. Naïve Bayes

Consider the following binary classification problem. Given a training set of independently and identically distributed samples $T = \{(x^{(i)}, y^{(i)}); i = 1, \dots, m\}$ of feature and binary class variables from an unknown distribution D , estimate a classifier $f(x)$ that predicts the binary class variable $y \in \{0, 1\}$ (whether or not the protein pair interacts) from the features x . Without loss of generality, suppose that we have two binary feature variables $x = (x_1, x_2)$, where $x_1, x_2 \in \{0, 1\}$. (We will discuss continuous feature variables later.) The goal here is to come up with a classifier $f(x)$ that minimizes the expected prediction error $\mathbf{E}_{(x,y) \in D} \mathbf{1}\{y \neq f(x)\}$, where $\mathbf{1}\{X\}$ is equal to 1 when statement X is true, and 0 otherwise.

According to statistical decision theory, the optimal classifier $f(x)$ as defined above can be written in the following way:

$$f(x) = \begin{cases} 1, & \text{when } \frac{p(y=1|x)}{p(y=0|x)} > 1 \\ 0, & \text{when } \frac{p(y=1|x)}{p(y=0|x)} \leq 1 \end{cases} \quad [1]$$

Now we make the Naïve Bayes assumption that features are conditionally independent: $p(x_1, x_2|y) = p(x_1|y)p(x_2|y)$. Under this assumption,

$$\frac{p(y=1|x)}{p(y=0|x)} = \frac{p(y=1)p(x_1|y=1)p(x_2|y=1)}{p(y=0)p(x_1|y=0)p(x_2|y=0)} \quad [2]$$

The five independent parameters in the above equation can be easily estimated from the training set.

2.4. Logistic Regression

Equation [2] is equivalent to the following equation:

$$\ln \frac{p(y=1|x)}{p(y=0|x)} = w_0 + w_1 x_1 + w_2 x_2 \quad [3]$$

This equation relates linearly the posterior log-odds of an interaction given the evidence with the presence or absence of each piece of evidence.

Naïve Bayes classifiers assume that features are conditionally independent. Such assumptions are often incorrect. In logistic regression, the linear model in **Eq. [3]** is fit to the data, without the extra assumption of conditional independence (32). The weights w_0 , w_1 , w_2 are obtained by maximizing the following likelihood function: $L_C(w_0, w_1, w_2) = \Gamma_{i=1}^m p(y^{(i)} | x^{(i)})$.

2.5. SVM and Boosting

The above maximum likelihood (ML) estimate of the weights w_0 , w_1 , w_2 is equivalent to minimizing the following function: $\sum_{i=1}^m \phi_{LR}(\alpha^{(i)})$, where $\alpha = (2y - 1)(w_0 + w_1x_1 + w_2x_2)$ is called the margin, and the loss function $\phi_{LR}(\alpha) = \ln(1 + e^{-\alpha})$ is a convex surrogate for the 0–1 loss function $\phi_{0-1}(\alpha) = I\{\alpha < 0\}$. Let us now relax the requirement for ML estimation and consider other ways to estimate the weights. The different estimation methods generally aim at minimizing the empirical classification error $\frac{1}{m} \sum_{i=1}^m \phi_{0-1}(\alpha^{(i)})$, with the 0–1 loss function surrogated by a convex loss function so as to make efficient global optimization possible. In the case of logistic regression, this convex surrogate loss function is $\phi_{LR}(\alpha) = \ln(1 + e^{-\alpha})$. But we are free to choose other appropriate convex surrogate loss functions; in particular, support vector machine (SVM) and AdaBoost use different loss functions (33): $\phi_{SVM}(\alpha) = \max(1 - \alpha, 0)$, and $\phi_{AdaBoost}(\alpha) = e^{-\alpha}$.

2.6. Regularization

In some cases even the linear model in **Eq. [3]** is too complex and causes over-fitting. For example, we usually have a small number of annotated protein–protein interactions, and a large number of genomic features most of which are irrelevant. In this case, we want to make the linear model even simpler by imposing the additional constraint that only a small subset of all features has non-zero weights. Such regularization can be done in several different ways. For example, a feature selection step can be performed prior to the model-fitting step. Alternatively, a regularization term can be added to the model-fitting step to penalize complex models, as done in SVM. Finally, AdaBoost uses greedy optimization coupled with early stopping to control the complexity of the model.

2.7. Nonlinear Continuous and Graph-Based Features

We previously focused on binary features. A categorical feature with n categories can be easily decomposed into n binary features. What about continuous features, such as expression correlation? In general, the posterior log-odds of interaction may depend on these continuous features in a nonlinear way. However, we can convert a nonlinear continuous feature into several linear binary features by binning the data. For example, we can bin the expression correlation data into three binary features: expression-correlation-high, expression-correlation-medium, and

expression-correlation-low. We can then fit a linear model to the transformed feature space, assigning three different weights to protein pairs with high, medium, and low expression correlation. Notice that even though the model is linear in the transformed categorical feature space, it is actually nonlinear in the original continuous feature space. This simple binning procedure allows us to extend the linear model to many nonlinear cases. There are also other more complex procedures, such as kernel-based methods (34).

Some genomic features are based on graphs such as interactome maps and genetic interaction maps. Several different metrics have been proposed to measure the distance between a pair of proteins in these graphs, such as diffusion distance (35), linear kernel (36, 37), and congruence score (38). These metrics can then be combined with the rest of the genomic features to predict protein–protein interaction.

2.8. Decision Tree and Random Forests

Sometimes the dependence of protein–protein interaction on genomic features is so complex that the linear relationship in Eq. [3] is no longer valid. The most common machine learning method that deals with such irreducible nonlinearity is decision tree and its variants, such as random forests. These methods have been successfully applied to the prediction of protein–protein interaction (39).

3. Notes



1. *How large should the gold-standard negative set be?* We usually fix the size of the gold-standard positive set to be a constant, as determined by the MIPS complex catalog, but we are free to vary the size of the gold-standard negative set. As the gold-standard negative set gets bigger, the classifier applies a stricter cut-off, and as a result predicts a smaller number of positive interactions. For all classifiers except Naïve Bayes, individual evidence weights will also change.

What, then, is the right choice for the negative example size? Shall we pick the same number of negative examples as positive examples? Or to the other extreme, shall we pick a lot more negative examples than positive examples to approximately preserve the ratio of positive to negative interactions in the entire proteome? The right choice depends on the prediction task at hand. If our task is not to correctly predict *all* interactions but rather to come up with a list of predicted interactions that are *accurate*, then none of the

above two methods are appropriate. Rather, we should choose the appropriate negative example size so that there are roughly equal numbers of true and false positives in the predicted interactions (17).

2. *Features whose predictive powers are difficult to quantify.* It is sometimes difficult to assess in a quantitative way the predictive powers of certain features, such as functional similarity based on Gene Ontology annotations. Because a subset of the Gene Ontology annotations are themselves derived from interaction information, part of the observed correlation between functional similarity and interaction is spurious. One way to solve this problem is to exclude the subset of the Gene Ontology annotations that are derived from interaction information (18).

Let us now consider a hypothetical situation where we do not know which subset of the Gene Ontology annotations are derived from interaction information. It then becomes impossible to quantify the predictive power of the functional similarity feature. However, this does not mean that this feature is not useful at all for making new predictions. Even without quantitative assessment, we can infer the usefulness of this feature based on biological common sense: interacting proteins should tend to share common biological functions. We argue that the best way to deal with this situation is to exclude the functional similarity feature from training-testing so as to obtain a conservative estimate of the prediction performance, but then to include the functional similarity feature in the integrated classifier for making final genome-wide predictions.

3. *Effect of size and bias in experimental interactome maps.* It is important to keep in mind that the statistical machine learning approach outlined here can only be applied straightforwardly to integrate large-scale, unbiased interactome mapping experiments, where the overlap with gold-standard data sets provides an accurate measurement of data quality. However, a significant fraction (34%) of the physical and genetic interactions contained in BioGRID (40) are from small-scale experiments, each mapping 100 or less interactions. It is difficult to assess individual small-scale data sets, but we can assess different methods by pooling together all data sets carried out using the same method. As shown in Table 5.1, the predictive power for co-complex memberships decreases from affinity capture to two-hybrid to genetic interaction, as expected. At the same time, co-complexed proteins are significantly enriched for almost all methods.

Table 5.1

The most popular methods for mapping physical and genetic interactions, compiled from BioGRID (40). Methods are sorted by decreasing number of interactions deposited in BioGRID, and only methods with more than 1,500 interactions are shown. For each method, we compute the fold enrichment, i.e., the fraction of co-complexed protein pairs that are detected using this method, divided by the fraction of all protein pairs that are detected using this method. A fold enrichment larger than 1 indicates that the method is predictive for co-complex memberships

Method	Number of interactions	Fold enrichment
Affinity capture – MS	18,747	166.7
Two-hybrid	9,642	75.4
Synthetic lethality	9,019	47.9
Synthetic growth defect	5,002	18.8
Affinity capture – western	3,523	354.8
Epistatic mini-array profile	3,416	7.5
Dosage rescue	2,442	138.4
Synthetic rescue	1,605	72.7
Phenotypic enhancement	1,425	80.8
Reconstituted complex	1,327	311.7

Many large-scale physical and genetic interaction mapping experiments are biased: these experiments are concerned with a specific subset of genes that share a common biological function or disease phenotype. Here, the use of a generic gold-standard positive data set is questionable, as it will tend to underestimate the data quality. For example, as shown in **Table 5.2**, there are three data sets with apparently unusually low prediction power for co-complex membership. However, close examination reveals that they are all biased maps that are concerned with a subset of the interactome. These sub-networks usually involve a specific function that is not previously well characterized and therefore underrepresented in the gold-standard positives, such as the proteins involved in DNA integrity and secretion, and membrane proteins. As a result of this bias, the quality of these data sets is significantly underestimated by standard machine learning methods. New methods are needed to accurately assess the quality of such biased interactome maps.

Table 5.2

Large-scale physical and genetic interaction data sets, compiled from BioGRID. Data sets are sorted by decreasing number of interactions deposited in BioGRID, and only data sets with more than 1,000 interactions are shown. For each data set, we again compute the fold enrichment. A fold enrichment larger than 1 indicates that the data set is predictive for co-complex memberships, if we assume no biases in these data sets

Data set	Method	Number of interactions	Fold enrichment
Krogan et al., 2006 (6)	Affinity capture – MS	7,076	275.8
Gavin et al., 2006 (5)	Affinity capture – MS	6,531	287.2
Pan et al., 2006 (41)	Synthetic growth defect	4,533	0.3
Ito et al., 2001 (2)	Two-hybrid	3,959	43.6
Ho et al., 2002 (3)	Affinity capture – MS	3,596	82.2
Schuldiner et al., 2005 (42)	Epistatic mini-array profile	3,416	7.5
Tong et al., 2004 (43)	Synthetic lethality	3,411	10.7
Gavin et al., 2002 (4)	Affinity capture – MS	3,210	324.8
Miller et al., 2005 (44)	Two-hybrid	1,941	9.4

Acknowledgments

Y.X. thanks Mark Gerstein for advice and support.

References

1. Uetz P, Giot L, Cagney G, Mansfield TA, Judson RS, Knight JR, Lockshon D, Narayan V, Srinivasan M, Pochart P, Qureshi-Emili A, Li Y, Godwin B, Conover D, Kalbfleisch T, Vijayadamodar G, Yang M, Johnston M, Fields S, Rothberg JM. A comprehensive analysis of protein-protein interactions in *Saccharomyces cerevisiae*. *Nature* 2000, 403(6770):623–7.
2. Ito T, Chiba T, Ozawa R, Yoshida M, Hattori M, Sakaki Y. A comprehensive two-hybrid analysis to explore the yeast protein interactome. *Proc Natl Acad Sci USA* 2001, 98(8):4569–74.
3. Ho Y, Gruhler A, Heilbut A, Bader GD, Moore L, Adams SL, Millar A, Taylor P, Bennett K, Boutilier K, Yang L, Wolting C, Donaldson I, Schandorff S, Shewnarane J, Vo M, Taggart J, Goudreault M, Muskat B, Alfarano C, Dewar D, Lin Z, Michalickova K, Willems AR, Sassi H, Nielsen PA, Rasmussen KJ, Andersen JR, Johansen LE, Hansen LH, Jespersen H, Podtelejnikov A, Nielsen E, Crawford J, Poulsen V, Sorensen BD, Matthiesen J, Hendrickson RC, Gleeson F, Pawson T, Moran MF, Durocher D, Mann M, Hogue CW, Figey D, Tyers M. Systematic identification

- of protein complexes in *Saccharomyces cerevisiae* by mass spectrometry. *Nature* 2002, 415(6868):180–3.
4. Gavin AC, Bosche M, Krause R, Grandi P, Marzioch M, Bauer A, Schultz J, Rick JM, Michon AM, Cruciat CM, Remor M, Hofert C, Schelder M, Brajenovic M, Ruffner H, Merino A, Klein K, Hudak M, Dickson D, Rudi T, Gnau V, Bauch A, Bastuck S, Huhse B, Leutwein C, Heurtier MA, Copley RR, Edelmann A, Querfurth E, Rybin V, Drewes G, Raida M, Bouwmeester T, Bork P, Seraphin B, Kuster B, Neubauer G, Superti-Furga G. Functional organization of the yeast proteome by systematic analysis of protein complexes. *Nature* 2002, 415(6868):141–7.
 5. Gavin AC, Aloy P, Grandi P, Krause R, Boesche M, Marzioch M, Rau C, Jensen LJ, Bastuck S, Dumpelfeld B, Edelmann A, Heurtier MA, Hoffman V, Hoefert C, Klein K, Hudak M, Michon AM, Schelder M, Schirle M, Remor M, Rudi T, Hooper S, Bauer A, Bouwmeester T, Casari G, Drewes G, Neubauer G, Rick JM, Kuster B, Bork P, Russell RB, Superti-Furga G. Proteome survey reveals modularity of the yeast cell machinery. *Nature* 2006, 440(7084):631–6.
 6. Krogan NJ, Cagney G, Yu H, Zhong G, Guo X, Ignatchenko A, Li J, Pu S, Datta N, Tikuisis AP, Punna T, Peregrin-Alvarez JM, Shales M, Zhang X, Davey M, Robinson MD, Paccanaro A, Bray JE, Sheung A, Beattie B, Richards DP, Canadien V, Lalev A, Mena F, Wong P, Starostine A, Canete MM, Vlasblom J, Wu S, Orsi C, Collins SR, Chandran S, Haw R, Rilstone JJ, Gandhi K, Thompson NJ, Musso G, St Onge P, Ghanny S, Lam MH, Butland G, Altaf-Ul AM, Kanaya S, Shilatifard A, O'Shea E, Weissman JS, Ingles CJ, Hughes TR, Parkinson J, Gerstein M, Wodak SJ, Emili A, Greenblatt JF. Global landscape of protein complexes in the yeast *Saccharomyces cerevisiae*. *Nature* 2006, 440(7084):637–43.
 7. Li S, Armstrong CM, Bertin N, Ge H, Milstein S, Boxem M, Vidalain PO, Han JD, Chesneau A, Hao T, Goldberg DS, Li N, Martinez M, Rual JF, Lamesch P, Xu L, Tewari M, Wong SL, Zhang LV, Berriz GF, Jacotot L, Vaglio P, Reboul J, Hirozane-Kishikawa T, Li Q, Gabel HW, Elewa A, Baumgartner B, Rose DJ, Yu H, Bosak S, Sequerra R, Fraser A, Mango SE, Saxton WM, Strome S, Van Den Heuvel S, Piano F, Vandenhoute J, Sardet C, Gerstein M, Doucette-Stamm L, Gunsalus KC, Harper JW, Cusick ME, Roth FP, Hill DE, Vidal M. A map of the interactome network of the metazoan *C. elegans*. *Science* 2004, 303(5657):540–3.
 8. Rual JF, Venkatesan K, Hao T, Hirozane-Kishikawa T, Dricot A, Li N, Berriz GF, Gibbons FD, Dreze M, Ayivi-Guedehoussou N, Klitgord N, Simon C, Boxem M, Milstein S, Rosenberg J, Goldberg DS, Zhang LV, Wong SL, Franklin G, Li S, Albala JS, Lim J, Fraughton C, Llamas E, Cevik S, Bex C, Lamesch P, Sikorski RS, Vandenhoute J, Zoghbi HY, Smolyar A, Bosak S, Sequerra R, Doucette-Stamm L, Cusick ME, Hill DE, Roth FP, Vidal M. Towards a proteome-scale map of the human protein-protein interaction network. *Nature* 2005, 437(7062):1173–8.
 9. Jeong H, Tombor B, Albert R, Oltvai ZN, Barabasi AL. The large-scale organization of metabolic networks. *Nature* 2000, 407(6804):651–4.
 10. Fraser HB, Hirsh AE, Steinmetz LM, Scharfe C, Feldman MW. Evolutionary rate in the protein interaction network. *Science* 2002, 296(5568):750–2.
 11. Milo R, Shen-Orr S, Itzkovitz S, Kashtan N, Chklovskii D, Alon U. Network motifs: Simple building blocks of complex networks. *Science* 2002, 298(5594):824–7.
 12. Han JD, Dupuy D, Bertin N, Cusick ME, Vidal M. Effect of sampling on topology predictions of protein-protein interaction networks. *Nat Biotechnol* 2005, 23(7):839–44.
 13. Marcotte EM, Pellegrini M, Thompson MJ, Yeates TO, Eisenberg D. A combined algorithm for genome-wide prediction of protein function. *Nature* 1999, 402(6757):83–6.
 14. Jansen R, Yu H, Greenbaum D, Kluger Y, Krogan NJ, Chung S, Emili A, Snyder M, Greenblatt JF, Gerstein M. A Bayesian networks approach for predicting protein-protein interactions from genomic data. *Science* 2003, 302(5644):449–53.
 15. Lu LJ, Xia Y, Paccanaro A, Yu H, Gerstein M. Assessing the limits of genomic data integration for predicting protein networks. *Genome Res* 2005, 15(7):945–53.
 16. Mewes HW, Heumann K, Kaps A, Mayer K, Pfeiffer F, Stocker S, Frishman D. MIPS: A database for genomes and protein sequences. *Nucleic Acids Res* 1999, 27(1):44–8.
 17. Jansen R, Gerstein M. Analyzing protein function on a genomic scale: The importance of gold-standard positives and negatives for network prediction. *Curr Opin Microbiol* 2004, 7(5):535–45.
 18. Xia Y, Lu LJ, Gerstein M. Integrated prediction of the helical membrane protein

- interactome in yeast. *J Mol Biol* 2006, 357(1):339–49.
19. Ben-Hur A, Noble WS. Choosing negative examples for the prediction of protein-protein interactions. *BMC Bioinformatics* 2006, 7(Suppl 1):S2.
 20. Yu H, Luscombe NM, Lu HX, Zhu X, Xia Y, Han JD, Bertin N, Chung S, Vidal M, Gerstein M. Annotation transfer between genomes: Protein-protein interologs and protein-DNA regulogs. *Genome Res* 2004, 14(6):1107–18.
 21. Pellegrini M, Marcotte EM, Thompson MJ, Eisenberg D, Yeates TO. Assigning protein functions by comparative genome analysis: protein phylogenetic profiles. *Proc Natl Acad Sci USA* 1999, 96(8):4285–8.
 22. Tamames J, Casari G, Ouzounis C, Valencia A. Conserved clusters of functionally related genes in two bacterial genomes. *J Mol Evol* 1997, 44(1):66–73.
 23. Goh CS, Cohen FE. Co-evolutionary analysis reveals insights into protein-protein interactions. *J Mol Biol* 2002, 324(1):177–92.
 24. Bowers PM, Pellegrini M, Thompson MJ, Fierro J, Yeates TO, Eisenberg D. Prolinks: A database of protein functional linkages derived from coevolution. *Genome Biol* 2004, 5(5):R35.
 25. Marcotte EM, Pellegrini M, Ng HL, Rice DW, Yeates TO, Eisenberg D. Detecting protein function and protein-protein interactions from genome sequences. *Science* 1999, 285(5428):751–3.
 26. Enright AJ, Iliopoulos I, Kyripides NC, Ouzounis CA. Protein interaction maps for complete genomes based on gene fusion events. *Nature* 1999, 402(6757):86–90.
 27. Ge H, Liu Z, Church GM, Vidal M. Correlation between transcriptome and interactome mapping data from *Saccharomyces cerevisiae*. *Nat Genet* 2001, 29(4):482–6.
 28. Jansen R, Greenbaum D, Gerstein M. Relating whole-genome expression data with protein-protein interactions. *Genome Res* 2002, 12(1):37–46.
 29. Yu H, Luscombe NM, Qian J, Gerstein M. Genomic analysis of gene expression relationships in transcriptional regulatory networks. *Trends Genet* 2003, 19(8):422–7.
 30. Yu H, Greenbaum D, Xin Lu H, Zhu X, Gerstein M. Genomic analysis of essentiality within protein networks. *Trends Genet* 2004, 20(6):227–31.
 31. Lu L, Arakaki AK, Lu H, Skolnick J. Multimeric threading-based prediction of protein-protein interactions on a genomic scale: Application to the *Saccharomyces cerevisiae* proteome. *Genome Res* 2003, 13(6A):1146–54.
 32. Ng AY, Jordan MI. On discriminative vs. generative classifiers: A comparison of logistic regression and Naive Bayes. *Adv Neural Inform Process Syst* 2002, 2(14):841–8.
 33. Zhang T. Statistical behavior and consistency of classification methods based on convex risk minimization. *Ann Statist* 2004, 32(1):56–85.
 34. Ben-Hur A, Noble WS. Kernel methods for predicting protein-protein interactions. *Bioinformatics* 2005, 21(Suppl 1):i38–46.
 35. Kondor RI, Lafferty JD. Diffusion kernels on graphs and other discrete input spaces. In: *Proc 19th International Conf on Machine Learning*. Morgan Kaufmann Publishers Inc., 2002, pp. 315–22.
 36. Rives AW, Galitski T. Modular organization of cellular networks. *Proc Natl Acad Sci USA* 2003, 100(3):1128–33.
 37. Lanckriet GR, De Bie T, Cristianini N, Jordan MI, Noble WS. A statistical framework for genomic data fusion. *Bioinformatics* 2004, 20(16):2626–35.
 38. Ye P, Peysers BD, Pan X, Boeke JD, Spencer FA, Bader JS. Gene function prediction from congruent synthetic lethal interactions in yeast. *Mol Syst Biol* 2005, 1:2005.0026.
 39. Lin N, Wu B, Jansen R, Gerstein M, Zhao H. Information assessment on predicting protein-protein interactions. *BMC Bioinformatics* 2004, 5:154.
 40. Stark C, Breitkreutz BJ, Reguly T, Boucher L, Breitkreutz A, Tyers M. BioGRID: A general repository for interaction datasets. *Nucleic Acids Res* 2006, 34(Database issue):D535–9.
 41. Pan X, Ye P, Yuan DS, Wang X, Bader JS, Boeke JD. A DNA integrity network in the yeast *Saccharomyces cerevisiae*. *Cell* 2006, 124(5):1069–81.
 42. Schuldiner M, Collins SR, Thompson NJ, Denic V, Bhamidipati A, Punna T, Ihmels J, Andrews B, Boone C, Greenblatt JF, Weissman JS, Krogan NJ. Exploration of the function and organization of the yeast early secretory pathway through an epistatic miniarray profile. *Cell* 2005, 123(3):507–19.
 43. Tong AH, Lesage G, Bader GD, Ding H, Xu H, Xin X, Young J, Berriz GF, Brost RL, Chang M, Chen Y, Cheng X, Chua G, Friesen H, Goldberg DS, Haynes J, Humphries C, He G, Hussein S, Ke L, Krogan N, Li Z, Levinson JN, Lu H, Menard P, Munyana C, Parsons AB,

- Ryan O, Tonikian R, Roberts T, Sdicu AM, Shapiro J, Sheikh B, Suter B, Wong SL, Zhang LV, Zhu H, Burd CG, Munro S, Sander C, Rine J, Greenblatt J, Peter M, Bretscher A, Bell G, Roth FP, Brown GW, Andrews B, Bussey H, Boone C. Global mapping of the yeast genetic interaction network. *Science* 2004, 303(5659):808–13.
44. Miller JP, Lo RS, Ben-Hur A, Desmarais C, Stagljar I, Noble WS, Fields S. Large-scale identification of yeast integral membrane protein interactions. *Proc Natl Acad Sci USA* 2005, 102(34):12123–8.

Chapter 6

Prediction and Integration of Regulatory and Protein–Protein Interactions

Duangdao Wichadakul, Jason McDermott, and Ram Samudrala

Abstract

Knowledge of transcriptional regulatory interactions (TRIs) is essential for exploring functional genomics and systems biology in any organism. While several results from genome-wide analysis of transcriptional regulatory networks are available, they are limited to model organisms such as yeast (1) and worm (2). Beyond these networks, experiments on TRIs study only individual genes and proteins of specific interest. In this chapter, we present a method for the integration of various data sets to predict TRIs for 54 organisms in the Bioverse (3). We describe how to compile and handle various formats and identifiers of data sets from different sources and how to predict TRIs using a homology-based approach, utilizing the compiled data sets. Integrated data sets include experimentally verified TRIs, binding sites of transcription factors, promoter sequences, protein subcellular localization, and protein families. Predicted TRIs expand the networks of gene regulation for a large number of organisms. The integration of experimentally verified and predicted TRIs with other known protein–protein interactions (PPIs) gives insight into specific pathways, network motifs, and the topological dynamics of an integrated network with gene expression under different conditions, essential for exploring functional genomics and systems biology.

Key words: Regulog, interolog, protein–DNA interaction prediction, transcriptional regulatory interaction (TRI) prediction, protein–protein interaction (PPI) prediction, homology-based approach, transferability of homologs.

1. Introduction

Transcriptional regulation controls the production of functional gene products essential for determining cell structure and function. It controls the amount of gene product and replenishment of degraded protein. This is fundamental for the differentiation, morphogenesis, versatility, and adaptability of the cell. Expanding the knowledge of gene regulation and understanding how it

relates to protein–protein interactions and gene expression provides insight into gene function and the mapping from genotypes to phenotypes. This knowledge is fundamental for advancing the design and development of biotechnology and medical treatments.

Though there have been several genome-wide studies of transcriptional regulatory networks, they have been focused on only a few model organisms (1, 4–7). Aside from those formed by genome-wide studies, only networks comprising small, specific, well-studied pathways are available (8–10). Based on publicly accessible databases of genome-wide transcriptional data and regulatory interactions with experimental verification (9, 11–13), several computational studies have reported the building of transcriptional regulatory networks (14–19), based upon the prediction of binding sequences of DNA and binding sites of transcription factors (20–26).

Among these approaches, the transferability of biological functions between homologous genes, originally proposed by Yu et al. (27), has been widely studied and deployed (3, 27–35). Thus, this chapter explores the transferability of protein–DNA interactions between organisms, or regulogs. This approach presumes that similarities in the sequence and structure of gene products suggest similar function.

We predict TRIs for an organism based on the transferability of similar interactions from other source organisms (*see Fig. 6.1*). In other words, we try to map the available TRIs in a source organism onto the target organism to find similar interactions.

The similarities of a predicted TRI ($I_{TFx' \rightarrow TFTy'}$) transferred from a source organism is defined as the geometric mean (the square root) of sequence similarities between (1) a transcription factor of an interaction in a source organism (TF) and its ortholog (**Note 1**) in a target organism (TF') defined as $I_{TF-TFx'}$, and (2) a sequence of transcription factor target in the source organism (TFT) and its ortholog in the target organism (TFT') defined as $I_{TFT-TFTy'}$. To map a TRI from a source organism onto a target organism, the interaction in the target organism needs to satisfy the following three conditions (27):

- i) TF and TF' are orthologs.
- ii) TFT and TFT' are orthologs.
- iii) The binding sites and binding sequences of TF appear in the upstream region of the TFT'.

Corresponding to the above conditions, if a TF regulating a TFT in a source organism has orthologs TF' and TFT' in a target organism, the pair of the interactions $TF \rightarrow TFT$ and $TF' \rightarrow TFT'$ are called regulogs (**Note 2**) (*see Fig. 6.1*). We can improve the accuracy and coverage of the resulting predictions

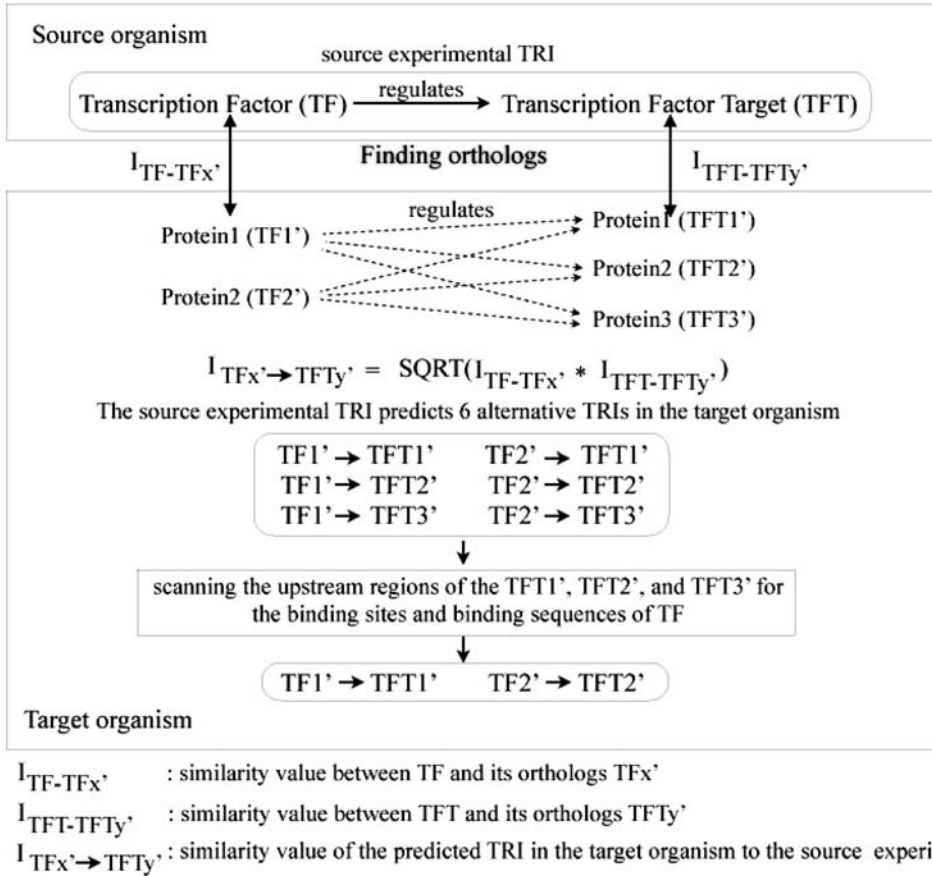


Fig. 6.1. Homology-based transcriptional regulatory interaction prediction.

by filtering out false-positive predictions using other data sources such as gene expression in differing cell cycle stages, protein localization, and membership in protein families. We have used these methods to predict regulogs for all 54 organisms in the Bioverse (3).

The major procedures involved in the prediction and integration of regulatory protein–DNA and protein–protein interactions include: (1) the preparation of essential data sets, including source experimental TRIs, binding sites and binding sequences of the experimentally verified transcription factors, the upstream regions of genes in target organisms, and name mapping between different identification systems; (2) the preparation of additional data sets, such as protein localization and assignment to protein families, for filtering and improving the accuracy of the predictions; (3) the determination of similarity between two protein sequences; (4) the prediction of the TRIs; and (5) the benchmarking of the prediction (*see Fig. 6.2*).

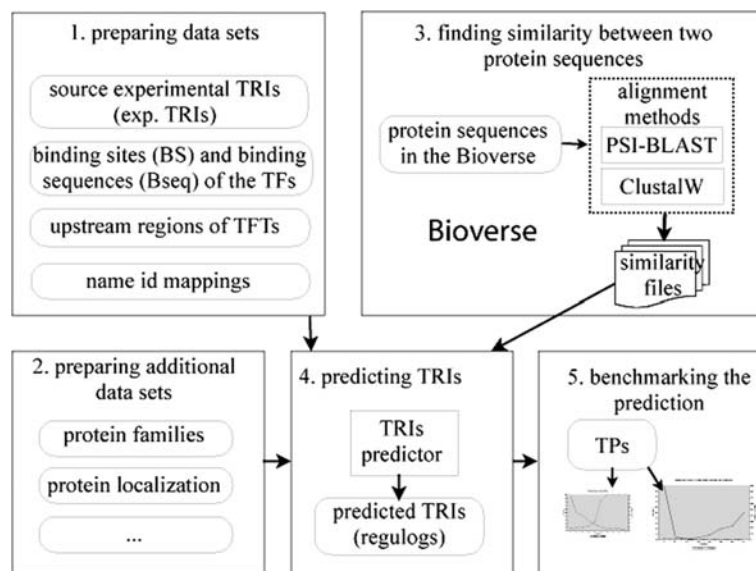


Fig. 6.2. Major steps in the prediction of transcriptional regulatory interactions.

While we present these steps in the context of TRI prediction, the methods and problems of data preparation are common steps in a large number of bioinformatics processes, especially in large-scale systems covering the genomes of multiple organisms. Specifically, the name mapping problem is ubiquitous; it makes all bioinformatics of this type difficult and decreases the coverage and certainty of predictions.

2. Methods

2.1. Preparing Data Sets

As homology-based approaches exploit the transferability of TRIs available in a source organism onto a target organism, the gathering of available interactions from different source organisms is the first essential step. This step is complicated as multiple sources provide different sets of non-comprehensive interactions for specific organisms, with varied data formats. The collection of data from several different sources, however, is essential for expanding the coverage of source TRIs for the prediction and construction of a gold-standard test set for the benchmarking. **Table 6.1** summarizes the sources of our experimental TRIs.

2.1.1. Source Experimental TRIs, Binding Sites, and Binding Sequences

Source experimental TRIs, binding sites, and binding sequences were compiled from two main sources: (1) public databases TRANSFAC[®] 7.0 (12), SCPD (36), BIND (37, 38), WormBase (39) via WormMart (40), RegulonDB (13), and DBTBS (11),

Table 6.1
Sources of experimental TRIs for source organisms

Source experimental TRIs	Source organisms	Description
TRANSFAC [®] (1) (+ binding sites and binding sequences)	<i>S. cerevisiae</i> , <i>H. sapiens</i> , <i>M. musculus</i> , <i>R. norvegicus</i> , <i>D. melanogaster</i> , <i>C. elegans</i> , <i>A. thaliana</i> , <i>O. sativa</i>	A database of eukaryotic transcription factors, genomic binding sites, and DNA-binding profiles
SCPD (36) (+binding sequences)	<i>S. cerevisiae</i>	The promoter database of <i>S. cerevisiae</i>
BIND (37, 38)	<i>H. sapiens</i>	The biomolecular interaction network database
WormBase (39, 40)	<i>C. elegans</i>	A database for genomics and biology of <i>C. elegans</i>
RegulonDB (13)	<i>E. coli</i>	A database of <i>Escherichia coli</i> K-12 transcriptional regulatory network, operon organization, and growth conditions
DBTBS (11) (+ binding sites and binding sequences)	<i>B. subtilis</i>	A database of transcriptional regulation in <i>B. subtilis</i>
Supplemental data from literature (1)	<i>S. cerevisiae</i>	Transcriptional regulatory networks in <i>S. cerevisiae</i>

and (2) supplemental data from experimentally determined TRIs described in the literature (1). As methods for gathering and transforming experimental TRIs, binding sites, and binding sequences into a unified format vary among different sources (Notes 3, 4, 5), we describe each of them in the following sections. To extend the compiled TRIs for the same organism from different sources, see Note 6.

2.1.1.1. TRANSFAC[®]

TRANSFAC[®] (12) is a database of transcription factors, their genomic binding sites, and DNA-binding profiles for eukaryotes. This database has two versions: (1) TRANSFAC[®] Professional, allowing bulk data downloads, and (2) TRANSFAC[®] Public Database, for online query only. We describe how to compile the experimental TRIs for each eukaryote in the Bioverse from the TRANSFAC[®] public database.

1. Go to the main searching page found at <http://www.gene-regulation.com/cgi-bin/pub/databases/transfac/search.cgi>, select “Factor” as the table to search.

2. On the page “searching in table *Factor*,” select “Organism Species (OS)” as the table field to search, and specify a specific organism name (e.g., *Saccharomyces cerevisiae*, *Homo sapiens*) as the search item. Set the limit of hits per page to the highest available (100), and then submit the search request. Save the results and edit them to contain only a list of accession numbers (one per line). These accession numbers correspond to transcription factors of the specified species.
3. Use our Python script:run_queryWeb to query detailed information about each transcription factor listed in the results. The Python script programmatically specifies a CGI search for a specific transcription factor, retrieves the search result, and writes it into a new file with a name matching the accession number. Following this, use the script: run_extractInfoFrom TRANSFACTFHtml-Files to extract the transcription factors and their (1) target genes, (2) binding sites, and (3) synonyms, into three separate files. To extract the binding sites to binding sequences, use script: run_extractBindingSeq.

2.1.1.2. SCPD

SCPD (36) is a database of promoters found within the *S. cerevisiae* genome. This database contains experimentally mapped transcription factor binding sites and transcription start sites as main entries. We manually compile the experimental TRIs from SCPD, using the following steps:

1. Go to <http://rulai.cshl.edu/cgi-bin/SCPD/getfactorlist>, click on the link for each transcription factor (e.g., ACE1, ADRI), and the corresponding page will appear.
2. Click on “Get regulated genes” button, and a list of genes regulated by the transcription factor will appear. Click on each of the regulated gene, a new window will appear. Save this window into a local file with the name of the regulated gene. Make this local file under a directory named by the transcription factor.
3. Make a name list of the transcription factors as an input for the script: run_parseSCPDToGetTRIs. Use this script to extract the source experimental TRIs for *S. cerevisiae* from SCPD into a file. Append this file to the source experimental TRIs of *S. cerevisiae* from other sources.

2.1.1.3. BIND

BIND (37, 38) is a database of biomolecular interactions, reactions, complexes, and pathway information. This database includes imported experimental TRIs from published research. BIND now becomes a component database of BOND (Biomolecular Object Network Databank). In addition to TRANSFAC[®], we compile the experimental TRIs of *H. sapiens* described in (41–43) from BIND, using the following steps:

1. Go to BOND (**Note 7**) <http://bond.unleashedinformatics.com/Action?> and register for a free account. Log in to BOND after getting the account. A BOND search page will appear.
2. Click on the “Identifier search”, a new window will appear. Select “PubMed Id” as the identifier from the list box on the left, and input the PubMed identifiers (PMIDs, **Note 8**) of the papers (41–43) one at a time into the text input on the right. Then, click on the “Search” button. A search result window will appear.
3. Click on the “Interactions” tab, a new window will appear. On the “Export Results:” list box, select “Cytoscape SIF”, a pop-up window for saving the exported result will appear. Save file into a local directory, edit it to have the format ready for use by the system. Append this file to the source experimental TRIs of human from other sources.

2.1.1.4. WormBase

WormBase (39, 40) is a database of genomics and biology of *Caenorhabditis elegans* and related nematodes. We compile the experimental TRIs of *C. elegans* from the database using the following steps:

1. Go to <http://www.wormbase.org/> and select a tab “Worm Mart” at the top of the page. A martview window will appear.
2. In this window, select the latest release of WormBase (i.e., “WormBase Release WS198”) for the “Version:” list box, select “Gene” for the “Dataset:” list box, and click on the “next” button. A window for filtering the queried data set will appear.
3. Under the “Identification” section on this window, check box “[Gene] Species” and select “Caenorhabditis elegans” in the list box, which corresponds to the check box. Also, check box “[Gene] Status,” and select “Live” in its corresponding list box.
4. Under the “Annotation” section, check box “Limit to Entries Annotated with:,” select “[Function] Trans. Regulator Gene” and “Only” in the corresponding list box, and radio box, respectively. Leave all other boxes as defaults. Click on the “next” button. A new page for formatting the output will appear.
5. Under the “IDs” section, uncheck boxes “Gene WB ID” and “Gene Public Name”. Under the “Gene Regulation” section, check boxes “Regulator Gene (Public Name)” and “Regulated Gene (Public Name).” Under the “Select the output format:” section, check radio box “Text, tab separated.” Under the “File compression:” section, check the radio box “gzip (.gz).” Under the “Enter a name for this result set:” section, enter a file name for the exported result. Leave all other boxes as defaults. Click on the “export” button. Save the exported file to a local directory.

6. Use the script: `run_parseWormBaseToGetTRIs` to extract the source experimental TRIs for *C. elegans* into a file. Append this file to the source experimental TRIs of *C. elegans* from other sources.

2.1.1.5. RegulonDB

RegulonDB (13) is database of the regulatory network and operon organization of *Escherichia coli* K-12. It is one of the two public databases of prokaryotes from which we compile source experimental TRIs. To compile experimental TRIs from RegulonDB, use the following steps:

1. Go to <http://regulondb.ccg.unam.mx/>, follow the tab “Downloads” and click on the “Data Sets” item.
2. On the page “Downloadable DataSets,” save “File 1. TF – gene interactions” (for experimental TRIs) and “TF binding sites” files (for binding sites and sequences) into a local directory. Edit these two files to have the same format as of the files generated for TRANSEAC.

2.1.1.6. DBTBS

DBTBS (11) is a database of transcriptional regulation of *Bacillus subtilis*. It is the other public database of prokaryotes used in this study. This database provides online access, but does not allow bulk download or programmatic search via CGI interface. To get the experimental TRIs of *B. subtilis*, we contacted the authors of (11) and asked for the experimental TRIs. The authors kindly gave us the requested data set in XML format. We wrote two scripts: `run_extractDBTBSForTFsAndBS` and `run_extractDBTBSForTRIs` that call our Python codes to parse and extract the (1) TRIs, (2) binding sites, and (3) binding sequences from this XML file, and write them to the experimental TRIs, binding sites, and binding sequence files, respectively.

2.1.2. Upstream Regions

Upstream regions of transcription factor target genes were compiled from (1) SGD (44) for *S. cerevisiae*, (2) UCSC (45) for *H. sapiens* (46), *Mus musculus* (47), *Rattus norvegicus* (48), and *Drosophila melanogaster* (49), (3) WormBase (39) for *C. elegans* (50), (4) TAIR (51) for *Arabidopsis thaliana* (52–55), (5) TIGR Rice Genome Annotation database (56) for *Oryza sativa* (57), and (6) NCBI for all prokaryotes, including *E. coli*, *B. subtilis*, etc. As methods for gathering and extracting the upstream regions and transforming them into a unified format vary among sources (Notes 9, 10), we describe these methods as follows.

2.1.2.1. SGD for *S. cerevisiae*

1. Go to <http://www.yeastgenome.org/> (44).
2. On the left side of this main page, in section “Download Data,” select “FTP.” A list of a directory in a new page will appear. From here, go into the “sequence” directory, then “genomic_sequence” directory, and then the “orf_dna”

directory. Copy the file `orf_genomic_1000_all.FASTA.gz` into a local directory. This file contains ORF sequences with introns, untranslated regions 1,000 bp upstream of the initial ATG and 1,000 bp downstream of the stop codon.

3. Use our code script: `run_extractUpstreamRegions_1000 bp_saccharomyces_cerevisiae` to parse and extract the saved file into a mapping file between the ORFs and their corresponding upstream regions.

See (**Note 11**) for an alternative way to get the upstream regions for *S. cerevisiae*.

2.1.2.2. UCSC Genome Browser for *H. sapiens*, *M. musculus*, *R. norvegicus*, and *D. melanogaster*

1. Go to <http://hgdownload.cse.ucsc.edu/downloads.html> (45).
2. In the box “Sequence and Annotation Downloads,” search for a specific organism. Select “Human,” for instance. This jumps to the “Human Genome” box. In the box “Human Genome,” select “Full data set,” which leads to a directory page containing a list of finished human genome assemblies with their descriptions.
3. Click on the `upstream<xxx>.zip` to download the files. These files are zipped and are in FASTA format, with each upstream sequence associated with an identifier system that is specific to an organism (i.e., NM_XXXX RefSeq in case of human, mouse, and rat, and FlyBase symbol in case of fly). The detailed descriptions of these upstream files are described on the same page. Basically, xxx in the name of an upstream region file stands for 1,000, 2,000, and 5,000 to represent the number of bases of each upstream region in each file (**Note 12**).
4. After downloading these files, use script: `run_extract_NP_NM_homo_sapiens` to parse and extract the mapping between NCBI GenBank identifiers (GIs) of human proteins to their corresponding RefSeq identifiers and use `run_extractUpstreamRegions_<xxx>bp_homo_sapiens` to generate a mapping file from GIs to upstream regions ready for use by the system.
5. Repeat Steps 1–4 for “Mouse” and “Rat.”
6. Use script: `run_extractUpstreamRegions_<xxx>bp_drosophila_melanogaster` to extract the upstream region file of *Drosophila* into a mapping file between FlyBase symbols and their corresponding upstream regions.

See **Note 13** for an alternative way to get the upstream regions for *H. sapiens*.

2.1.2.3. WormBase for *C. elegans*

1. Go to <http://www.wormbase.org/db/searches/advanced/dumper> (39).
2. In the “1. Input Options” box, type in “I II III IV V X XX XO.” These correspond to the chromosomes of *C. elegans*. In

the “2. Select one feature to retrieve,” click on “5 UTRs.” In the “3. Output options,” check box “flanking sequences only,” specify the flanking sequence lengths (i.e., 1,000 bp 5' flank, 0 bp 3' flank), leave the coordinates relative to “Chromosome,” select the sequence orientation as “Always on canonical strand,” select the output format “Save to disk (Plain TEXT),” then click “DUMP” button. The saved file is in FASTA format in which each upstream region is associated with a sequence name (gene model) and genetic nomenclature for *C. elegans*.

3. Use script: `run_extractUpstreamRegions_1000_bp_caenorhabditis_elegans` to parse, extract, and transform the saved file into a mapping file from GIs to upstream regions ready for use by the system.

2.1.2.4. TAIR for *A. thaliana*

1. Go to ftp://ftp.arabidopsis.org/home/tair/Sequences/blast_datasets/ (51).
2. Save files `TAIR_upstream_xxx_yyyymmdd`, where xxx represents the number of base pairs, and yyyymmdd represents the date the files are generated. These files are in FASTA format, with each upstream sequence associated with an *Arabidopsis* Genome Initiative locus identifier (AGI ID) (e.g., At1g01120).
3. Use script: `run_extractUpstreamRegions_1000_bp_arabidopsis_thaliana` to parse, extract, and transform the saved files into a mapping file from AGI IDs to their corresponding upstream regions.

2.1.2.5. TIGR for *O. sativa*

1. Go to ftp://ftp.tigr.org/pub/data/Eukaryotic_Projects/o_sativa/annotation_dbs/pseudomolecules/ (56), select the directory “version_x.x,” where x.x is the latest (i.e., version_5.0, for the current release), and then select the directory “all_chrs.” Under this directory, save file “all.1kUpstream” into a local directory (**Note 14**).
2. Use our script: `run_extractUpstreamRegions_1000_bp_TIGRRice` to parse, extract, and transform the saved file into a mapping file from TIGR_LOCUS IDs (e.g. LOC_Os01g01030.1) to their corresponding upstream regions.

2.1.2.6. NCBI for Prokaryotes

1. For *E. coli* K-12, go to ftp://ftp.ncbi.nlm.nih.gov/genomes/Bacteria, and enter the “Escherichia_coli_K12” directory.
2. Save files “xxx.fna” and “xxx.ptt” into `<organism>_Genome.fna` and `<organism>_ProteinMap.ptt`, respectively, where file with fna extension contains complete genome sequence, and file with ptt extension contains locations, the start and stop positions, for each gene on the genome sequence. For genomes that

have only sequences for chromosomes such as *Plasmodium falciparum*, save “xxx.fna” and “xxx.ptt” into <organism>_chr<n>.fna and <organism>_chr<n>_ProteinMap.ptt, respectively.

3. Repeat Steps 1 and 2 for all bacteria and other prokaryotes.
4. Use script: run_extractGIsToUpstreamRegionsFromGenomeSeqAndProteinMap to parse, extract, and transform the saved files into a mapping file from GIs to upstream regions for the organisms.

2.1.3. Name Mapping

Name mapping is another essential part of data preparation. Data sets such as transcription factors and transcription factor targets in the experimental TRIs and the upstream regions of gene sequences from several sources are associated with their own identifiers (e.g., common names of TFs and TFTs in TRANSFAC[®], ORFs from SGD, GenBank Identifiers at NCBI (GIs), gene IDs from Entrez Gene, WormBase IDs, FlyBase symbols, and AGI IDs for worm, fly, and *Arabidopsis*, and Refseq for upstream regions). Therefore, we map these identifiers to protein identifiers in the Bioverse (**Note 15**) for the prediction of protein–DNA interactions and the integration of protein–DNA and protein–protein interaction networks. In the following text we describe various name mappings required by the system.

2.1.3.1. Name Mapping from TFs, TFTs to Protein IDs in the Bioverse

To integrate the regulatory and protein–protein interactions, the TFs and TFTs from source experimental TRIs described in **Section 2.1.1** map to protein IDs in the Bioverse. While the Bioverse provides an ID-mapping file consisting of different ID systems (i.e., GIs from NCBI, ORFs from SGD, AGI IDs from TAIR) to protein IDs in the Bioverse, what we mainly have for the TFs and TFTs from source experimental TRIs are their common names. Hence, we establish an intermediate mapping that links these common names to protein IDs in the Bioverse (**Notes 16, 17**). The building process of an intermediate mapping file varies according to the ID system that will be used as the intermediate. We describe how to handle name mapping from the common names of TFs and TFTs in source experimental TRIs to protein IDs in the Bioverse, according to the formats for respective organisms.

Saccharomyces cerevisiae

1. Go to http://www.yeastgenome.org/gene_list.shtml (44).
2. Save file SGD_features.tab into a local directory.
3. Use script: run_extractNameMappingFromSGD to extract SGD_features.tab into a mapping file between the systematic ORF names and their common names.

Prokaryotes and Other Eukaryotes

1. Go to <http://www.ncbi.nlm.nih.gov/entrez/query.fcgi?db=gene>, on the left side, click on “Downloads (FTP).” A new page of directories will appear. Go into directory “DATA” and save the files `gene2refseq.gz` and `gene_info.gz` into a local directory.
2. Use scripts: `run_extract_gi_to_gene_id` and `run_extract_gene_id_to_names` to extract the `gene2refseq` and `gene_info`, respectively, and then use `run_buildGIToNames` to generate a mapping file from GIs to names for the organisms listed as an input of the script. For eukaryotes, we can improve the GI to name mapping with additional synonyms extracted from TRANSFAC[®].

Homo sapiens

Human genes do not have well-defined gene names as do genes in organisms such as yeast (**Note 18**). As the ID-mapping file for human in the Bioverse largely contains GI records, we decided to use GIs as intermediate ID mapping from a common name to a protein ID in the Bioverse. The original mapping file from GIs to common names is generated from `gene2refseq` and `gene_info` in Entrez Gene as described above. To refine the name mapping file, we combine synonyms (aliases) from additional sources such as TRANSFAC, HUGO (58), and OMIM (59), using methods listed as follows (**Note 19**).

- *Method to compile and extract synonyms from TRANSFAC[®]*

The synonyms of transcription factors are a part of the source TRIs compiled from TRANSFAC[®]. Hence, we do not need a separate compilation. As the script: `run_extractInfoFromTRANSFACTFHtmlFiles_homo_sapiens` also extracts the synonyms for each human transcription factor, we only need to combine the resulting file to the original GI-to-name mapping file from Entrez Gene using the script: `run_addSynonyms`.

- *Method to compile and extract synonyms from HUGO Gene Nomenclature Committee (HGNC)*

We compile and extract synonyms from HGNC using the following steps:

- Go to <http://www.genenames.org/> and click on the “Downloads” button at the top. The new page of database downloads will appear. Click on “Custom Download” listed in a box at the top of the page. A “Custom Downloads” page will appear.
- Check boxes: “Approved Symbol,” “Approved Name,” “Previous Symbols,” “Previous Names,” and “Aliases.” Check boxes “Approved” for the select status, and “Select all Chromosomes.” Scroll down and select the “ORDER BY” to change to the “Approved Symbol,” and the

“Output format” to be “Text.” Then, click the “submit” button. The result of the customized query will pop up in a new window. Save the result into a local file.

- Use script: `run_extracHumanGeneSynonymsFromHUGO`. This script extracts approved symbols, approved names, previous symbols, previous names, and aliases from all approved human genes into a file that will be used by script: `run_addSynonyms` to combine these names into the human GI-to-name mapping file.
- *Method to compile and extract synonyms from OMIM*
 - Go to <http://www.ncbi.nlm.nih.gov/entrez/query.fcgi?db=OMIM>. On the left side, under the FAQ section, click “Download.” A new OMIM FAQs page will appear. Under “Downloading OMIM” section in item 1, click on “omim.txt.Z” to download the complete text of OMIM and save it into a local directory.
 - Use script: `run_extracHumanGeneSynonymsFromOMIM` to extract the gene_ symbols and their synonyms from file `omim.txt` and write these extracted names into an output file. The script: `run_addSynonyms` combines these names into the human GI-to- name mapping file.

Caenorhabditis elegans

To extend the mapping from GIs to names for *C. elegans* generated by `run_buildGIToNames`, we compile gene aliases of *C. elegans* from WormBase via WormMart using the following steps:

1. Go to <http://www.wormbase.org/> and select a tab “WormMart” at the top of the page. A martview window will appear.
2. In this window, select the latest release of WormBase (i.e., “WormBase Release WS198”) for the “Version:” list box, select “Gene” for the “Dataset:” list box, and click on the “next” button. A window for filtering the queried data set will appear.
3. Under the “Identification” section on this window, check box “[Gene] Species” and select “Caenorhabditis elegans” in the list box, which corresponds to the check box. Also, check box “[Gene] Status,” and select “Live” in its corresponding list box. Leave all other boxes as defaults. Click on the “next” button. A new page for formatting the output will appear.
4. Under the “IDs” section, check box “Gene Names (merged).” Under the “Proteins” section, check box “NCBI Protein GI.” Under the “Select the output format:” section, check radio box “Text, tab separated.” Under the “File compression:” section, check the radio box “gzip (.gz).” Under the “Enter a name for this result set:” section, enter a file name for the exported result. Leave all other boxes as defaults. Click on the “export” button. Save the exported file to a local directory.

5. Use the script: `run_addSynonymsForC_elegans` to append these aliases into the available name mapping file for *C. elegans*.

Arabidopsis Thaliana

1. Go to <ftp://ftp.arabidopsis.org/home/tair/Genes/>.
2. Save file `gene_aliases.20080716` into a local directory. This file contains the mapping from AGI IDs to gene aliases in the format “AGI ID name1 name2” which is ready for use by the system.

Oryza Sativa

The ID name mapping file provided by the Bioverse for *O. sativa* does not include any intermediate ID that could be linked to the common names of transcription factors and their targets in the source experimental TRIs of rice, so we use the following steps to build a mapping file from the common names of rice proteins to the protein IDs in the Bioverse.

1. Go to ftp://ftp.tigr.org/pub/data/Eukaryotic_Projects/o_sativa/annotation_dbs/pseudomolecules/ (56), select the directory “version_x.x,” where x.x is the latest (i.e., version_5.0, for the current release), and then select the directory “all_chrs.” Under this directory, save file “all.pep” into a local directory.
2. Perform BLASTP from rice protein sequences retrieved from the Bioverse to protein sequences in all.pep using script: `run_blastp_bioverse_<rice species>_to_TIGR-rice`, where rice species could be “`oryza_sativa_japonica_fl`,” “`oryza_sativa_japonica_syngenta`,” and “`oryza_sativa_indica_9311`.”
3. Use scripts: `run_extractBLASTPSimilarity` and `run_extractTIGRLOCUSTo BioverseId` to extract the BLASTP result and then transform them into a mapping file from TIGR_LOCUS IDs to Bioverse IDs.
4. Use script: `run_buildTIGRRiceCommonNamesToBid` to extract all.pep file into a mapping file from TIGR_LOCUS to names.

2.1.3.2. Name Mapping of an ID System Associated with the Upstream Sequences from Different Sources to Protein IDs in the Bioverse

In this section, we describe how we build a mapping from a specific ID system associated with the upstream sequences of a specific organism to their corresponding protein IDs in the Bioverse.

Saccharomyces cerevisiae

The upstream regions of *S. cerevisiae* are annotated with the same sets of ORFs from SGD. Hence, we do not have the problem of mapping from these ORFs to protein IDs in the Bioverse.

Homo sapiens, Mus musculus, and Rattus norvegicus

The upstream regions of human, mouse, and rat compiled from UCSC Genome Browser (45) are annotated with NM_XXXX, which are the RefSeq accession numbers for nucleotide sequences. However, the name mapping from common names to protein IDs in the Bioverse is via NCBI GenBank Identifiers (GIs). Hence, these RefSeq numbers are not directly usable by the system. To handle this mapping issue, we use the following steps to transform the RefSeq accession numbers to protein GIs.

1. Go to ftp://ftp.ncbi.nih.gov/refseq/H_sapiens/mRNA_Prot/ and save file `human.protein.gpff.gz` into a local directory. This file contains GenBank records of NP_XXXX, which are the RefSeq accession numbers for protein sequences in human that are associated with GIs and NM_XXXX.
2. Extract the mapping between NP_XXXX, NM_XXXX RefSeq numbers and protein GIs using our script: `run_extract_NP_NM_homo_sapiens`. This code will result in a mapping file of GIs, NP_XXXX and NM_XXXX. The extracted mapping will be used as an input of script: `run_extractUpstreamRegions_1000bp_homo_sapiens` for extracting the upstream regions mentioned in **Section 2.1.2** for human.
3. Repeat Steps 1 and 2 for mouse and rat by accessing: ftp://ftp.ncbi.nih.gov/refseq/M_musculus/mRNA_Prot, and ftp://ftp.ncbi.nih.gov/refseq/R_norvegicus/mRNA_Prot, respectively.

Oryza sativa

The compiled upstream sequences of *O. sativa* (cultivar Nipponbare of *Oryza sativa* L. ssp. japonica) are associated with TIGR_LOCUS IDs. As we have built a mapping file from TIGR_LOCUS IDs to protein IDs in the Bioverse, we do not encounter the mapping problem for this genome.

Arabidopsis thaliana

The compiled upstream sequences of *A. thaliana* are associated with AGI IDs that are also available in the ID-mapping file provided by the Bioverse, so we do not encounter the mapping problem for this genome.

Drosophila melanogaster

While we compiled the upstream region files of *D. melanogaster* from UCSC as of human, mouse, and rat, the upstream sequences of the fly are not associated with RefSeq numbers. Instead, they are associated with FlyBase symbols that are also available in the ID-mapping file provided by the Bioverse. So, in case of fly, to extract the upstream regions, we use a script similar to the script of extracting the upstream region from FASTA format for yeast and *Arabidopsis*.

Prokaryotes

As we compile and extract the upstream regions of prokaryotes from NCBI, where genes and proteins are already associated with nucleotide and protein GIs, we do not have a mapping problem from an ID system associated with the upstream sequences to protein IDs in the Bioverse.

2.2. Preparing Additional Data Sets

This section describes the preparation of additional data sets utilized for improving the accuracy of TRI predictions.

2.2.1. Preparing Protein Localization

Protein localization is employed as a filter for improving the accuracy of the predicted TRIs. It strongly correlates with mRNA co-expression, as well as physical and functional interactions (60, 61). We compile protein localization data for *S. cerevisiae* from the TRIPLES database (62, 63) and Yeast GFP Fusion Localization database (60) (Notes 20, 21). To retrieve the protein localization data from these databases, use the following steps.

2.2.1.1. TRIPLES

1. Go to ftp://ygac.med.yale.edu/ygac_pub_ftp/.
2. Save the file `localization_pub_data_9_4_01.tab` into a local directory and use script: `run_extractPLOCFromTRIPLES` to extract the ORFs and their localizations into a file ready for use by the system.

2.2.1.2. Yeast GFP Function Localization Database

1. Go to <http://yeastgfp.ucsf.edu/>.
2. Within the banner at the top of the page, click on “Go” for the advanced query. A new page will appear. In this page, leave “Search Criteria” as default, where the inputs of all search criteria including of the “Subcellular Localization” will be wildcards (*). For the “Display Options,” check the box “Download the selected dataset as a tab-delimited file” and box “include localization table.” Press the “submit” button. The system will write the query result into a file “downloadxxxxxxx.txt” and put it in the “Search Results” section.
3. Save the result file into a local directory and use script: `run_extract PLOCFromYeastGFP` to extract the ORFs and their localizations into a file and use script: `run_combinePLOCs` to combine the results from both TRIPLES and Yeast GFP into a single file ready for use by the system.

2.2.2. Preparing Protein Families

The protein family is considered as another filter for improving the accuracy of the predicted TRIs. We hypothesize that a predicted transcription factor should share protein domains with its source transcription factor. At present, all protein families are compiled from TRANFAC (12) (Note 22), using the following steps:

1. Go to <http://www.gene-regulation.com/cgi-bin/pub/databases/transfac/search.cgi?>.

2. On this page, click on the “Class” button. The new page for searching the Class table will appear. Input a wildcard (*) in the “Search term” text field. Select “Class (CL)” as the field to search in the table, and “100” as number of hits per page. Then, click the “Submit” button. The new page of protein classes will appear.
3. Save this page into the local directory as a mapping file between class accession numbers and their descriptions. Then, click on each accession number in this page to save as a file on a local directory. These saved files will be used by the prediction method for filtering the predicted TRIs. A predicted TRI will be filtered out if its TF has no sharing of any protein families with the source TF (**Notes 23, 24**).

2.3. Finding Similarity Among Protein Sequences

As we use homology-based approaches for TRI prediction, the determination of similarity among protein sequences is an essential step. In the following section, we describe the preparation of protein sequences and the use of alignment methods for finding sequence similarity.

2.3.1. Preparing Protein Sequences

We compile protein sequences for a specific organism from the Bioverse (3), using the following steps:

1. Prepare a text file that lists the names of the organisms (one per line), which will be queried for all protein sequences.
2. Use script: `run_getMoleculeSeqsViaRPC` to retrieve the protein sequences for the organisms listed in the prepared text file in Step 1 from the Bioverse via XML RPC server (see Chapter 22) (**Note 25**).

2.3.2. Finding Similarity Among Protein Sequences

Similarities between protein sequences can be determined using several alternative alignment methods. We summarize each method and discuss their effect on the results.

2.3.2.1. Alignment Methods

1. **BL2SEQ** (64) is a BLAST-based tool for aligning two protein or nucleotide sequences that are presumably known to be homologous. It utilizes the BLAST (**Note 26**) engine (65) for local alignment. The main purpose of BL2SEQ is to compare the similarity between two sequences and reduce the processing time of using the standard BLAST program; BL2SEQ is the fastest, but least sensitive, method compared with the other alignment methods described here.
2. **PSI-BLAST** (**Note 27**) (Position-specific iterative BLAST) (66) is a feature of BLAST 2.0. It improves the sensitivity of protein–protein BLAST (BLASTP) using a position-specific scoring matrix (PSSM) constructed from a multiple sequence alignment of the best hits in each of the most recent iteration, such that it refines the PSSM over sequential iterations. Each

position in the PSSM will contain varied scores according to the conservation of the position. A position that is highly conserved will get a higher score. PSI-BLAST is much more sensitive than the standard BLAST program in capturing the distant evolutionary relationships or weak relationships between protein sequences.

3. **SSearch** implements the Smith-Waterman algorithm (67, 68) for local sequence alignment. Its main purpose is to avoid misalignment due to high noise levels in the low similarity regions of the distantly related sequences. Hence, it ignores all these regions and focuses only on regions that have highly conserved signals with positive scores. The Smith-Waterman algorithm guarantees optimal local alignment with the trade-off of moderately demanding computing resources. Hence, it is too slow for searching a large genomic database such as GenBank.
4. **ClustalW** (69) is a progressive *global multiple alignment* method that improves the sensitivity of highly divergent sequence alignment. It incorporates (1) an individual weight for each sequence, (2) varied amino acid substitution matrices at different stages of the alignment, (3) residue-specific gap penalties, and (4) position-specific gap penalties. ClustalW consists of three main steps: (1) performing pairwise alignments for all pairs of sequences in order to generate the distance matrix, (2) building a guide tree from the calculated distance matrix, and (3) carrying out a multiple alignment guided by the tree.

In our benchmarking process, we search for similar protein sequences of a source transcription factor (TF) or a source transcription factor target (TFT) in a target organism using PSI-BLAST. Then, we use ClustalW to create multiple alignments of the source protein sequence and the similar protein sequences found with PSI-BLAST.

2.3.2.2. Similarity Assessment

BLASTP and PSI-BLAST assess the similarity between query and protein sequences in a database by creating a bit score, an E value (expectation value), and match types with identities, positives, and gaps. The “bit score” is the normalized raw score (**Note 28**) according to the statistical variables defined in the scoring system. This score allows the comparison between different alignments with different scoring matrices. The “E value” is the probability that the similarity found in this alignment might happen by chance, with a lower E value corresponding to a more significant score. The “identity” is the ratio of the number of identical residues over the total number of aligned residues and gaps between a query and a target protein sequence. The “positive” is the ratio of the number of identical plus the non-identical but

conserved residues (represented by a minus sign in the alignment section of the blast result) over the total number of aligned residues and gaps between a query and a target protein sequence. The “gap” is the number of gaps (represented by a dash symbol), either in the query or in the target protein sequence, over the total number of aligned residues and gaps between a query and a target protein sequence.

SSearch assesses the similarity between two sequences via the Z-score, Smith-Waterman score, E() value, percentage identity, and percentage similarity. The Smith-Waterman score is calculated from a scoring matrix that includes the match and mismatch scores, a gap creation penalty, and a gap extension penalty. The Z-score is a normalized score calculated from a linear regression performed on the natural log of the sequence length of the search set. SSearch uses the distribution of Z-scores to estimate the expected number of sequences (represented by E() value) produced by chance with equal or greater Z-score than that attained from the search. The greater the Z-score, the lower the E() value. The percentage identity and percentage similarity represent the number of identical (represented by two vertical dots in the alignment section of the SSearch result) and the number of conserved but not identical (represented by a single dot) residues over the number of overlapping amino acids, respectively.

While higher scores and lower E values imply a better hit (such that two sequences are significantly similar), these values are calculated based on local alignment. Likewise, the percentage identity and percentage similarity are calculated only from aligned segments. Hence, in the case of two sequences with distant evolutionary relationships or weak relationships, these values are not directly representative of the similarity between them. Therefore, in our benchmarking, we assess the similarity between two sequences using the following steps:

1. For each protein (either TF or TFT) in the test set, we use PSI-BLAST to (i) find its top hits from all other proteins in the test set, and (ii) find its top hits from proteins in the Uniprot (again, we need to handle name mapping from the Uniprot protein identifier to the protein ID in the Bioverse), and then (iii) put these two sets of protein sequences into a file (including the query protein sequence). The number of files will be equal to the number of proteins in the test set (all distinct TFs and TFTs).
2. We use ClustalW to create multiple alignments for the set of sequences in each file. Based on the global multiple alignment results, we assess the similarity between the query protein sequences to every other hit sequence in the resulting file, as the number of identical residues over the total number of residues of the hit sequence. We call this ratio the *fraction*

identity (FI). Identical protein sequences will have $FI = 1.0$, while the FI between two proteins with no similar sequences is equal to 0.0.

2.4. Predicting TRIs

To predict TRIs, we developed a Python script following the homology-based approach described in the introduction. This code implements the following steps (as shown in **Fig. 6.3**).

1. For each source experimental TRI compiled in **Section 2.1**, find the orthologous proteins TFx' and $TFTy'$ of TF and TFT in a target organism, from the similarity values (i.e., $I_{TF-TFx'}$ and $I_{TFT-TFTy'}$ in **Fig. 6.3**) generated in **Section 2.3**. The numbers of orthologous proteins are limited by the cutoffs of similarity values (i.e., E-values, bit score, Z-scores, percentage identity, percentage similarity; with each varied according to the alignment methods). The predicted TRIs stem from all combinations of homologous TFx' and $TFTy'$ in the target organism. Each is assigned a similarity of interaction $I_{TFx' \rightarrow TFTy'}$, where

$$I_{TFx' \rightarrow TFTy'} = \text{sqrt}(I_{TF-TFx'} * I_{TFT-TFTy'})$$

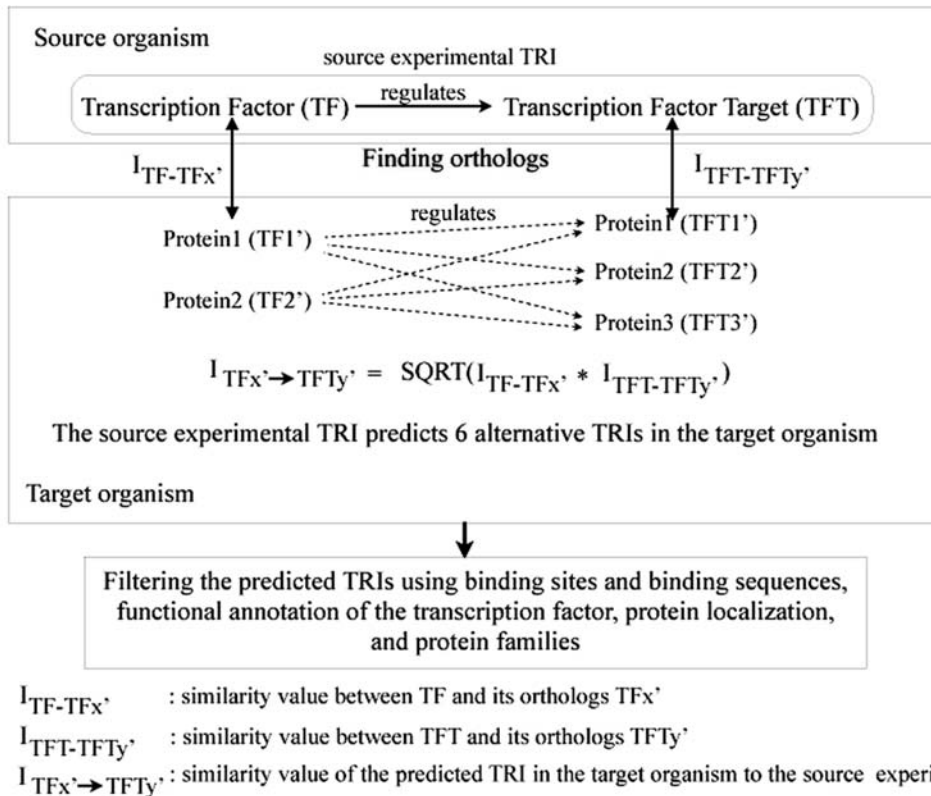


Fig. 6.3. Predicting TRIs.

The following steps are optionally used for improving the accuracy of TRI prediction.

1. Filter out the predicted TRIs for which functional annotation does not include “transcription factor.” Note that the annotation of a predicted TF (TF_x) is queried from XML RPC server in the Bioverse.
2. Filter out the predicted TRIs for which an upstream region (prepared in **Section 2.1.2**) of their TFTs is not found with any binding sites and binding sequences (prepared in **Section 2.1.1**) of the TF of source experimental TRI (prepared in **Section 2.1.1**).
3. Filter out the predicted TRIs for which the TF and TFT do not share any protein localization (prepare in **Section 2.2.1**).
4. Filter out the predicted TRIs for which the protein families (prepared in **Section 2.2.2**) of its TF and of the TF of the source experimental TRI do not overlap.

2.5. Benchmarking

One of the important issues for a prediction method is its accuracy and coverage. In this section, we describe a design experiment for measuring accuracy and coverage for this prediction method.

2.5.1. TP and Test Set

Accuracy and coverage are defined as follows.

$$\text{Accuracy}_x = A * 100 / (A + B)$$

$$\text{Coverage}_x = A * 100 / |TP|$$

for which:

x = a similarity value cutoff of E-values, Z-scores, percentage identity, or fraction identity used for discarding the predicted TRIs.

A = the number of predicted TRIs in the true positive set (TP) at cutoff x .

B = the number of predicted TRIs not in TP at cutoff x .

$|TP|$ = the number of source experimental TRIs in TP.

The true positive set (TP) of a target organism contains all experimental TRIs of the organism. If TRIs in the true positive set contain N distinct transcription factors (TFs) and M distinct transcription factor targets (TFTs), the test set will contain $N \times M$ TRIs, which result from all-against-all combinations between TFs and TFTs in the TP (**Note 29**). We define accuracy as the fraction of TRIs predicted by the system that are in TP out of all the predicted TRIs (either in TP or not in TP) at a specific cutoff. We define the coverage as the fraction of TRIs predicted by the system that are in TP at a specific cutoff over all TRIs in TP. Higher FI threshold cutoffs correspond to higher accuracy and lower coverage.

In general, if the system is optimized, the cross-point between the accuracy and the coverage plots represents an appropriate cutoff for the system to include or exclude predicted TRIs.

2.5.2. Measuring Accuracy and Coverage

To benchmark the predicted TRIs, we measure the accuracy and coverage of the predicted TRIs at specific cutoffs, we use the script: `run_regulogBenchmarking_<organism>_sprot`, where `organism` could be human, mouse, rat, yeast, and fly. This script calls our Python code that implements the following steps:

1. For each TRI in the test set, find the source experimental TRIs that give the TRI from the test set with the highest geometric mean of the FI product (**Note 30**). Assign this source experimental TRI and the geometric mean of the FI product to the TRI from the test set.
2. For each FI threshold cutoff ranging from 0.0 to 0.95, count the number of TRIs in the test set that are in TP and not in TP, and the FIs between the source TF and target TF and the source TFT and target TFTs at or above the cutoff.
3. Calculate the accuracy and coverage for each cutoff using the results from Step 2 and write the results into output files.

We use the above code to benchmark the accuracy and coverage of predicted TRIs for five organisms: human, mouse, rat, fly, and yeast. **Table 6.2** shows the numbers of pairs in the TP, TFs in TP, TFTs in TP, and TRIs in the test set of the five organisms. The source experimental TRIs came from the combination of all TRIs in TP from these organisms plus the experimental TRIs of *E. coli* and *B. subtilis* that had protein IDs in the Bioverse. For the benchmarking, we exclude TRIs in TP of the target organism from the source experimental TRIs. To obtain numbers for **Table 6.2** and generate the test

Table 6.2
Numbers of TP, TFs in TP, TFTs in TP, and TRIs in the test set of human, mouse, rat, fly, and yeast used for benchmarking

Organisms	TP	TFs in TP	TFTs in TP	TRIs in the test set (TFs in TP × TFTs in TP)
<i>S. cerevisiae</i>	136	38	104	3,952
<i>D. melanogaster</i>	171	72	65	4,680
<i>M. musculus</i>	309	109	159	17,331
<i>R. norvegicus</i>	46	18	36	648
<i>H. sapiens</i>	900	118	553	65,254

sets for the benchmarking, we use the script: run_generate-TestsetForBenchmarking. **Figure 6.4** shows the accuracy and coverage without any filtering for the test sets of human, mouse, rat, fly, and yeast. In general, the cross-point between the accuracy and coverage lines could be an appropriate cutoff. In this figure, the cross-points of the plots vary according to the available TRIs in TP of the target organisms.

To measure the errors of the method, we generate new test sets comprising randomly selected sets of 80% of the TRIs in TP. We repeat this benchmarking process 50 times. **Figure 6.5** shows

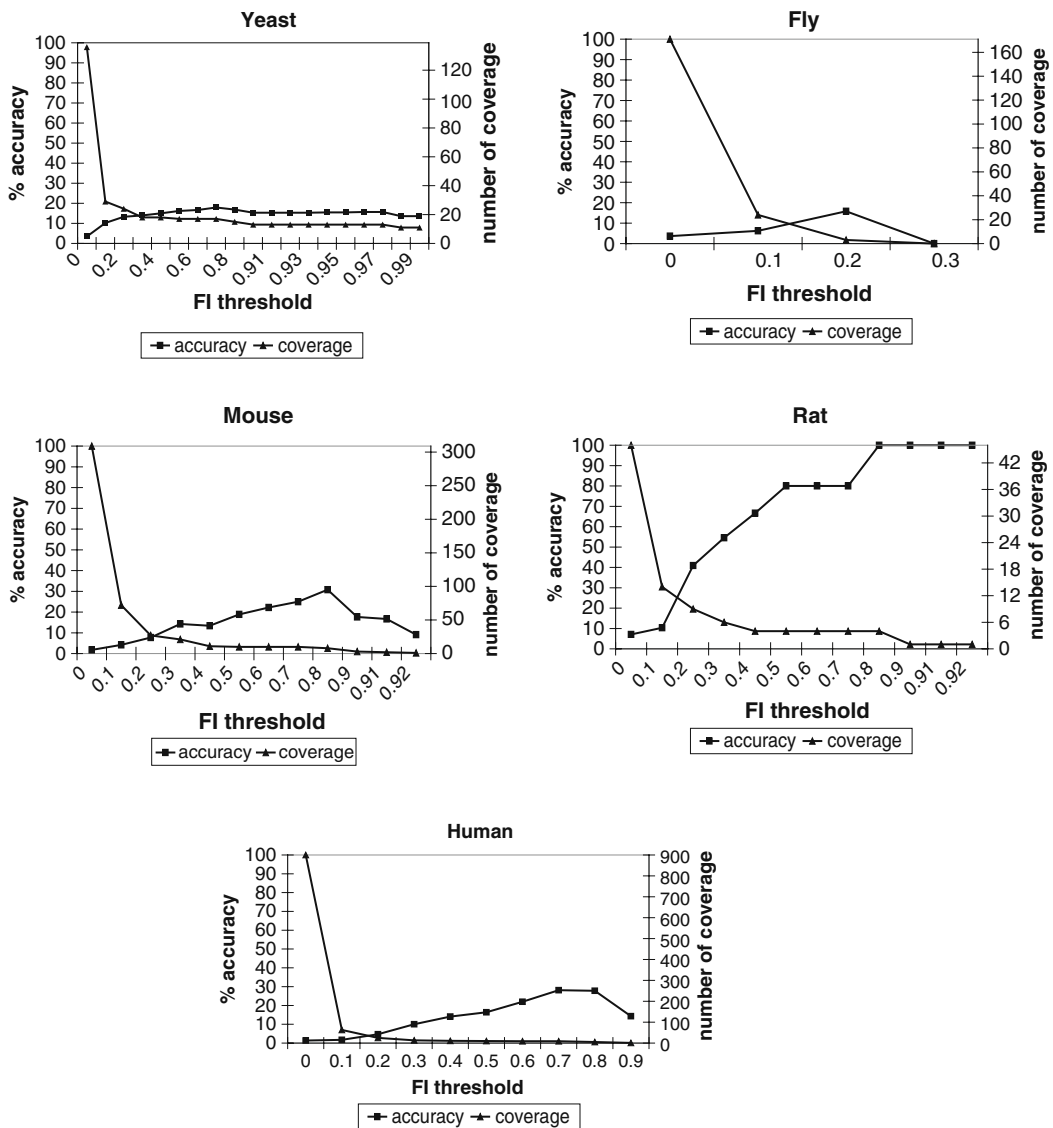


Fig. 6.4. Accuracy and coverage without any filtering for the test sets of yeast, fly, mouse, rat, and human.

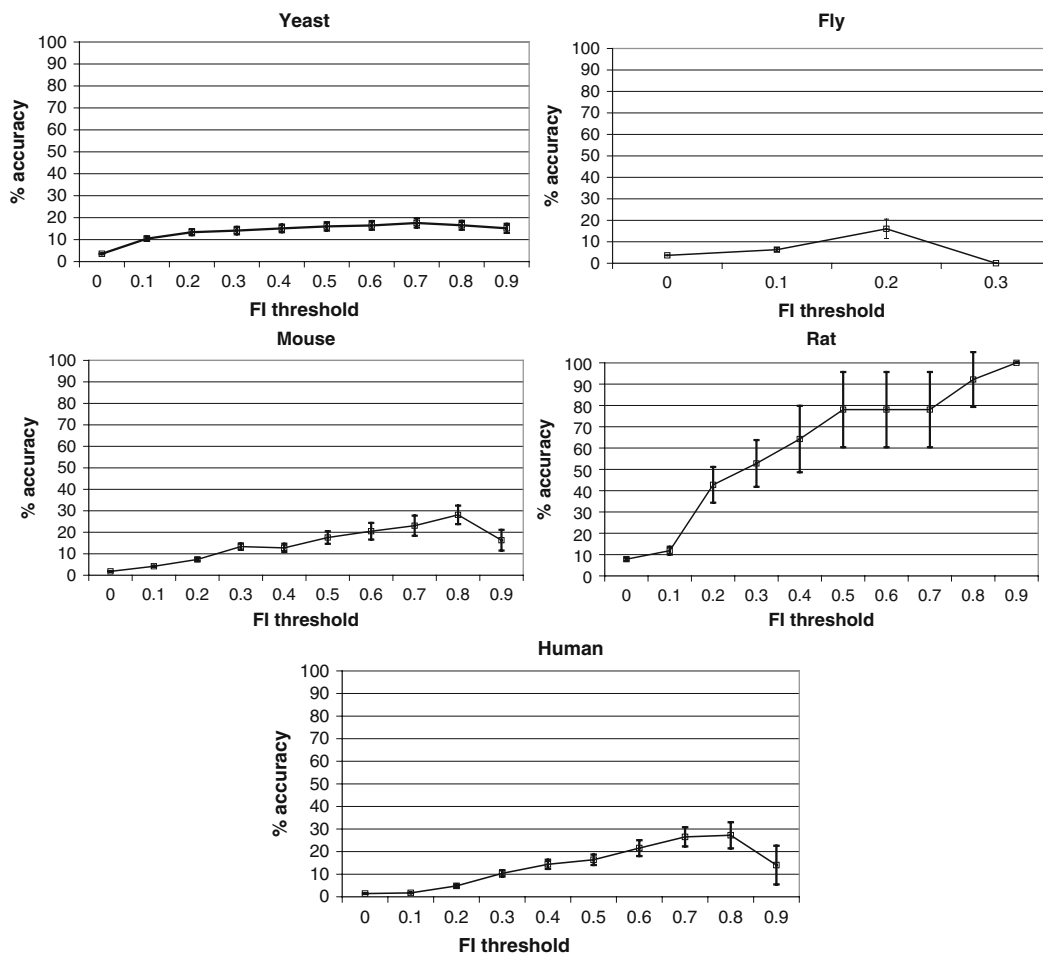


Fig. 6.5. Accuracy with error bars of the TRI prediction method without any filtering for the test sets of yeast, fly, mouse, rat, and human. *Data point* represents the mean of 50 bootstrapped data sets (randomly selected 80% of TRIs in TP) and *error bars* indicate the standard deviations above and below the mean.

the mean accuracy without any filters and the standard deviations below and above the mean at each FI threshold. Results from human, mouse, and yeast do not vary substantially among 50 tries. The method does not work well with fly data sets due to a low number of significant homologs within the available source experimental TRIs. It is likely that this also explains the results from fly in **Fig. 6.4**. The high standard deviations in the case of rat indicate heterogeneity for the rat TP. As we do not have a complete set of TRIs in TP for any target organism, the accuracy and coverage of predictions can only be evaluated as minimum accuracy and coverage.

2.5.3. Correctness of Benchmarking Method

We perform a sanity check to measure the correctness of the benchmarking method by including TRIs in the TP of the target organism as part of the source experimental TRIs and

calculate the accuracy and coverage using high threshold cutoffs ranging from 0.95 to 1.0. If the method is correct, then the predicted TRIs in TP should be the same as their source experimental TRIs and both accuracy and coverage will be 100%.

In the case of mouse TRI prediction, there is one predicted TRI (SP1→TTF-1) that is not a TRI in TP at the FI threshold cutoff ≥ 0.95 . This TRI is transferred from the SP1→TTF-1 in the source experimental TRIs of human. While this TRI is not in the source experimental TRIs of mouse, it is highly likely to be a real but not yet experimentally validated TRI.

In case of human TRI prediction, there are two predicted TRIs (ATF-2→HIST3H2A and HOXA5→HOXA5) not in the human TP at the FI threshold cutoff ≥ 0.95 . These TRIs are transferred from the ATF-2 → HIST1H2AC and the HOXA5→HOXA5 in the source experimental TRIs of human and mouse, respectively. HIST1H2AC is transferred to HIST3H2A with the very high cutoff value, as they are isoforms, but at the FI threshold cutoff ≥ 0.98 , only ATF-2→HIST3H2A remains (**Note 31**). While HOXA5→HOXA5 is not in the source experimental TRIs of human, it is also likely to be a real but not yet experimentally validated TRI.

The predicted TRI of yeast (ARS→ENO1), which is not in TP at the FI threshold ≥ 0.95 , is transferred from the ARS→ENO2 in the source experimental TRIs of yeast, where ENO2 and ENO1 are isoforms.

Overall, the results of our sanity check (shown in **Table 6.3**) for the five organisms confirm that the method is correct.

2.5.4. Effects of Filters

Figure 6.6 shows how different filters affect the accuracy of TRI prediction (**Note 32**). Judging from our results, the use of binding sites for filtering predictions improves the accuracy of TRI prediction for all organisms (except fly, due to the limited number of source experimental TRIs) (**Notes 33, 34**). The use of the functional annotation of “transcription factor” from the Bioverse as a filter slightly improves the prediction accuracy for all organisms except human, which might be caused by too narrow a search with the Bioverse options limited to “transcription factor” or unavailable. In the case of yeast, the use of protein localization as a filter slightly improves the accuracy of prediction at low-to-medium FI threshold cutoffs. **Table 6.4** shows the numbers of predicted TRIs of different filters for the five target organisms at the FI threshold cutoff ≥ 0.3 .

We investigate how protein families relate to the predicted TRIs by counting the number of TRIs for which TFs are sharing or not sharing protein families with their corresponding source TFs. **Table 6.5** lists the counting results for

Table 6.3
Results of the sanity checks for human, mouse, rat, fly, and yeast

	FI threshold	TRIs in TPs	TRIs not in TPs	% accuracy	% coverage
<i>D. melanogaster</i>	0.95	171	0	100	100
	0.96	171	0	100	100
	0.97	171	0	100	100
	0.98	171	0	100	100
	0.99	171	0	100	100
	1	171	0	100	100
<i>H. sapiens</i>	0.95	900	2	99.778	100
	0.96	900	2	99.778	100
	0.97	900	2	99.778	100
	0.98	900	1	99.889	100
	0.99	900	1	99.889	100
	1	900	0	100	100
<i>M. musculus</i>	0.95	309	1	99.677	100
	0.96	309	0	100	100
	0.97	309	0	100	100
	0.98	309	0	100	100
	0.99	309	0	100	100
	1	309	0	100	100
<i>R. norvegicus</i>	0.95	46	0	100	100
	0.96	46	0	100	100
	0.97	46	0	100	100
	0.98	46	0	100	100
	0.99	46	0	100	100
	1	46	0	100	100
<i>S. cerevisiae</i>	0.95	136	1	99.27	100
	0.96	136	0	100	100
	0.97	136	0	100	100
	0.98	136	0	100	100
	0.99	136	0	100	100
	1	136	0	100	100

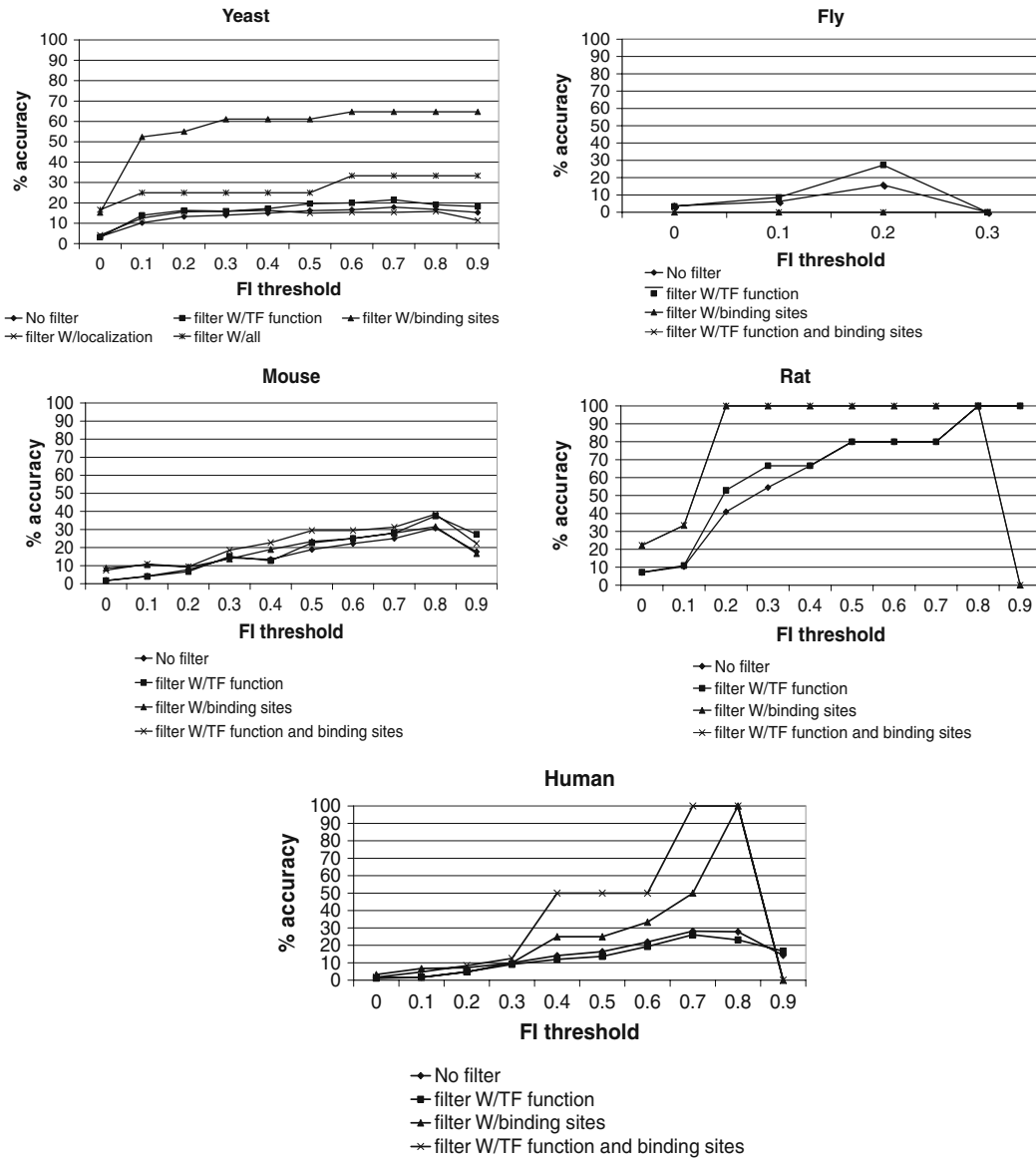


Fig. 6.6. Accuracy with no filters, accuracy with TF function filter, accuracy with binding site filter, accuracy with localization filter (in yeast only), accuracy with all filters for the test sets of yeast, fly, mouse, rat, and human.

the five organisms at the FI threshold cutoff ≥ 0.3 . The numbers of predicted TRIs that display sharing are the largest compared to the numbers of predicted TRIs not sharing or not having protein family information, for all FI threshold cutoffs for all organisms except yeast (data not shown here). In case of yeast, most of the predicted TRIs are the TRIs for which TFs or their source TFs do not have protein family information.

Table 6.4
Numbers of predicted TRIs (in TP, not in TP) with different filters at the FI threshold cutoff ≥ 0.3 of yeast, fly, mouse, rat, and human from the benchmarking process

Organisms	No filter	W/TF function filter	W/binding site filter	W/all filters
<i>S. cerevisiae</i>	18, 111	11, 58	11, 7	1, 3
<i>D. melanogaster</i>	0, 1	0, 1	0, 0	0, 0
<i>M. musculus</i>	21, 126	13, 73	7, 44	5, 22
<i>R. norvegicus</i>	6, 5	6, 3	2, 0	2, 0
<i>H. sapiens</i>	13, 117	7, 70	1, 9	1, 7

Table 6.5
Numbers of predicted TRIs of TP and not in TP for which TFs (1) share protein families with their corresponding source TFs, (2) have no overlapped protein families with their source TFs, and (3) have no information of protein families, with no filters, at the FI threshold cutoff ≥ 0.3

Organisms	TRIs in TP	TRIs in TP, sharing protein family	TRIs in TP, no sharing protein family	TRIs in TP, no protein family info.	TRIs not in TP	TRIs not in TP, sharing protein family	TRIs not in TP, no sharing protein family	TRIs not in TP, no protein family info.
<i>S. cerevisiae</i>	18	3	0	15	111	32	0	79
<i>D. melanogaster</i>	0	0	0	0	1	0	0	1
<i>M. musculus</i>	21	15	3	3	126	93	13	20
<i>R. norvegicus</i>	6	5	0	1	5	4	0	1
<i>H. sapiens</i>	13	10	0	3	117	100	3	14

3. Notes



1. Orthologs are defined as best-matching homologs between a source and a target organism.
2. Similarly, an interolog is defined as the pair of interacting proteins $A \leftrightarrow B$ in a source organism and its orthologous proteins $A' \leftrightarrow B'$ in a target organism (29).

3. The sources of experimental TRIs are limited, and the ways to access and gather them are varied. Among our source databases, RegulonDB is the only database that provides a way to download the TF to gene interactions in bulk. In the case of TRANSFAC[®], we needed to write a script using the *urllib* module in Python to fetch data from the http server of TRANSFAC[®]. In case of DBTBS, we resorted to a personal communication requesting the experimental TRIs from the authors.
4. The formats of the experimental TRIs differ from source to source. For instance, TRANSFAC[®] provides the experimental TRIs of eukaryotes via the records of transcription factors in html files, where each record will contain various information of the transcription factor and its regulating genes. DBTBS provides all experimental TRIs of *B. subtilis* in a single xml file. To handle these various formats, we wrote code for extracting the experimental TRIs from each specific source as described in **Section 2.1.1**.
5. We encountered the same problems for gathering and preparing the binding sites and binding sequences. The binding sites in TRANSFAC[®] came as a part of the transcription factor records and linked to their own records of specific binding sequences. Hence, we wrote code to extract the binding site accession numbers. Then we fetched the binding site records from the TRANSFAC[®] http server and parsed these records for the binding sequences. In case of DBTBS, the binding sites and binding sequences appeared in specific xml tags. Hence, we developed code to parse and extract them. As RegulonDB provides bulk download of the TF binding sites, we only needed to edit the format of the downloaded file.
6. To extend a set of experimental TRIs for an organism from different sources, we use the script: `run_appendTRIs_<organism>`, where examples of organisms are “caenorhabditis_elegans,” “homo_sapiens,” and “saccharomyces_cerevisiae.” This script calls a code that appends additional TRIs compiled from other sources listed in an input file (e.g., `./inputs/TRIs/caenorhabditis_elegans_tris_fileList.txt`, for *C. elegans*) to the available TRI file. For instance, the `run_appendTRIs_caenorhabditis_elegans` appends the TRIs compiled from WormBase (in file `./inputs/TRIs/WormBaseTRIs.csv`) to the compiled TRIs from TRANSFAC.
7. BIND becomes a component of BOND (Biomolecular Object Network Databank), which contains not only BIND but also GenBank data and related tools.

8. We get a PubMed identifier for a paper by searching the paper at Entrez PubMed <http://www.ncbi.nlm.nih.gov/entrez/query.fcgi?DB=pubmed>.
9. The lengths of the upstream regions are specific and limited for different sources. For instance, the UCSC Genome Browser provides the upstream regions of human, rat, mouse, and fly with lengths 1,000, 2,000, and 5,000 bps whereas SGD provides only 1,000 bps upstream regions. In the case of prokaryotes, we extracted the upstream regions of each organism from their complete genome sequences, with the length of 500 bps (**Notes 35, 36**).
10. In this work, we do not take the directions (i.e., forward, reverse) of the strand of the upstream regions into account. Also, we do not consider the possible binding sites at the downstream region. At present, we are interested only in predicting the transcriptional regulatory interactions for which transcription factors bind to specific sequences in the upstream regions of a target gene. Nevertheless, the overall methods described for TRI prediction should be usable for these extensions as these affect only the scanning of the binding sequences during the filtering process. The data preparation should be extended for the downstream regions, and the scanning of binding sites should evaluate the reverse strand and downstream regions.
11. In the case of *S. cerevisiae*, instead of getting the upstream regions via the ftp server as described in **Section 2.1.2**, you might follow the following steps:
 - Go to <http://www.yeastgenome.org/> (44).
 - On the left side of this main page, in section “Download Data,” select “Batch Download.” A page “SGD Batch Download Tool” will appear. Specify the input chromosome on the right-hand side of Step 1, and then specify the type of data that you would like to retrieve in Step 2. Under the “Sequence data” section, check box “Genomic DNA + 1 kb upstream and 1 kb downstream of flanking sequence,” and click the submit button.
 - After getting the result file for each chromosome, concatenate these files together for the upstream regions of all genes in the complete genome. Then, use the same script: `run_extractUpstreamRegions_1000bp_saccharomyces_cerevisiae` to parse, extract, and transform this file into the upstream region file from *S. cerevisiae*, ready for use by the system.
12. The upstream region files downloaded from UCSC Genome Browser contain only the upstream regions from transcription starts annotated separately from the coding initiation region.

So, they are not the complete sets of upstream regions. An alternative way to compile the upstream regions of genes in an organism is to find the location of the genes in the complete genome sequence and extract the sequence in front of the genes starting from their transcription start sites as the upstream regions.

13. In the case of *H. sapiens*, *M. musculus*, *R. norvegicus*, and *D. melanogaster*, one might use the ftp server instead of the http server:
 - Go to <ftp://hgdownload.cse.ucsc.edu/goldenPath/>.
 - Use the code in parentheses at the first line of a specific genome box at <http://hgdownload.cse.ucsc.edu/downloads.html> (e.g., hg18 for human genome) to select the directory under the <ftp://hgdownload.cse.ucsc.edu/goldenPath/>. Under this directory, select “bigZips” directory and save files upstreamxxx.zip into a local directory. The xxx represents the number of base pairs of each upstream region. Use our script: `run_extract UpstreamRegions_1000 bp_homo_sapiens` to parse, extract, and transform the saved files into the format ready for use by the system.
14. TIGR uses the results from IRGSP. Hence, the upstream regions of rice downloaded from TIGR are of the cultivar Nipponbare of *Oryza sativa* L. ssp. *japonica* (57). The first draft sequence of *O. sativa* L. ssp. *indica* was also available and published in the same journal (70).
15. The naming systems of the upstream regions of organisms are different from source to source. We needed to find the appropriate mapping from a specific ID system to the protein IDs in the Bioverse. The complexity in finding the mapping varied according to different sources. For instance, the upstream regions of eukaryotes (i.e., human, mouse, rat) downloaded from the UCSC Genome Browser were identified by RefSeq accession numbers for nucleotide sequences. However, we only had the mapping of NCBI GenBank Identifiers (GIs) to protein IDs in the Bioverse for these organisms. Hence, we needed to find the mapping from these RefSeq numbers to the GIs. On the other hand, the upstream regions of yeast, fly, worm, and *Arabidopsis* had been annotated by their specific ID systems already in the Bioverse. Hence, finding the mapping for these organisms was less complex. In case of rice, as TIGR uses TIGR_LOCUS as the main identifier for the rice genome to refer to the upstream regions, genes, and proteins, we needed to find the mapping from TIGR_LOCUS IDs to protein IDs in the Bioverse.

16. Several names of TFs and TFTs in the source experimental TRIs are not mapped with any intermediate ID system. Hence, the TRIs with these TFs and/or TFTs will be discarded, such that several source TRIs are lost during the mapping process.
17. We could improve the quality of the name mappings by finding and adding the synonyms of the common names from different sources to the name ID mapping file.
18. Building name mapping from TFs, TFTs to protein IDs in the Bioverse for *H. sapiens* is the most complicated. We encountered the following problems and limitations during the process of building the name mapping:
 - The naming of human genes is still not well defined. Even though several sources of human genes with common names are available, some of them are not updated and some others are obsolete. Sources of common names for *H. sapiens* are GenBank, the synonyms field associated with each transcription factor information files compiled from TRANSFAC[®], OMIM (59), Entrez Genes (71), and HUGO (58).
 - Human genes are much more complex than yeast. Several of them have the same common names but different protein products according to different isoforms. Hence, a straightforward many-to-one mapping of a common name and its synonyms to a specific intermediate ID (i.e., systematic ORF name from SGD) and to a protein ID in the Bioverse as in case of yeast is not always true in human.
19. Even though the name mapping could be refined with the synonyms of common genes from different sources, in general, these will not be complete. The better and more reliable mapping from the common names of TF and TFT in the experimental TRIs to protein IDs in the Bioverse uses sequence mapping. However, this is not applicable because this method involves protein sequences of all TFs and TFTs in all experimental TRIs. Nevertheless, TRANSFAC[®] provides only the protein sequences of TFs but not of TFTs, while other sources of experimental TRIs do not provide protein sequences. To get protein sequences for these TFs and TFTs, we return to the name mapping problem.
20. At present, we have only the protein localization of *S. cerevisiae* from two public databases, as described in **Section 2.2.1**. Also, in our current assumption for filtering using protein localization, a predicted TRI will be discarded if its TF and TFT do not share localization. This assumption might be too restrictive and needs further investigation.

Table 6.6
Public databases of protein subcellular localization from experiments

Database	Description	URL
UniProtKB/Swiss-Prot (76)	An annotated protein sequence database	http://www.ebi.ac.uk/swissprot/FTP/ftp.html
Comprehensive Yeast Genome Database (CYGD) (78)	MIPs <i>Saccharomyces cerevisiae</i> genome database	http://mips.gsf.de/genre/proj/yeast/index.jsp
MitoP2 (79)	A database for mitochondria-related genes, proteins, and diseases	http://www.mitop.de:8080/mitop2/
LOCATE (80)	A curated database of membrane, organization and subcellular localization mouse proteins of RIKEN FANTOM3	http://locate.imb.uq.edu.au/

21. In addition to TRIPLES and YEAST GFP mentioned in **Section 2.2.1**, we list additional public databases of experimental protein localization compiled from a literature search in **Table 6.6** and sources of protein localization based on the predictions in **Table 6.7**. Beside these two tables, additional systems and programs for protein subcellular localization can be found at <http://www.molecularstation.com/protein/bioinformatics/subcellularlocalization/>.
22. Besides protein families compiled from TRANSFAC[®], we list other public databases of protein families in **Table 6.8**. Additional resources of protein families can be found at <http://www.proweb.org/other.html>.
23. At present, we have only protein families compiled from TRANSFAC[®], as described in **Section 2.2.2**. Our current filtering method using protein families will discard all predicted TRIs for which the related TFs do not share any protein families with their corresponding TFs from source TRIs. Hence, with protein families compiled only from TRANSFAC[®], the filtering might discard the predicted TRIs that are real.
24. In addition to using protein families for filtering the predicted TRIs as described in **Section 2.2.2**, we might utilize the combination of protein domains and domain architectures (72–75) in finding the similarity between two protein sequences. Also, instead of assigning the same weights for all overlapped locations between two aligned protein sequences,

Table 6.7
Sources of protein subcellular localization from prediction

Source	Description	URL
PSORT (81)	Predicting protein subcellular localization based on stored rules and prediction of sorting signals	http://psort.hgc.jp/
Yeast Protein Localization Server (82)	Predicting subcellular location of proteins in yeast using Bayesian formalism	http://bioinfo.mbb.yale.edu/genome/localize/
LOC3d (83)	A database of predicted subcellular localization for eukaryotic PDB chains	http://cubic.bioc.columbia.edu/db/LOC3d/
TargetP1.1 (84)	Predicting subcellular localization of protein in eukaryotes based on the predicted presence of N-terminal pre-sequences, chloroplast transit peptide (cTP), mitochondrial targeting peptide (mTP), or secretory pathway signal peptide (SP)	http://www.cbs.dtu.dk/services/TargetP/
SubLoc v1.0 (85)	A system for predicting protein subcellular localization using Support Vector Machine (SVM)	http://www.bioinfo.tsinghua.edu.cn/SubLoc/

Table 6.8
Public databases of protein domains and families

Database	Description	URL
InterPro (86, 87)	A resource of protein families, domains and functional sites, integrated from other databases such as Pfam, PROSITE, PRINTS, etc.	http://www.ebi.ac.uk/interpro/
Pfam (88)	A database of curated protein domains and families based on multiple sequence alignments and hidden Markov models	http://www.sanger.ac.uk/Software/Pfam/
PROSITE (89)	A database of protein families and domains that consists of biologically significant sites, patterns, and profiles used for identifying a family for a new protein	http://www.expasy.org/prosite/

(continued)

Table 6.8 (continued)

Database	Description	URL
ProDom (90)	A database of protein domain families automatically generated from Swiss-Prot and TrEMBL databases	http://prodom.prabi.fr/prodom/current/html/home.php
BLOCKS (91, 92)	A database of aligned ungapped segments derived from the most highly conserved regions in groups of proteins	http://blocks.fhcrc.org/
PRINTS (93)	A collection of protein fingerprints, where each is a group of conserved motifs used to classify a protein family	http://www.bioinf.man.ac.uk/dbbrowser/PRINTS/
TIGRFAMs (94)	A collection of curated protein families that consists of various models including, multiple sequence alignments, hidden Markov models (HMMs), Gene Ontology (GO) assignments, and literature references, for instance.	http://www.tigr.org/TIGRFAMs/
SYSTEMS (95)	A database of protein families based on graph-based algorithms for protein sequences partitioning, clustering, and searching	http://systems.molgen.mpg.de/
SCOP (96, 97)	A database of proteins ordered by structural classification; protein domains are classified into families, superfamilies, folds, and classes	http://scop.mrc-lmb.cam.ac.uk/scop/
SMART (98)	A web-based tool and database, which allows the identification of signaling domains, the genetically mobile domains, and the analysis of domain architectures	http://smart.embl-heidelberg.de/
SUPERFAMILY (99, 100)	A database of hidden Markov models (HMMs) of known structures with protein domain classification based on SCOP	http://supfam.org/SUPERFAMILY/
CATH (101)	A database of protein domain structures, which categorizes proteins at four levels: Class (C), Architecture (A), Topology (T), and Homologous superfamily (H)	http://www.cathdb.info/

(continued)

Table 6.8 (continued)

Database	Description	URL
Gene3D (102)	A database of combined structures, functions, and evolutions of proteins, with HMMs based on CATH domain families, for structural annotation	http://gene3d.biochem.ucl.ac.uk/Gene3D/
PIRSF (103)	A super family classification system based on the relationships of protein evolutions	http://pir.georgetown.edu/pirsf/
PANTHER (104, 105)	A database of protein families, subfamilies, functions, pathways, and sequence and function evolutions	http://www.pantherdb.org/
CDD (106)	A conserved domain database at NCBI for protein classification	http://130.14.29.110/Structure/cdd/cdd.shtml

we need to give more weight for the locations that are considered protein domains. This higher similarity specificity should help to improve the accuracy of prediction.

25. Instead of getting protein sequences from the Bioverse via an XML RPC server, we might get protein sequences from other sources such as protein database at NCBI, UniProt (the universal protein resource) (76), TAIR for *A. thaliana*, TIGR Rice Genome Annotation for *O. sativa*, WormBase for *C. elegans*, FlyBase for *D. melanogaster*, and SGD for *S. cerevisiae*.
26. For BLAST practical usage, see BLAST tutorial at <http://www.ncbi.nlm.nih.gov/Education/BLASTinfo/tut1.html> for additional information.
27. For PSI-BLAST practical usage, see PSI-BLAST tutorial at <http://www.ncbi.nlm.nih.gov/Education/BLASTinfo/psi1.html> for additional information.
28. Raw score, “S”, is calculated as the sum of the substitution and gap scores (source: http://www.ncbi.nlm.nih.gov/Education/BLASTinfo/Blast_output.html).
29. We define accuracy as the fraction of TRIs predicted by the system that are in true positive set (TP) over the total predicted TRIs at a specific cutoff. As we did not have a set of false positives (FP), we define the test set of the TRIs that includes all combinations of individual TF to individual TFTs in the TP set. This means that if we have 121 TRIs in the TP which consists of 49

TFs and 83 TFTs, then the test set will consist of $49 \times 83 = 4,067$ TRIs. Note that it is possible that some TRIs in the 4,067 that are not in the TP might be real. Hence, what we have for accuracy calculation is minimum coverage. To improve accuracy and coverage, we need to find a gold standard of TP for which TRIs are known with high confidence so that any other TRI is not possible among the set of their TFs and TFTs.

30. The FI product is the multiplication of fraction identity (FI) between a source TF and a target TF and a source TFT to a target TFT. The fraction identity is the number of identical amino acids that are overlapped between the aligned source and the target TFs (or source and target TFTs) over the total number of amino acids of the target TF (or target TFT).
31. The predicted TRIs that have high similarity of interaction but are not in the TP might come from isoforms. To clean up accuracy and coverage at the high similarity of interactions for benchmarking, we omit the predicted TRIs resulting from the isoforms of target TF and TFT.
32. The available experimental TRIs are incomplete. Even though we tried to gather the experimental TRIs of each source organism from different sources such as public databases and supplemental data from the literature, there is no way to compile the complete set of true positives for a target organism. This resulted in lower accuracy and coverage in benchmarking the TRI predictions for almost all target organisms. This is a limitation of the available data set. In fact, predictions that are not in our TP might be real TRIs.

To alleviate this limitation, we attempt to filter out the predicted TRIs using additional data sets such as binding sites, binding sequences, protein localization, protein families, and functional annotation. Also, several predicted TRIs are the same among organisms (**Note 37**). Even though they are not yet verified by experiments, nor are they in the TPs, we consider them as real interactions that should be removed from the set of predicted TRIs not in the TP.

33. The available binding sites and binding sequences are incomplete. We cannot determine a predicted TRI as false even though the upstream region of the TFT of the predicted TRI does not have the binding sites and binding sequences of the TF of its source experimental TRI. It is possible that the TF might have alternative sites and sequences that are not yet studied. Hence, we mainly use the available binding sites and binding sequences for improving the confidence of prediction.
34. The location of binding sites and their sequences do not need to be exactly the same between the TF of the source experimental TRI and the TF of a predicted TRI. Hence, we relaxed

this restriction using 80% matching for scanning the binding sequences with the varied lengths of upstream regions of target TFT in the predicted TRI. Note that the scanning of binding sequences of *B. subtilis* needed a special treatment, such that some of its binding sequences required exact matching while others could be relaxed.

35. We could change this length of upstream regions and rerunning the extraction.
36. RegulonDB also provides promoters of *E. coli* K-12. However, we have not integrated these promoters as part of the extracted upstream regions of *E. coli*.
37. A predicted TRI in a target organism is considered the same as a source experimental TRI if its TF' and TFT' have the same names as of the TF and TFT of the source experimental TRI.
38. Databases at NCBI such as Nucleotide, Protein have been providing XML as an alternative formats for their downloadable data.

4. Conclusions

Data preparation and integration is one of the major tasks in our prediction of TRIs by a homology-based approach. As data from different sources have different formats, we need to handle them differently. In this chapter, we provide a set of Python programs and shell scripts that help to retrieve and manipulate data from various sources. Definitely, these make this process easier. However, public data sets are being released every day, and they might have rearrangements of data formats and identifiers. Hence, in the long term it would be better if various sources provide their data sets in a standard data exchange format like XML (**Note 38**) and/or provide standard interfaces (e.g., SOAP, RPC, Web Services) for remote programs to be able to automatically interconnect and work together with each other. In terms of name mapping, while different sources might have their own identification systems, it would be much easier for users if they also provided a mapping from their ID to a standard ID. Hence, while what we presented in this chapter seems straightforward, it becomes very complicated in terms of bookkeeping, and this is very rarely talked about explicitly in papers that describe results like these.

The compiled experimental TRIs and the predicted TRIs can be integrated with protein–protein interactions (PPIs) and other data types such as gene expression for investigating specific pathways. **Figure 6.7** shows an example of the integrated network between TRIs and PPIs for investigating the regulation of CBF1

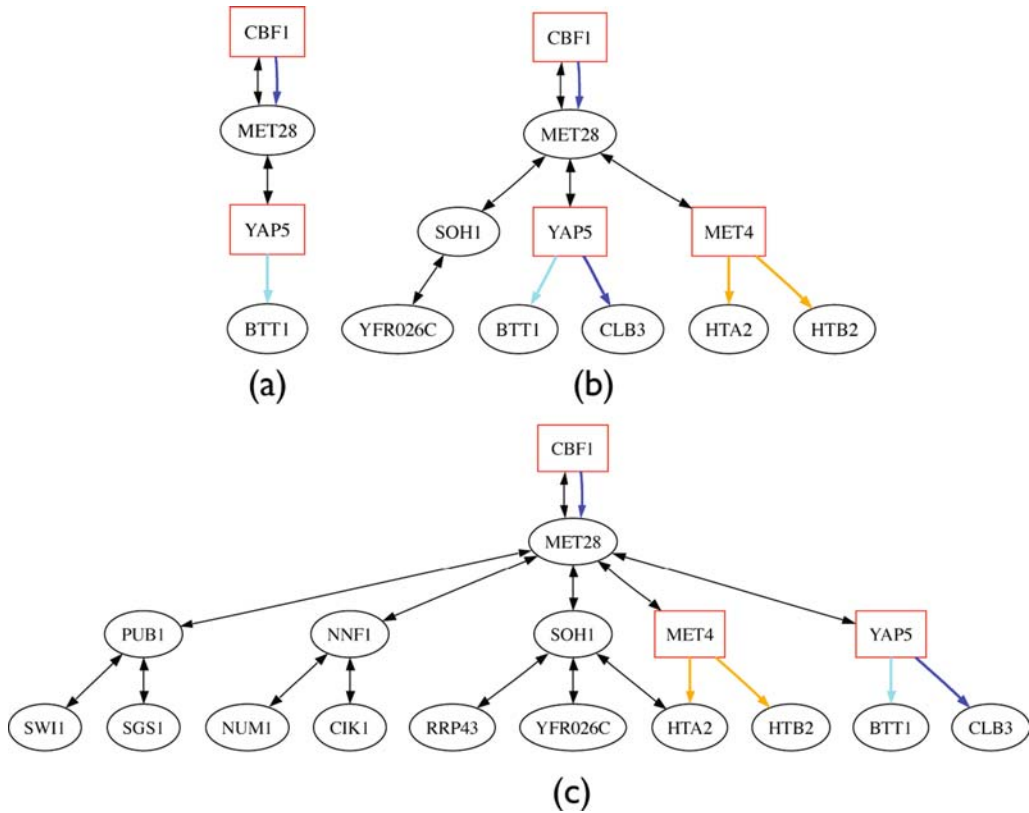


Fig. 6.7. The integrated networks that include experimentally verified TRIs from TRANSFAC and (a) TRIs from high-throughput experiment (1) with p -value ≤ 0.005 and PPIs in the Bioverse with confidence = 1, (b) TRIs from high-throughput experiment (1) with p -value ≤ 0.01 and PPIs in the Bioverse with confidence = 1, and (c) TRIs from high-throughput experiment (1) with p -value ≤ 0.01 and PPIs in the Bioverse with confidence ≥ 0.8 . All these three networks had been filtered with differentially expressed genes listed in (2). An arrow with heads on both ends represents a PPI and an arrow with a head only at its end represents a TRI. A cyan, blue, and orange TRI has p -value ≤ 0.001 , 0.005 , and 0.01 , respectively.

to MET28 (CBF1→MET28) and their related genes, proteins, and interactions, in yeast cell cycle phase S. **Figure 6.7a, b and c** represent the integrated networks with different settings and filtered by differentially expressed genes downloaded from Luscombe et al. (77).

Acknowledgments

This work was supported in part by a Searle Scholar Award and NSF Grant DBI-0217241 to R.S., and the University of Washington’s Advanced Technology Initiative in Infectious

Diseases. Also, it is supported in part by the National Center for Genetic Engineering and Biotechnology (BIOTEC), National Science Technology & Development Agency (NSTDA), Thailand.

References

1. Lee, T.I., et al., Transcriptional regulatory networks in *Saccharomyces cerevisiae*. *Science*, 2002. **298**(5594): 799–804.
2. Deplancke, B., et al., A gene-centered *C. elegans* protein-DNA interaction network. *Cell*, 2006. **125**(6): 1193–1205.
3. McDermott, J., et al., *BIOVERSE*: Enhancements to the framework for structural, functional and contextual modeling of proteins and proteomes. *Nucl. Acids Res.*, 2005. **33**(suppl_2): W324–W325.
4. H Caron, et al., The Human Transcriptome Map reveals a clustering of highly expressed genes in chromosomal domains. *Science*, 2001. **291**: 1289–1292.
5. Shen-Orr, S.S., et al., Network motifs in the transcriptional regulation network of *Escherichia coli*. *Nat. Genet.*, 2002. **31**(1): 64–68.
6. Martinez-Antonio, A. and J. Collado-Vides, Identifying global regulators in transcriptional regulatory networks in bacteria. *Curr. Opin. Microbiol.*, 2003. **6**(5): 482–489.
7. Harbison, C.T., et al., Transcriptional regulatory code of a eukaryotic genome. *Nature*, 2004. **431**(7004): 99.
8. Proft, M., et al., Genomewide identification of Sko1 target promoters reveals a regulatory network that operates in response to osmotic stress in *Saccharomyces cerevisiae*. *Eukaryot. Cell*, 2005. **4**(8): 1343–1352.
9. Sharma, M.R., et al., Transcriptional networks in a rat model for nonalcoholic fatty liver disease: A microarray analysis. *Exp. Mol. Pathol.*, 2006. [Epub ahead of print].
10. Reymann, S. and J. Borlak, Transcriptome profiling of human hepatocytes treated with Aroclor 1254 reveals transcription factor regulatory networks and clusters of regulated genes. *BMC Genomics*, 2006. **7**(1): 217.
11. Makita, Y., et al., *DBTBS*: Database of transcriptional regulation in *Bacillus subtilis* and its contribution to comparative genomics. *Nucl. Acids Res.*, 2004. **32**(suppl_1): D75–D77.
12. Matys, V., et al., TRANSFAC(R) and its module TRANSCOMP(R): Transcriptional gene regulation in eukaryotes. *Nucl. Acids Res.*, 2006. **34**(suppl_1): D108–D110.
13. Salgado, H., et al., RegulonDB (version 5.0): *Escherichia coli* K-12 transcriptional regulatory network, operon organization, and growth conditions. *Nucl. Acids Res.*, 2006. **34**(suppl_1): D394–D397.
14. Segal, E., et al., Module networks: Identifying regulatory modules and their condition-specific regulators from gene expression data. *Nat. Genet.*, 2003. **34**(2): 166–176.
15. Pilpel, Y., P. Sudarsanam, and G. M. Church, Identifying regulatory networks by combinatorial analysis of promoter elements. *Nat. Genet.*, 2003. **29**: 153–159.
16. Yeager-Lotem, E., et al., Network motifs in integrated cellular networks of transcription-regulation and protein-protein interaction. *PNAS*, 2004. **101**(16): 5934–5939.
17. Yu, T. and K.-C. Li, Inference of transcriptional regulatory network by two-stage constrained space factor analysis. *Bioinformatics*, 2005. **21**(21): 4033–4038.
18. Zhang, L., et al., Motifs, themes and thematic maps of an integrated *Saccharomyces cerevisiae* interaction network. *J. Biol.*, 2005. **4**(2): 6.
19. Jiang, R., et al., Network motif identification in stochastic networks. *PNAS*, 2006. **103**(25): 9404–9409.
20. Mandel-Gutfreund, Y. and H. Margalit, Quantitative parameters for amino acid-base interaction: Implications for prediction of protein-DNA binding sites. *Nucl. Acids Res.*, 1998. **26**(10): 2306–2312.
21. Luscombe, N.M. and J.M. Thornton, Protein–DNA interactions: Amino acid conservation and the effects of mutations on binding specificity. *J. Mol. Biol.*, 2002. **320**(5): 991–1009.
22. Kato, M., et al., Identifying combinatorial regulation of transcription factors and binding motifs. *Genome Biol.*, 2004. **5**(8): R56.
23. Morozov, A.V., et al., Protein-DNA binding specificity predictions with structural models. *Nucl. Acids Res.*, 2005. **33**(18): 5781–5798.
24. Gertz, J., et al., Discovery, validation, and genetic dissection of transcription factor binding sites by comparative and functional genomics. *Genome Res.*, 2005. **15**(8): 1145–1152.

25. Tompa, M., et al., Assessing computational tools for the discovery of transcription factor binding sites. *Nat. Biotechnol.*, 2005. **23**: 137–144.
26. GuhaThakurta, D., Computational identification of transcriptional regulatory elements in DNA sequence. *Nucl. Acids Res.*, 2006. **34**(12): 3585–3598, doi: 10.1093/nar/gkl372.
27. Yu, H., et al., Annotation transfer between genomes: Protein-protein interologs and protein-DNA regulogs. *Genome Res.*, 2004. **14**(6): 1107–1118.
28. Du, W., et al., RBF, a novel RB-related gene that regulates E2F activity and interacts with cyclin E in *Drosophila*. *Genes Dev.*, 1996. **10**(10): 1206–1218.
29. Walhout, A.J.M., et al., Protein interaction mapping in *C. elegans* using proteins involved in vulval development. *Science*, 2000. **287**: 116–122.
30. Matthews, L.R., et al., Identification of potential interaction networks using sequence-based searches for conserved protein-protein interactions or “Interologs”. *Genome Res.*, 2001. **11**(12): 2120–2126.
31. Lehner, B. and A.G. Fraser, A first-draft human protein-interaction map. *Genome Biol.*, 2004. **5**(9): R63.1–9.
32. Huang, T.-W., et al., POINT: A database for the prediction of protein-protein interactions based on the orthologous interactome. *Bioinformatics*, 2004. **20**(17): 3273–3276.
33. Kemmer, D., et al., Ulysses – an application for the projection of molecular interactions across species. *Genome Biol.*, 2005. **6**(12): R106.
34. Brown, K.R. and I. Jurisica, Online predicted human interaction database. *Bioinformatics*, 2005. **21**(9): 2076–2082.
35. von Mering, C., et al., STRING: Known and predicted protein-protein associations, integrated and transferred across organisms. *Nucl. Acids Res.*, 2005. **33**(suppl_1): D433–D437.
36. Zhu, J. and M.Q. Zhang, *SCPD: A promoter database of the yeast *Saccharomyces cerevisiae**. *Bioinformatics*, 1999. **15**(7): 607–611.
37. Bader, G.D., D. Betel, and C.W. Hogue, *BIND: The biomolecular interaction network database*. *Nucl. Acids Res.*, 2003. **31**(1): 248–250.
38. Alfarano, C., et al., The biomolecular interaction network database and related tools 2005 update. *Nucl. Acids Res.*, 2005. **33**(suppl_1): D418–D424, doi: 10.1093/nar/gki051.
39. Chen, N., et al., WormBase: A comprehensive data resource for *Caenorhabditis* biology and genomics. *Nucl. Acids Res.*, 2005. **33**(suppl_1): D383–D389.
40. Schwarz, E.M., et al., WormBase: Better software, richer content. *Nucl. Acids Res.*, 2006. **34**(suppl_1): D475–D478, doi: 10.1093/nar/gkj061.
41. Hayakawa, J., et al., Identification of promoters bound by c-Jun/ATF2 during rapid large-scale gene activation following genotoxic stress. *Mol. Cell*, 2004. **16**(4): 521.
42. Kim, J., et al., Mapping DNA-protein interactions in large genomes by sequence tag analysis of genomic enrichment. *Nat. Meth.*, 2005. **2**(1): 47.
43. Kim, T.H., et al., Direct isolation and identification of promoters in the human genome. *Genome Res.*, 2005. **15**(6): 830–839.
44. Hong, E.L., et al., *Saccharomyces* Genome Database. <ftp://ftp.yeastgenome.org/yeast/September>, 2006.
45. Hinrichs, A.S., et al., The UCSC genome browser database: Update 2006. *Nucl. Acids Res.*, 2006. **34**(suppl_1): D590–D598.
46. Michael, J.M., Initial sequencing and analysis of the human genome. *Nature*, 2001. **409**(6822): 860.
47. Waterston, R.H., et al., Initial sequencing and comparative analysis of the mouse genome. *Nature*, 2002. **420**(6915): 520.
48. Gibbs, R.A., et al., Genome sequence of the Brown Norway rat yields insights into mammalian evolution. *Nature*, 2004. **428**(6982): 493.
49. Adams, M.D., et al., The genome sequence of *Drosophila melanogaster*. *Science*, 2000. **287**(5461): 2185–2195.
50. The *C. elegans* Sequencing Consortium, Genome sequence of the nematode *C. elegans*: A platform for investigating biology. *Science*, 1998. **282**(5396): 2012–2018.
51. Rhee, S.Y., et al., The Arabidopsis Information Resource (TAIR): A model organism database providing a centralized, curated gateway to Arabidopsis biology, research materials and community. *Nucl. Acids Res.*, 2003. **31**(1): 224–228.
52. The Arabidopsis Genome, I., Analysis of the genome sequence of the flowering plant *Arabidopsis thaliana*. *Nature*, 2000. **408**(6814): 796.
53. Theologis, A., et al., Sequence and analysis of chromosome 1 of the plant *Arabidopsis thaliana*. *Nature*, 2000. **408**(6814): 816.

54. European Union Chromosome 3 Arabidopsis Genome Sequencing, C., R. The Institute for Genomic, and D.N.A.R.I. Kazusa, Sequence and analysis of chromosome 3 of the plant *Arabidopsis thaliana*. *Nature*, 2000. **408**(6814): 820.
55. Kazusa, D.N.A.R.I., et al., Sequence and analysis of chromosome 5 of the plant *Arabidopsis thaliana*. *Nature*, 2000. **408**(6814): 823.
56. Yuan, Q., et al., The institute for genomic research Osa1 rice genome annotation database. *Plant Physiol.*, 2005. **138**(1): 18–26.
57. Goff, S.A., et al., A draft sequence of the rice genome (*Oryza sativa* L. ssp. *japonica*). *Science*, 2002. **296**(5565): 92–100.
58. HUGO Gene Nomenclature Committee http://www.genenames.org/data/gdlw_index.html September 2006.
59. Online Mendelian Inheritance in Man, OMIM (TM). McKusick-Nathans Institute of Genetic Medicine, Johns Hopkins University (Baltimore, MD) and National Center for Biotechnology Information, National Library of Medicine (Bethesda, MD), {September 2006}. World Wide Web URL: <http://www.ncbi.nlm.nih.gov/omim/>
60. Huh, W.-K., et al., Global analysis of protein localization in budding yeast. *Nature*, 2003. **425**: 686–691.
61. Drawid, A., R. Jansen, and M. Gerstein, Genome-wide analysis relating expression level with protein subcellular localization. *Trends Genet.*, 2000. **16**(10): 426.
62. Ross-Macdonald, P., et al., Large-scale analysis of the yeast genome by transposon tagging and gene disruption. *Nature*, 1999. **402**(6760): 413.
63. Kumar, A., et al., TRIPLES: A database of gene function in *Saccharomyces cerevisiae*. *Nucl. Acids Res.*, 2000. **28**(1): 81–84.
64. Tatiana, T.A. and T.L. Madden, Blast 2 sequences – a new tool for comparing protein and nucleotide sequences. *FEMS Microbiol Lett.*, 1999. **174**: 247–250.
65. Altschul, S.F., et al., Basic local alignment search tool. *J. Mol. Biol.*, 1990. **215**(3): 403–410.
66. Altschul, S.F., et al., Gapped BLAST and PSI-BLAST: A new generation of protein database search programs. *Nucl., Acids Res.*, 1997. **25**(17): 3389–3402.
67. Smith, T.F. and M.S. Waterman, Identification of common molecular subsequences. *J. Mol. Biol.*, 1981. **147**(1): 195–197.
68. Pearson, W.R., Searching protein sequence libraries: Comparison of the sensitivity and selectivity of the Smith-Waterman and FASTA algorithms. *Genomics*, 1991. **11**(3): 635–650.
69. Thompson, J.D., D.G. Higgins, and T.J. Gibson, CLUSTAL W: Improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. *Nucl. Acids Res.*, 1994. **22**(22): 4673–4680.
70. Yu, J., et al., A draft sequence of the rice genome (*Oryza sativa* L. ssp. *indica*). *Science*, 2002. **296**(5565): 79–92.
71. Donna Maglott, et al., Entrez Gene: Gene-centered information at NCBI. *Nucl. Acids Res.*, 2005. **33**(Database): D54–D58.
72. Bashton, M. and C. Chothia, The geometry of domain combination in proteins. *J. Mol. Biol.*, 2002. **315**(4): 927.
73. Bjorklund, A.K., et al., Domain rearrangements in protein evolution. *J. Mol. Biol.*, 2005. **353**(4): 911.
74. Geer, L.Y., et al., CDART: Protein homology by domain architecture. *Genome Res.*, 2002. **12**(10): 1619–1623, doi: 10.1101/gr.278202,.
75. Hegyi, H. and M. Gerstein, Annotation transfer for genomics: Measuring functional divergence in multi-domain proteins. *Genome Res.*, 2001. **11**(10): 1632–1640, doi: 10.1101/gr.183801.
76. The UniProt Consortium, The Universal Protein Resource (UniProt). *Nucl. Acids Res.*, 2007. **35**(suppl_1): D193–D197, doi: 10.1093/nar/gkl929.
77. Luscombe, N.M., et al., Genomic analysis of regulatory network dynamics reveals large topological changes. *Nature*, 2004. **431**: 308–312.
78. Guldener, U., et al., CYGD: The comprehensive yeast genome database. *Nucl. Acids Res.*, 2005. **33**(suppl_1): D364–D368, doi: 10.1093/nar/gki053.
79. Andreoli, C., et al., MitoP2, an integrated database on mitochondrial proteins in yeast and man. *Nucl. Acids Res.*, 2004. **32**(1): D459–D462.
80. Fink, J.L., et al., LOCATE: A mouse protein subcellular localization database. *Nucl. Acids Res.*, 2006. **34**(suppl_1): D213–D217, doi: 10.1093/nar/gkj069.
81. Nakai, K. and P. Horton, PSORT: A program for detecting the sorting signals of proteins and predicting their subcellular localization. *Trends Biochem. Sci.*, 1999. **24**(1): 34–35.
82. Drawid, A. and M. Gerstein, A Bayesian system integrating expression data with sequence patterns for localizing proteins: Comprehensive application to the yeast genome. *J. Mol. Biol.*, 2000. **301**: 1059–1075.

83. Nair, R. and B. Rost, LOC3D: Annotate subcellular localization for protein structures. *Nucl. Acids Res.*, 2003. **31**(13): 3337–3340.
84. Olof Emanuelsson, et al., Predicting subcellular localization of proteins based on their N-terminal amino acid sequence. *J. Mol. Biol.*, 2000. **300**: 1005–1016.
85. Hua, S. and Z. Sun, Support vector machine approach for protein subcellular localization prediction. *Bioinformatics*, 2001. **17**(8): 721–728.
86. Mulder, N.J., et al., InterPro, progress and status in 2005. *Nucl. Acids Res.*, 2005. **33**(Database issue): D201–D205.
87. Mulder, N.J., et al., New developments in the InterPro database. *Nucl. Acids Res.*, 2007. **35**(suppl_1): D224–D228, doi: 10.1093/nar/gkl841.
88. Finn, R.D., et al., Pfam: Clans, web tools and services. *Nucl. Acids Res.*, 2006. **34**(Database issue): D247–D251.
89. Hulo, N., et al., The PROSITE database. *Nucl. Acids Res.*, 2006. **34**(Database issue): D227–D230.
90. Catherine B., et al., The ProDom database of protein domain families: More emphasis on 3D. *Nucl. Acids Res.*, 2005. **33**(Database Issue): D212–D215.
91. Henikoff, S., J.G. Henikoff, and S. Pietrokovski, Blocks+: A non-redundant database of protein alignment blocks derived from multiple compilations. *Bioinformatics*, 1999. **15**(6): 471–479.
92. Henikoff, J.G., et al., Increased coverage of protein families with the blocks database servers. *Nucl. Acids Res.*, 2000. **28**: 228–230.
93. Attwood, T.K., et al., PRINTS and its automatic supplement, prePRINTS. *Nucl. Acids Res.*, 2003. **31**: 400–402.
94. Haft, D.H., J.D. Selengut, and O. White, The TIGRFAMs database of protein families. *Nucl. Acids Res.*, 2003. **31**: 371–373.
95. Meinel, T., A. Krause, H. Luz, M. Vingron, and E. Staub, The SYSTERS protein family database in 2005. *Nucl. Acids Res.*, 2005. **33**(Database issue): D226–D229.
96. Murzin, A.G., et al., SCOP: A structural classification of proteins database for the investigation of sequences and structures. *J. Mol. Biol.*, 1995. **247**: 536–540.
97. Andreeva A., et al., SCOP database in 2004: Refinements integrate structure and sequence family data. *Nucl. Acid Res.*, 2004. **32**: D226–D229.
98. Letunic, I., et al., SMART 5: Domains in the context of genomes and networks. *Nucl. Acids Res.*, 2006. **34**(suppl_1): D257–D260, doi: 10.1093/nar/gkj079.
99. Gough, J., et al., Assignment of homology to genome sequences using a library of hidden Markov models that represent all proteins of known structure. *J. Mol. Biol.*, 2001. **313**(4): 903–919.
100. Gough, J. and C. Chothia, SUPERFAMILY: HMMs representing all proteins of known structure. SCOP sequence searches, alignments and genome assignments. *Nucl. Acids Res.*, 2002. **30**(1): 268–272, doi: 10.1093/nar/30.1.268.
101. Pearl, F., et al., The CATH domain structure database and related resources Gene3D and DHS provide comprehensive domain family information for genome analysis. *Nucl. Acids Res.*, 2005. **33**(suppl_1): D247–D251, doi: 10.1093/nar/gki024.
102. Yeats, C., et al., Gene3D: Modelling protein structure, function and evolution. *Nucl. Acids Res.*, 2006. **34**(suppl_1): D281–D284, doi: 10.1093/nar/gkj057.
103. Wu, C.H., et al., PIRSE: Family classification system at the protein information resource. *Nucl. Acids Res.*, 2004. **32**(suppl_1): D112–D114, doi: 10.1093/nar/gkh097.
104. Mi, H., et al., The PANTHER database of protein families, subfamilies, functions and pathways. *Nucl. Acids Res.*, 2005. **33**(suppl_1): D284–D288, doi: 10.1093/nar/gki078.
105. Mi, H., et al., PANTHER version 6: Protein sequence and function evolution data with expanded representation of biological pathways. *Nucl. Acids Res.*, 2007. **35**(suppl_1): D247–D252, doi: 10.1093/nar/gkl869.
106. Marchler-Bauer, A., et al., CDD: A conserved domain database for protein classification. *Nucl. Acids Res.*, 2005. **33**(suppl_1): D192–D196, doi: 10.1093/nar/gki069.

Chapter 7

Detecting Hierarchical Modularity in Biological Networks

Erzsébet Ravasz

Abstract

Spatially or chemically isolated modules that carry out discrete functions are considered fundamental building blocks of cellular organization. However, detecting them in highly integrated biological networks requires a thorough understanding of the organization of these networks. In this chapter I argue that many biological networks are organized into many small, highly connected topologic modules that combine in a hierarchical manner into larger, less cohesive units. On top of a scale-free degree distribution, these networks show a power law scaling of the clustering coefficient with the node degree, a property that can be used as a signature of hierarchical organization. As a case study, I identify the hierarchical modules within the *Escherichia coli* metabolic network, and show that the uncovered hierarchical modularity closely overlaps with known metabolic functions.

Key words: Networks, modularity, hierarchical organization, clustering coefficient, metabolism.

1. Introduction

The identification and characterization of the system-level features of biological organization is a key issue in post-genomic biology (1–3). The concept of modularity, the assumption that cellular functionality can be partitioned into a collection of well-defined units (1, 4–8), is a very popular paradigm of this field, attempting to connect structural elements of living systems to the functions they perform. Spatially and chemically isolated molecular machines or protein complexes (such as ribosomes and flagella) are prominent examples of such functional units, but more extended modules, such as those achieving their isolation through the initial binding of a signaling molecule (9), are also apparent. Simultaneously, it is recognized that the thousands of components of a

living cell are dynamically interconnected, so that the cell's functional properties are ultimately encoded into a complex intracellular network of molecular interactions (2–6, 8). In biological systems networks emerge in many guises, from food webs in ecology to various biochemical nets in molecular biology. During the past decade, genomics has produced an incredible quantity of molecular interaction data, contributing to maps of specific cellular networks. Extensive protein–protein interaction maps have been generated for a variety of organisms including viruses (10, 11), prokaryotes, like *Helicobacter pylori* (12), and eukaryotes, like *Saccharomyces cerevisiae* (13–19), *Caenorhabditis elegans* (20) and *Drosophila melanogaster* (21). Mapping out the genetic regulatory interactions is central to uncovering the cellular program encoded in the genome; several incomplete networks have been compiled from literature searches (8, 22–26) as well as system-level experiments (27). Beyond the current focus on uncovering the structure of genomes, proteomes, and interactomes of various organisms, some of the most extensive data sets are the metabolic maps (28, 29), catalyzing an increasing number of studies focusing on the architecture of metabolism (19, 30, 31).

Networks offer us a new way to categorize systems of very different origin under a single framework (32–41). This approach has uncovered unexpected similarities between the organization of various complex systems, indicating that the networks describing them are governed by generic organizational principles and mechanisms. Two properties of real networks have generated considerable attention. First, many networks are fundamentally modular: one can easily identify groups of nodes that are highly interconnected with each other, but have only a few or no links to nodes outside of the group to which they belong. Each module is expected to perform an identifiable task, separate from the function of other modules (1, 3, 4, 8). For example, in protein–protein interaction networks such modules represent protein complexes like the ribosome. This clearly identifiable modular organization is responsible for the high clustering coefficient (42) (the average probability that a node's two first neighbors are also connected) seen in many real networks. Empirical results indicate that the average clustering coefficient is significantly higher for many real networks than for a random network of similar size (32, 34, 42), and it is to a high degree independent of the number of nodes in the network (32). At the same time, many networks of scientific or technological interest, ranging from biological networks (19, 31, 43, 44) to the World Wide Web (45), have been found to be scale-free (46, 47); there is no well-defined “connectivity scale” that approximates the degree (number of connections) of most nodes in the system. Instead, the distribution of degrees follows an inverse power law with exponents between 2 and 3, indicating that these systems have very large

connectivity fluctuations. Most nodes have one or two links, but there are a few hubs with very large degrees. The scale-free property and strong clustering are not exclusive, but they coexist in a large number of real networks including metabolic webs (31), the protein interaction network (43, 44), the WWW (45), and some social networks (48–50).

In order to bring modularity, the high degree of clustering and the scale-free topology under a single roof, we assume that modules combine with one another in a hierarchical manner, generating what we call a *hierarchical network* (51–53). The presence of a hierarchy and the scale-free property impose strict restrictions on the number and degree of cohesiveness of the different groups present in a network. This can be captured in a quantitative manner using a scaling law that describes the dependence of the clustering coefficient on the node degree. We can use this scaling law to look for the presence or absence of hierarchical architecture in real networks.

Here we focus on the topological organization of cellular metabolism, a fully connected biochemical network in which hundreds of metabolic substrates are densely integrated through biochemical reactions. Within this network, modular organization (i.e., clear boundaries between subnetworks) is not immediately apparent. The degree distribution $P(k)$ of a metabolic network decays as a power law $P(k) < k^{-\gamma}$ with $\gamma = 2.2$ in all studied organisms (19, 31), suggesting that metabolic networks have a scale-free topology. This implies the existence of a few highly connected nodes (e.g., pyruvate or coenzyme A), which participate in a very large number of metabolic reactions. With a large number of links, these hubs seem to link all substrates into a single, integrated web in which the existence of fully separated modules is prohibited by definition. Nonetheless, a number of approaches for analyzing the functional capabilities of metabolic networks indicate the existence of separable functional elements (54, 55). Also, metabolic networks are known to possess high clustering coefficients (31) suggestive of a modular organization. We show that hierarchical modularity reconciles all the observed properties of metabolic networks within a single framework (51). Moreover, the hierarchical module structure can be easily uncovered and it corresponds to known functional classification of metabolic reactions.

Hierarchical topology gives a precise and quantitative meaning to network modularity. It indicates that we should not think of modularity as the coexistence of relatively independent groups of nodes. Instead, we have many small clusters that are densely interconnected. These combine to form larger but less cohesive groups, which combine again to form ever larger and less interconnected clusters. This self-similar nesting of different groups or modules into each other forces a strict fine structure on real networks.

2. Methods

2.1. Scaling of the Clustering Coefficient: A Signature of Hierarchy

A hierarchically organized network is made of numerous small, highly integrated modules, which are assembled into larger ones (see an illustration on **Fig. 7.1**). These in turn are less integrated but still clearly separated from each other, and they in turn combine into even larger, less cohesive modules and so on. This type of hierarchy can be characterized in a quantitative manner using the finding of Dorogovtsev et al. (56) that in certain deterministic scale-free networks the clustering coefficient of a node with k links follows the scaling law $C(k) < k^{-1}$. This scaling law quantifies the

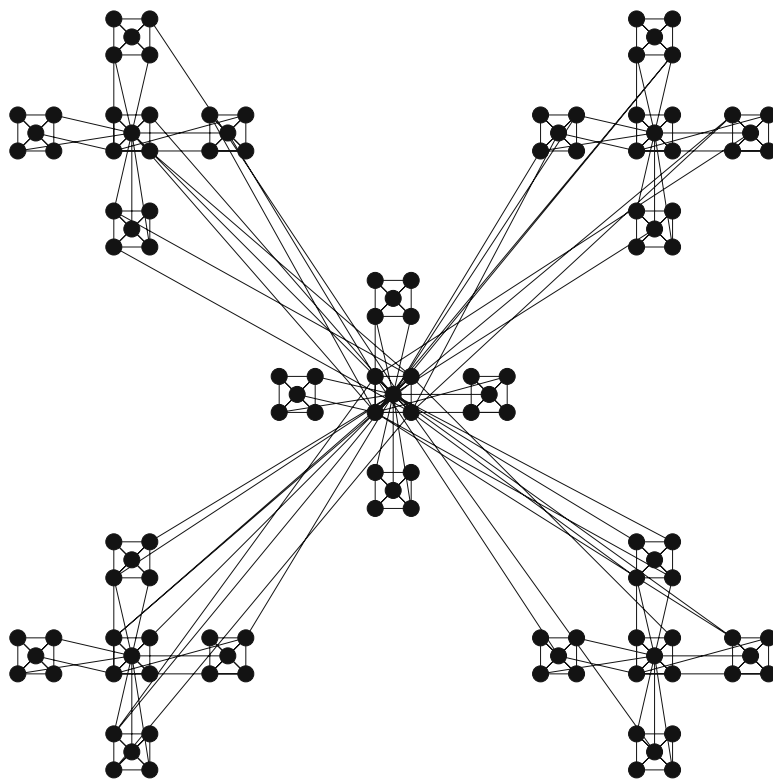


Fig. 7.1. Hierarchical model network ($n = 2$, $p = 3/5$). Its construction starts with a small core of five nodes, all connected to each other. In **step 1** ($n = 1$) four copies of the five-node module are made, then a fraction p of the newly copied nodes are picked at random and connected to a node from the central module (following preferential attachment (46, 47): the probability that a selected node connects to a node i of the central module is $k_i / \sum_j k_j$, where k_i is the degree of node i and the sum goes over all nodes of the central module). In the second step ($n = 2$), another four identical copies are created of the 25-node structure obtained thus far, but only a p^2 fraction of the newly added nodes are connected to the central module in step $n = 0$ (the original five nodes). Subsequently, in each iteration n the central module of size 5^n is replicated four times, and in each new module a p^n fraction is connected to the previous central module (i.e. the network at step $n-2$), requiring the addition of $(5p)^n$ new links (52).

coexistence of a hierarchy of nodes with different degrees of clustering. Nodes in the numerous small and cohesive modules have very large clustering coefficients. Nodes that hold together the larger but less cohesive modules have smaller clustering coefficients, indicating that the higher a node's degree the smaller its clustering coefficient becomes, asymptotically following the $1/k$ law. In contrast, Ref. (47), the Erdős-Rényi random network model (57, 58) or various small world models (42, 59), the clustering coefficient is independent of k .

The presence of such a hierarchical architecture reinterprets the role of the hubs in complex networks. Hubs, the highly connected nodes at the tail of the power law degree distribution, are known to play a key role in keeping complex networks together, playing a crucial role from the robustness of the network (60, 61) to the spread of viruses in scale-free networks (62). In a hierarchical structure the clustering coefficient of the hubs decreases linearly with their degree. This implies that while the small nodes are part of highly cohesive, densely interlinked clusters, the hubs are not, as their neighbors have a small chance of linking to each other. Therefore, the hubs play the important role of bridging many small communities of clusters into a single, integrated network. Most interesting, however, is the fact that the hierarchical nature of these networks is well captured by the $C(k)$ curve, offering us a relatively straightforward method to identify the presence of hierarchy in real networks (*see Note 1*). The scaling law does not have to have a universal exponent -1 ; the idea is that larger nodes have low clustering coefficients; their role is to integrate the smaller tight clusters on all scales (52).

Figure 7.2 shows examples of three types of biological networks displaying hierarchical scaling: metabolic (28, 43, 51), protein-protein interaction (16, 63, 64), and genetic regulatory networks (8, 22–27).

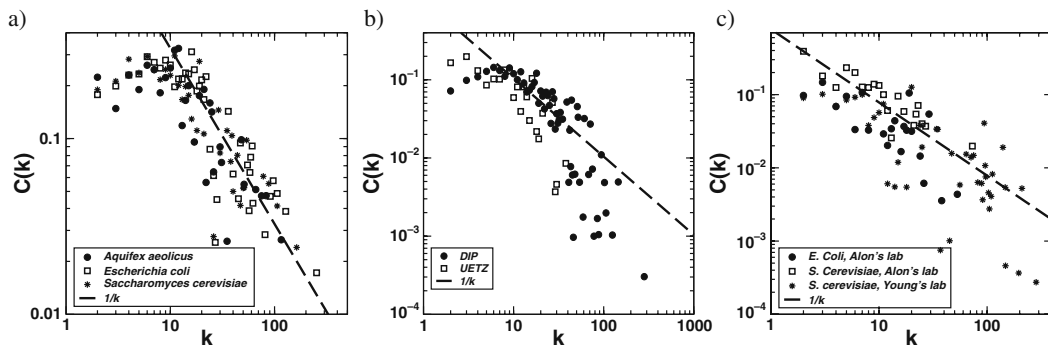


Fig. 7.2. Dependence of the clustering coefficient on the node's degree in three classes of biological networks: **(a)** metabolic networks of *Aquifex aeolicus* (archaea), *Escherichia coli* (bacterium), and *Saccharomyces cerevisiae* (eukaryote) (19, 51); **(b)** protein-protein interaction networks from the DIP database (63, 64) and from an extensive yeast two-hybrid experiment (16); **(c)** genetic regulatory interaction networks from the RegulonDB database (8, 22–26) (network data downloaded from Uri Alon's lab, <http://www.weizmann.ac.il/mcb/UriAlon/>) and a system-level experiment in yeast (27).

2.2. Method for Finding Network Modules

A key issue from a biological perspective is to identify the hierarchically embedded modules of biological networks, and understand how the uncovered structure relates to the true functional organization of the system. To this end we use a standard clustering algorithm that uses a similarity measure between nodes to group them onto a hierarchical tree (for a different clustering method *see Note 2*). We define the node-to-node distance through a measure we call topological overlap.

2.2.1. The Topological Overlap Matrix

In order to quantify whether two nodes are closely linked into the same local cluster, we introduce the topological overlap matrix, $O_{\Gamma}(i, j)$. Topological overlap of 1 between substrates i and j implies that they are connected to the same substrates, whereas a 0 value indicates that i and j do not share a link, nor links to common substrates among the metabolites they react with. Defining the adjacency matrix, $A_{i,j}$, of the network as

$$A_{i,j} = \begin{cases} 1 & \text{if } i \text{ and } j \text{ are connected} \\ 0 & \text{if } i \text{ and } j \text{ are not connected} \end{cases}$$

the elements of the overlap matrix are given by

$$O_{\Gamma}(i, j) = \frac{\sum_{k=1}^N A_{i,k} \cdot A_{j,k} + A_{i,j}}{\min(k_i, k_j) + 1 - A_{i,j}}.$$

As the topological overlap matrix is expected to encode the comprehensive functional relatedness of the biological network, we expect that functional modules encoded in the network topology can be automatically uncovered by a standard clustering algorithm.

2.2.2. Hierarchical Clustering Algorithm

We choose the unweighted average linkage algorithm (or Unweighted Pair Group Method with Arithmetic Mean) known as UPGMA (65, 66) for our hierarchical clustering method. This algorithm first finds the largest overlap present in the matrix, joins the corresponding nodes u and v to a branching point on the tree, and substitutes them with a “new” cluster $\{u, v\}$. This new unit replaces the original u and v in the overlap matrix. It has an overlap with an arbitrary substrate (cluster) w given by

$$O_{\Gamma}(\{u, v\}, w) = \frac{n_u \cdot O_{\Gamma}(u, w) + n_v \cdot O_{\Gamma}(v, w)}{n_u + n_v},$$

where n_u is the number of components in cluster u . This definition ensures that all original overlap values are represented with the same weight in the overlap value of the joint cluster, hence the method’s name “unweighted average linkage clustering.” Repeating this rule eventually shrinks the overlap matrix to a single unit, corresponding to the root of hierarchical tree. Thus, we obtain a tree with all the original substrates as its end-leaves, grouped naturally on branches

reflecting their hierarchical overlap. When overlap values between clusters are redundant (i.e., there are at least two groups of clusters with the same overlap value), the program automatically joins the pair found first. The ordering of two branches under a junction is irrelevant, thus arbitrary. The distance between (height of) two junction levels is defined to be one (**Note 3**).

We can follow how the clustering algorithm works on a small hypothetical network shown in **Fig. 7.3a**. The method placed those nodes that have a high topological overlap close to each other (**Fig. 7.3b**), correctly identifying the three distinct modules built into the network. It also identified the relationship between the three modules, as EFG and HIJK are closer to each other in a topological sense than the ABC module.

2.3. A Case Study: The *Escherichia coli* Metabolic Network

To uncover potential relationships between the topological modularity and the functional classification of different metabolites we concentrate on the metabolic network of *Escherichia coli*, whose metabolic reactions have been exhaustively mapped and studied (29). Here we used the network data compiled from the WIT database (19, 28, 51). In order to make our final module map smaller and easier to study, we take advantage of a few peculiarities of metabolism and generate a reduced network before we start the clustering procedure.

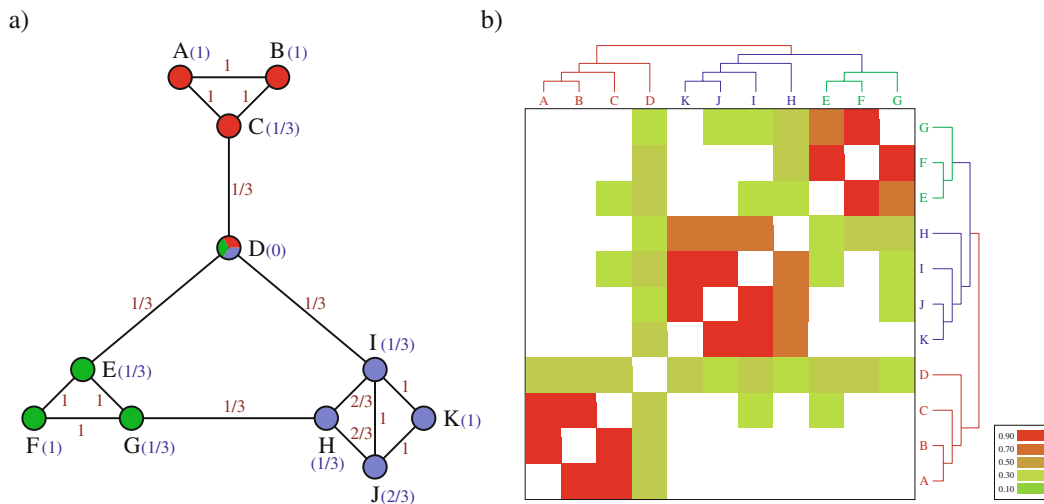


Fig. 7.3. Uncovering the underlying modularity of a complex network. **(a)** Topological overlap illustrated on a small hypothetical network. On each link, we indicate the topological overlap for the connected nodes; and in parentheses next to each node, we indicate the node's clustering coefficient. **(b)** The topological overlap matrix corresponding to the small network shown in **(a)**. The rows and columns of the matrix were reordered by the application of an average linkage clustering method to its elements, allowing us to identify and place close to each other those nodes that have high topological overlap. The color code denotes the degree of topological overlap between the nodes. The associated tree reflects the three distinct modules built into the model, as well as the fact that the EFG and HIJK modules are closer to each other in the topological sense than to the ABC module. (see Color Plate)

2.3.1. Generating the Reduced *E. Coli* Metabolic Network

Metabolism relies heavily on the usage of a few substrate pairs, which undergo very generic chemical changes in a large number of reactions of all types. A representative example is the ATP–ADP pair, the cell’s energy fuel molecules. As a phosphate group is broken off ATP (adenosine triphosphate), the energy released from the chemical bond fuels the chemical change of the substrate(s) which ATP reacts with. This mechanism is so generic that ATP and ADP are the greatest hubs of our network: they are linked to a significant fraction of all substrates. A link from ATP, ADP, water, etc. to a metabolite *A* often carries little biologically relevant information about the function of *A*. There are many different reactions where other pairs of metabolites help some reactions to take place: exchange of a proton or a methyl group, for example. In order to focus on biologically relevant substrate transformations, we have performed a biochemical reduction of the metabolic network. Our guiding principle was to maintain the main line of substrate transformation on each pathway (**Fig. 7.4**). It is important to note that the reduction process is completely local: it takes place at the level of each reaction, and does not result in the removal of metabolites, but only in the removal of links from the graph representation.

To further reduce the complexity of the metabolic graph, we continue with a two-step topological reduction. Many pathways uncovered by the first reduction are connected to the rest of the metabolic network by a single substrate, or represent a long chain of consecutive substrates that appear as an arc between two

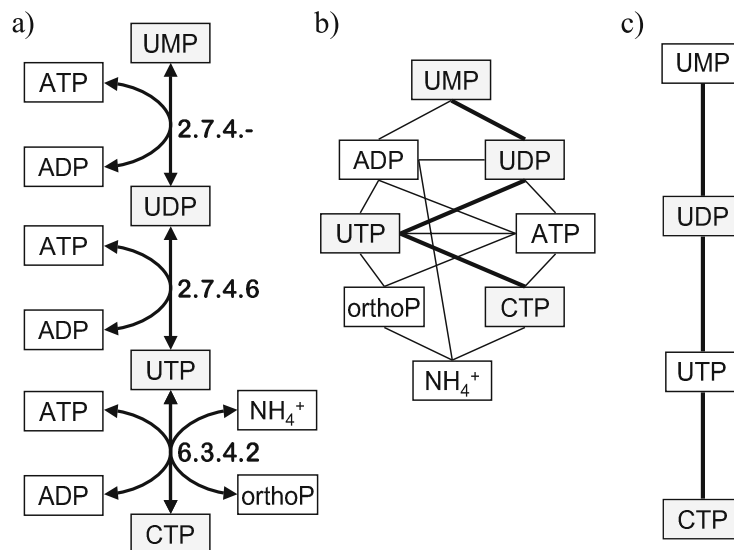


Fig. 7.4. Biochemical reduction of the pathways of the metabolic network. The *middle panel* shows the full graph theoretic representation of the path way shown in the *left panel* (19). The *right panel* displays the pathway after biochemical reduction.

substrates, and have no other side connections. Since the topological location of the strings of substrates depend only on one or two multiply connected terminal substrates, we can temporarily remove the long non-branching pathways or replace them with a direct link, without altering the topology of core metabolism. We define hairs as all sets of nodes that can be separated from the network by cutting one link. An arc is an array of nodes connected by only two links to the rest of the metabolism, leading from one well-connected substrate to another. To generate the reduced metabolic network we have removed all hairs from the network and replaced all arcs with a single link, directly connecting the substrates at the two ends of an arc (**Note 4**). While the substrates removed during the topological reduction process are biologically important components of the network, their removal does not change the way in which subunits that they were removed from connect to other parts of the metabolism. In this sense they are topologically irrelevant (**Note 5**).

2.3.2. Finding the Hierarchy of Modules

After reducing the metabolic network to a representative core, we proceed to break it up into clusters based on its wiring diagram. The ordering of the overlap matrix according to a substrate's horizontal location on the hierarchical tree leads to **Fig. 7.5**. This figure provides us a global topological representation of the metabolism. Groups of metabolites forming tightly interconnected clusters are visually apparent along the diagonal line of the matrix; and upon closer inspection, the hierarchy of nested topological modules of increasing sizes and decreasing interconnectedness can also be seen. To visualize the relationship between the topological modules and the known functional properties of the metabolites, we color-coded the branches of the derived hierarchical tree according to the predominant biochemical class of the substrates it produces, using a standard, small molecule biochemistry-based classification of metabolism (28). The biochemical classes we used to group the metabolites represent carbohydrate metabolism (blue), nucleotide and nucleic acid metabolism (red), protein, peptide, and amino acid metabolism (green), lipid metabolism (cyan), aromatic compound metabolism (dark pink), monocarbon compound metabolism (yellow), and coenzyme metabolism (light orange) (**Note 6**). The color-coding of the hierarchical tree according to biochemical classification of the metabolites proved a very good agreement between the uncovered modular hierarchy and the standard classes of the metabolism. As shown in **Fig. 7.5**, we find that most substrates of a given small molecule class are distributed on the same branch of the tree. Therefore, there are strong correlations between shared biochemical classification of metabolites and the global topological organization of *E. coli* metabolism (**Fig. 7.5**, bottom).

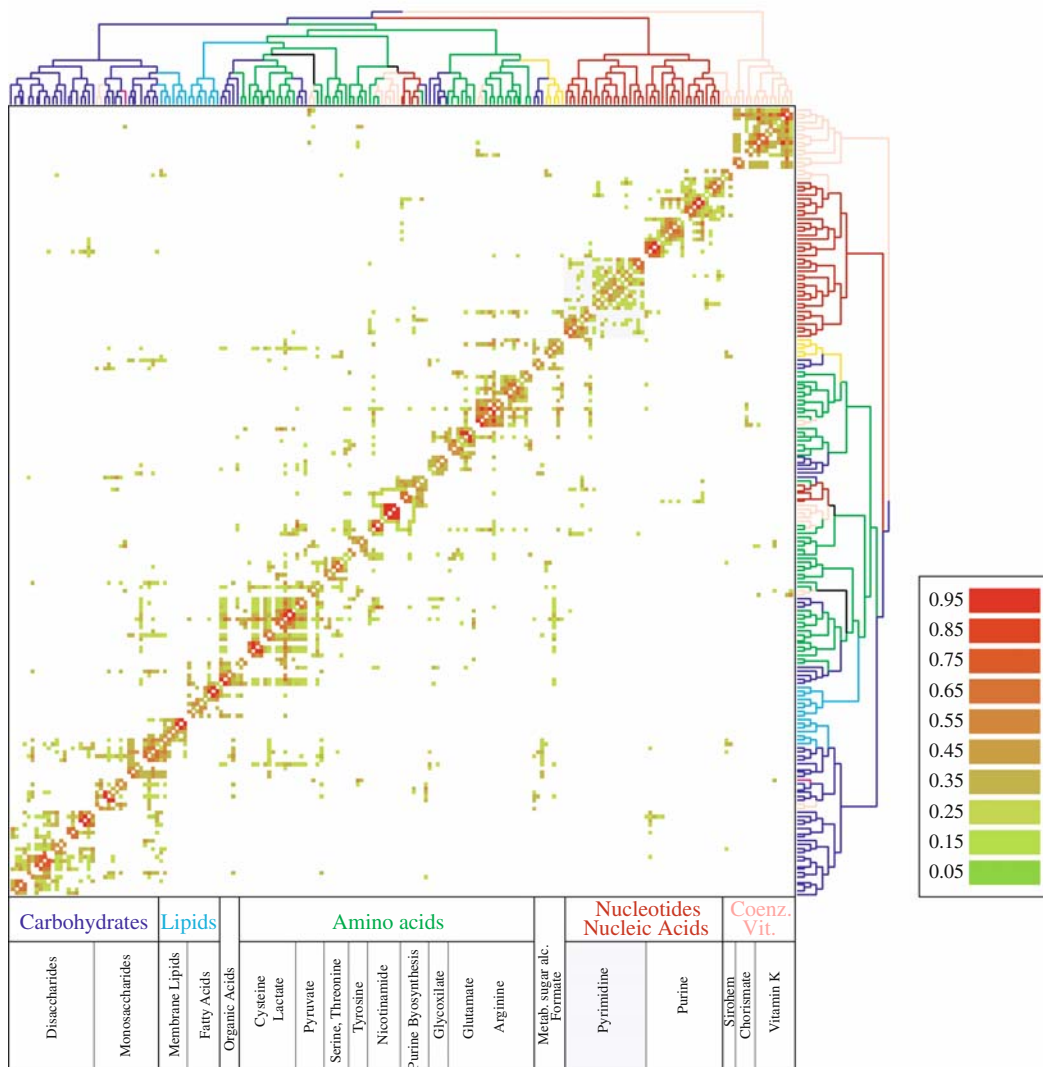


Fig. 7.5. Topological modules in the *Escherichia coli* metabolism: the topologic overlap matrix, together with the corresponding hierarchical tree (*top and right*) that quantifies the relation between the different modules. The branches of the tree are *color-coded* to reflect the predominant biochemical classification of their substrates. The *color code* of the matrix denotes the degree of topological overlap shown in the matrix. The large-scale functional map of the metabolism, as suggested by the hierarchical tree, is also shown (*bottom*). (see Color Plate)

2.3.3. Biochemical Pathways in the Pyrimidine Module

To correlate the modules obtained from graph theory-based analysis to actual biochemical pathways, we concentrate on the pathways of the pyrimidine metabolism. The clustering method divides these pathways into four modules, as shown in **Fig. 7.6**. All highly connected metabolites (red boxes) correspond to their respective biochemical reactions within pyrimidine metabolism, together with those substrates that were removed during the original network reduction procedure, and then re-added (**Fig. 7.6**, green boxes). However, it is also apparent that the putative module

boundaries do not always overlap with intuitive “biochemistry-based” boundaries. For instance, while the synthesis of UMP from L-glutamine is expected to fall within a single module based on a linear set of biochemical reactions, the synthesis of UDP from UMP leaps putative module boundaries.

3. Conclusions

Using the above-presented method revealed that the system-level structure of cellular metabolism is best approximated by a hierarchical network organization with seamlessly embedded modularity. In contrast to the intuitive picture of modularity, which assumes the existence of a set of modules with a non-uniform size potentially separated from other modules, we find that the metabolic network has an inherent self-similar property: there are many highly integrated small modules that group into a few larger modules, which in turn can be integrated into even larger modules. This is supported by visual inspection of the derived hierarchical tree (**Fig. 7.5**), which offers a natural breakdown of metabolism into several large modules, which are further partitioned into smaller, but more integrated, sub-modules. We expect the method to be very useful for automatically uncovering functionally relevant modules in many types of biological networks.

4. Notes



1. While the presence of $C(k)$ scaling law is a good indication of hierarchical network architecture, there are known exceptions. There is a model proposed by Klemm and Eguíluz (67), which obeys the scaling law, it is nonetheless not composed of hierarchically embedded modules. Instead, the topology of the networks generated by the model is similar to a chain of locally connected dense clusters (68). In the Klemm-Eguíluz model a new node joins the network in each

Fig. 7.6. (continued) pathways branching from a metabolite with multiple connections. *Blue-* and *black-outlined boxes* show the connections of pyrimidine metabolites to other parts of the metabolic network. *Black-outlined boxes* denote the core substrates belonging to other branches of the metabolic tree, and *blue-outlined boxes* denote non-branching pathways (if present) leading to those substrates. The *shaded blue boxes* along the links display the enzymes catalyzing the corresponding reactions, and the *arrows* show the direction of the reactions according to the WIT metabolic maps (28) (see Color Plate).

time step, connecting to all *active* nodes in the system. At the same time an active node is deactivated with probability $p \sim 1/k$. Thus, by constantly deactivating the less connected nodes, there is always a central core to which the incoming nodes tend to link. Once deactivated, a node (and in general its neighborhood) does not receive more links, and the cluster it sits in is not embedded in larger and looser structures. Rather, a series of power-law size-distributed clusters follow each other along a chain. This model draws attention to a shortcoming of the $C(k)$ scaling as an indicator of hierarchy: a collection of internally homogeneous modules of different sizes *neighboring* each other (with just a few links to join them into a network) can also give rise to $C(k) < k^{-1}$. Thus, even if the scaling law holds for a particular network, one should check if the found modules do indeed form a hierarchy or they are a loosely linked collection of simple modules.

2. Other approaches to module detection in metabolic networks are based on the idea that edges along a large number of shortest paths are likely to link different modules of the network (69, 70). Edges on the largest number of paths were iteratively removed, slowly breaking the network into its functional modules.
3. The height of a junction could, in principle, contain information about the module represented by the branch under it. For example, the average overlap between metabolites located at the leaves of the branch could determine the height of the junction. However, this additional information does not change the structure of the tree and the way the modules are organized.
4. Note that we do not repeat the reduction process on the once reduced network. Thus, after the reduced network is ready, it can have arcs and hairs in it. These appear, for example, when two linked nodes both have hair on them, and they both have three links. After the reduction they are left with two links and thus are parts of a newly created arc.
5. Removing the “hairs” from a network does not alter the way its core nodes connect to each other and form modules. However, the shortening of arcs only makes sense if the network carries some type of conserved flow. In case of the metabolic network two substrates are usually connected via a long arc (pathway) not due to functional distance but due to chemical constraints. Enzymes catalyzing the steps along this pathway typically make small alterations to a molecule. Thus some changes take many steps that are part of the same process of converting A to B . In other types of networks a link may signify a very different relationship; for example, if

two proteins in the protein–protein interaction network are connected via a long arc, there is no reason to believe that they are closely related by function. In this network a link means physical binding; the lack of a direct link indicates that the two proteins probably do not work together. As a consequence, the protein interaction network should be analyzed without the shortening of its arcs.

6. The functional groups mentioned above are defined by series of reactions; thus many metabolites belong to more than one functional category. The color (category) we choose to represent the metabolite on the tree was the one that matched the color (category) of its neighbors on the tree, indicating that it has stronger connections to this group than any other.

References

1. Hartwell, L. H., Hopfield, J. J., Leibler, S., and Murray, A. W., From molecular to modular cell biology. *Nature*, 1999, 402:C47–C52.
2. Kitano, H., Systems biology: A brief overview. *Science*, 2002, 295:1662–1664.
3. Wolf, Y., Karev, G., and Koonin, E., Scale-free networks in biology: New insights into the fundamentals of evolution? *Bioessays*, 2002, 24:105–109.
4. Lauffenburger, D., Cell signaling pathways as control modules: Complexity for simplicity. *Proc. Natl. Acad. Sci. USA*, 2000, 97:5031–5033.
5. Rao, C. V., and Arkin, A. P., Control motifs for intracellular regulatory networks. *Annu. Rev. Biomed. Eng.*, 2000, 3:391–419.
6. Holter, N. S., Maritan, A., Cieplak, M., Fedoroff, N. V., and Banavar, J. R., Dynamic modeling of gene expression data. *Proc. Natl. Acad. Sci. USA*, 2001, 98:1693–1698.
7. Hasty, J., McMillen, D., Isaacs, F., and Collins, J. J., Computational studies of gene regulatory networks: *In numero* molecular biology. *Nat. Rev. Genet.*, 2001, 2:268–279.
8. Shen-Orr, S., Milo, R., Mangan, S., and Alon, U., Network motifs in the transcriptional regulation network of *E. coli*. *Nat. Genet.*, 2002, 31:64–68.
9. Alon, U., Surette, M. G., Barkai, N., and Leibler, S., Robustness in bacterial chemotaxis. *Nature*, 1999, 397:168–171.
10. Flajolet, M., Rotondo, G., Daviet, L., Bergametti, F., Inchauspe, G., Tiollais, P., Transy, C., and Legrain, P., A genomic approach to the *Hepatitis C* virus. *Gene*, 2000, 242:369–379.
11. McGraith, S., Holtzman, T., Moss, B., and Fields, S., Genome-wide analysis of vaccinia virus protein-protein interactions. *Proc. Natl. Acad. Sci. USA*, 2000, 97:4879–4884.
12. Rain, J. C., Selig, L., De Reuse, H., Battaglia, V., Reverdy, C., Simon, S., Lenzen, G., Petel, F., Wojcik, J., and Schachter, V., The protein-protein interaction map of *Helicobacter pylori*. *Nature*, 2001, 409:211–215.
13. Ito, T., Chiba, T., Ozawa, R., Yoshida, M., Hattori, M., and Sakaki, Y., A comprehensive two-hybrid analysis to explore the yeast protein interactome. *Proc. Natl. Acad. Sci. USA*, 2001, 98:4569–4574.
14. Ito, T., Tashiro, K., Muta, S., Ozawa, R., Chiba, T., Nishizawa, M., Yamamoto, K., Kuhara, S., and Sakaki, Y., Toward a protein-protein interaction map of the budding yeast: A comprehensive system to examine two-hybrid interactions in all possible combinations between the yeast proteins. *Proc. Natl. Acad. Sci. USA*, 2000, 97:1143–1147.
15. Schwikowski, B., Uetz, P., and Fields, S., A network of protein-protein interactions in yeast. *Nat. Biotechnol.*, 2000, 18:257–1261.
16. Uetz, P., Giot, L., Cagney, G., Mansfield, T., Judson, R., Knight, J., Lockshorn, D., Narayan, V., Srinivasan, M., and Pochart, P., A comprehensive analysis of protein-protein interactions of *Saccharomyces cerevisiae*. *Nature*, 2000, 403:623–627.
17. Gavin, A., Bösch, M., Krause, R., Grandi, P., Marzioch, M., Bauer, A., Schultz, J., Rick, J., and Michon, A.-M., Functional

- organization of the yeast proteome by systematic analysis of protein complexes. *Nature*, 2002, 415:141–147.
18. Ho, Y., Gruhler, A., Heilbut, A., Bader, G., Moore, L., Adams, S.-L., Millar, A., Taylor, P., Bennett, K., and Boutillier, K., Systematic identification of protein complexes in *Saccharomyces cerevisiae* by mass spectrometry. *Nature*, 2002, 415:180–183.
 19. Jeong, H., Tombor, B., Albert, R., Oltvai, Z. N., and Barabási, A.-L., The large-scale organization of metabolic networks. *Nature*, 2000, 407:651–654.
 20. Walhout, A., Sordella, R., Lu, X., Hartley, J., Temple, G., Brasch, M., Thierry-Mieg, N., and Vidal, M., Protein interaction mapping in *C. elegans* using proteins involved in vulval development. *Science*, 2000, 287:116–122.
 21. Giot, L., Bader, J. S., Brouwer, C., Chaudhuri, A., Kuang, B., Li, Y., Hao, Y. L., Ooi, C. E., Godwin, B., Vitols, E. et al., A protein interaction map of *Drosophila melanogaster*. *Science*, 2003, 302:1727–1735.
 22. Thieffry, D., Huerta, A. M., Perez-Rueda, E., and Collado-Vides, J., From specific gene regulation to genomic networks: A global analysis of transcriptional regulation in *Escherichia coli*. *Bioessays*, 1998, 20:433–440.
 23. Salgado, H., Santos-Zavaleta, A., Gama-Castro, S., Millán-Zárate, D., Díaz-Peredo, E., Sánchez-Solano, F., Pérez-Rueda, E., Bonavides-Martínez, C., and Collado-Vides, J., RegulonDB (version 3.2): Transcriptional regulation and operon organization in *Escherichia coli* K-12. *Nuc. Acids Res.*, 2001, 29:72–74.
 24. Costanzo, M. C., Crawford, M. E., Hirschman, J. E., Kranz, J. E., Olsen, P., Robertson, L. S., Skrzypek, M. S., Braun, B. R., Hopkins, K. L., Kondu, P. et al., YPDTM, PombePDTM and WormPDTM: model organism volumes of the BioKnowledgeTM Library, an integrated resource for protein information. *Nuc. Acids Res.*, 2001, 29:75–79.
 25. Mangan, S., and Alon, U., The structure and function of the feed-forward loop network motif. *Proc. Natl. Acad. Sci. USA*, 2003, 100:11980–11985.
 26. Milo, R., Shen-Orr, S., Itzkovitz, S., Kashtan, N., Chklovskii, D., and Alon, U., Network motifs: Simple building blocks of complex networks. *Science*, 2002, 298:824–827.
 27. Lee, T. I., Rinaldi, N. J., Robert, F., Odom, D. T., Bar-Joseph, Z., Gerber, G. K., Hannett, N. M., Harbison, C. T., Thompson, C. M., Simon, I. et al., Transcriptional regulatory networks in *Saccharomyces cerevisiae*. *Science*, 2002, 298:799–804.
 28. Overbeek, R., Larsen, N., Pusch, G., D'Souza, M., Selkov, E., Jr, Kyrpides, N., Fonstein, M., Maltsev, N., and Selkov, E., WIT: Integrated system for high-throughput genome sequence analysis and metabolic reconstruction. *Science*, 2000, 28:123–125.
 29. Karp, P. D., Riley, M., Saier, M., Paulsen, I., Paley, S., and Pellegrini-Toole, A., The EcoCyc and MetaCyc databases. *Nucl. Acids Res.*, 2000, 28:56–59.
 30. Fell, D. A., and Wagner, A., The small world of metabolism. *Nat. Biotechnol.*, 2000, 189:1121–1122.
 31. Wagner, A., and Fell, D. A., The small world inside large metabolic networks. *Proc. Roy. Soc. London Series B*, 2001, 268:1803–1810.
 32. Albert, R., and Barabási, A.-L., Statistical mechanics of complex networks. *Rev. Mod. Phys.*, 2002, 74:67–97.
 33. Newman, M. E. J., The structure and function of complex networks. *SIAM Rev.*, 2003, 45:167–256.
 34. Dorogovtsev, S. N., and Mendes, J. F. F., Evolution of networks. *Adv. Phys.*, 2002, 51:1079–1187.
 35. Boccaletti, S., Latora, V., Moreno, Y., Chavez, M., and Hwang, D.-U., Complex networks: Structure and dynamics. *Phys. Rep.*, 2006, 424:175–308.
 36. Barabási, A.-L., *Linked: The New Science of Networks*, 2002, Perseus Publishing, Cambridge, MA.
 37. Newman, M. E. J., Barabási, A.-L., and Watts, D. J. (eds.), *The Structure and Dynamics of Complex Networks*, 2003, Princeton University Press, Princeton.
 38. Pastor-Satorras, R., Rubi, M., and Diaz-Guilera, A. (eds.), *Statistical Mechanics of Complex Networks, Lecture Notes in Physics*, 2003, 625, Springer Verlag, Berlin, Germany.
 39. Mendes, J. F. F., Dorogovtsev, S. N., Povolotsky, A., Abreu, F. V., and Oliveira, J. G. (eds.), *Science of Complex Networks. From Biology to the Internet and WWW, AIP Conference Proceedings*, 2004, 776, American Institute of Physics, New York.
 40. Ben-Naim, E., Frauenfelder, H., and Toroczkai, Z. (eds.), *Complex Networks, Lecture Notes in Physics*, 2004, 650, Springer Verlag, Berlin/Heidelberg, Germany.

41. Bornholdt, S., and Schuster, H. G. (eds.), *Handbook of Graphs and Networks: From the Genome to the Internet*, 2002, Wiley-VCH, Berlin.
42. Watts, D. J., and Strogatz, S. H., Collective dynamics of small-world networks. *Nature*, 1998, 393:440–442.
43. Jeong, H., Mason, S., Barabási, A.-L., and Oltvai, Z. N., Lethality and centrality in protein networks. *Nature*, 2001, 411:41–42.
44. Wagner, A., The yeast protein interaction network evolves rapidly and contains few redundant duplicate genes. *Mol. Biol. Evol.*, 2001, 18:1283–1292.
45. Albert, R., Jeong, H., and Barabási, A.-L., Diameter of the world-wide web. *Nature*, 1999, 401:130–131.
46. Barabási, A.-L., Albert, R., and Jeong, H., Mean-field theory for scale-free random networks. *Physica A*, 1999, 272:173–187.
47. Barabási, A.-L., and Albert, R., Emergence of scaling in random networks. *Science*, 1999, 286:509–512.
48. Newman, M. E. J., The structure of scientific collaboration networks. *Proc. Natl. Acad. Sci. USA*, 2001, 98:404–409.
49. Barabási, A.-L., Jeong, H., Nédá, Z., Ravasz, E., Schubert, A., and Vicsek, T., Evolution of the social network of scientific collaborations. *Physica A*, 2002, 311:590–614.
50. Newman, M. E. J., Scientific collaboration networks. I. Network construction and fundamental results. *Phys. Rev. E*, 2001, 64:016131.
51. Ravasz, E., Somera, A. L., Mongru, D. A., Oltvai, Z. N., and Barabási, A.-L., Hierarchical organization of modularity in metabolic networks. *Science*, 2002, 297:1551–1555.
52. Ravasz, E., and Barabási, A.-L., Hierarchical organization in complex networks. *Phys. Rev. E*, 2002, 67:026122.
53. Barabási, A.-L., Ravasz, E., and Vicsek, T., Deterministic scale-free networks. *Physica A*, 2001, 299:559–564.
54. Schilling, C. H., Letscher, D., and Palsson, B. O., Theory for the systemic definition of metabolic pathways and their use in interpreting metabolic function from a pathway-oriented perspective. *J. Theor. Biol.*, 2000, 203:229–248.
55. Schuster, H. G., *Complex Adaptive Systems*, 2002, Scator Verlag, Saarbruskey, Germany.
56. Dorogovtsev, S. N., Goltsev, A. V., and Mendes, J. F. F., Pseudofractal Scale-free Web. *Phys. Rev. E*, 2002, 65:066122.
57. Erdős, P., and Rényi, A., On random graphs I. *Publ. Math. (Debrecen)*, 1959, 6:290–297.
58. Bollobás, B., *Random Graphs*, 1985, Academic Press, London.
59. Newman, M. E. J., Models of the small world. *J. Stat. Phys.*, 2000, 101:819–841.
60. Albert, R., Jeong, H., and Barabási, A.-L., Error and attack tolerance of complex networks. *Nature*, 2000, 406:378–382.
61. Cohen, R., Erez, K., Ben-Avraham, D., and Havlin, S., Breakdown of the internet under intentional attack. *Phys. Rev. Lett.*, 2001, 86:3682–3685.
62. Pastor-Satorras, R., and Vespignani, A., Epidemic spreading in scale-free networks. *Phys. Rev. Lett.*, 2001, 86:3200–3203.
63. Xenarios, I., Rice, D. W., Salwinski, L., Baron, M. K., Marcotte, E. M., and Eisenberg, D. M. S., DIP: The Database of Interacting Proteins. *Nucl. Acids. Res.*, 2000, 28:289–291.
64. Xenarios, I., Fernandez, E., Salwinski, L., Duan, X., Thompson, M., Marcotte, E., and Eisenberg, D., DIP: The Database of Interacting Proteins: 2001 update. *Nucl. Acids Res.*, 2001, 29:239–241.
65. Eisen, M. B., Spellman, P. T., Brown, P. O., and Botstein, D., Cluster analysis and display of genome-wide expression patterns. *Proc. Natl. Acad. Sci. USA*, 1998, 95:14863–14868.
66. Sokal, J., *Numerical Taxonomy*, 1973, Freeman, San Francisco.
67. Klemm, K., and Eguíluz, V. M., Growing scale-free networks with small-world behavior. *Phys. Rev. E*, 2002, 65:057102.
68. Vázquez, A., Moreno, Z., Boguñá, M., Pastor-Satorras, R., and Vespignani, A., Topology and correlations in structured scale-free networks. *Phys. Rev. E*, 2003, 67:046111.
69. Girvan, M., and Newman, M. E. J., Community structure in social and biological networks. *Proc. Natl. Acad. Sci. USA*, 2002, 99:7821–7826.
70. Holme, P., Huss, M., and Jeong, H., Sub-network hierarchies in biochemical pathways. *Bioinformatics*, 2003, 19:532–538.

Chapter 8

Methods to Reconstruct and Compare Transcriptional Regulatory Networks

M. Madan Babu, Benjamin Lang, and L. Aravind

Abstract

The availability of completely sequenced genomes and the wealth of literature on gene regulation have enabled researchers to model the transcriptional regulation system of some organisms in the form of a network. In order to reconstruct such networks in non-model organisms, three principal approaches have been taken. First, one can transfer the interactions between homologous components from a model organism to the organism of interest. Second, microarray experiments can be used to detect patterns in gene expression that stem from regulatory interactions. Finally, knowledge of experimentally characterized transcription factor binding sites can be used to analyze the promoter sequences in a genome in order to identify potential binding sites. In this chapter, we will focus in detail on the first approach and describe methods to reconstruct and analyze the transcriptional regulatory networks of uncharacterized organisms by using a known regulatory network as a template.

Key words: Transcriptional regulatory network, network reconstruction, template-based method, network motif, lifestyle, statistical significance.

1. Introduction

Advances in genome sequencing techniques are yielding complete sequences of genomes of several organisms. Although methods to identify protein-coding genes within these genomes are highly advanced today, what this information does not tell us, however, is how the products of these genes interact and how they are regulated. In recent years, a variety of high-throughput techniques have been developed and employed to generate vast amounts of data that could be used to bridge this gap. For instance, high-throughput microarray experiments are providing expression data

for a number of genes under a variety of conditions (1–3); large-scale experiments using chromatin immuno-precipitation combined with microarray hybridization (ChIP-chip) are giving us specific evidence of the regulatory proteins binding to stretches of DNA (4–7), and additionally, large amounts of data on protein–DNA interactions from individually conducted experiments over the years are collected in databases, resulting in a wealth of literature on gene regulation (8, 9). For some model organisms, such as *Escherichia coli* and yeast, these data have been integrated to produce comprehensive models of their transcriptional regulatory interactions in the form of networks (10, 11). The challenge that we currently face is to develop computational techniques that would allow us to make the most of this information in order to understand regulation in organisms that are poorly characterized.

There are three fundamental approaches that can be taken to infer the structure of the organism’s regulatory interaction network from these data, at varying levels of resolution. These are (i) **Template-based methods**: This approach exploits the principle that orthologous transcription factors generally regulate the expression of orthologous target genes. Thus, in this network reconstruction method, one starts with a known regulatory network and transfers the information about interactions to genes that have been determined to be orthologous in a target genome of interest (12–14). This principle works best in prokaryotes, where orthologous one-component or two-component transcription factors effectively respond to the same signal and tend to regulate similar sets of target regulons. (ii) **Reverse engineering using gene expression data**: In this approach, one scans for patterns in gene expression data from time-series experiments and from experiments conducted across several different conditions. If a gene is upregulated following an increased production of a transcription factor, or downregulated following a knockout of a transcription factor, a regulatory interaction between the two is inferred. In the case of expression analysis over different experimental conditions, one infers sets of genes with a similar expression profile across many conditions to be co-regulated by the same set of transcription factors (15–19). Such inferences become more accurate as the number of measurements over a certain period of time (the time-scale resolution of the data) increases, since this allows direct regulatory interactions to be distinguished from indirect (multi-step) regulation. (iii) **Infering networks by predicting cis-regulatory elements**: The third approach makes use of the information about experimentally well-characterized transcription factor binding sites to make inferences about regulatory interactions. In this method, promoter regions in the genome of interest are scanned using known binding site profiles of characterized transcription factors.

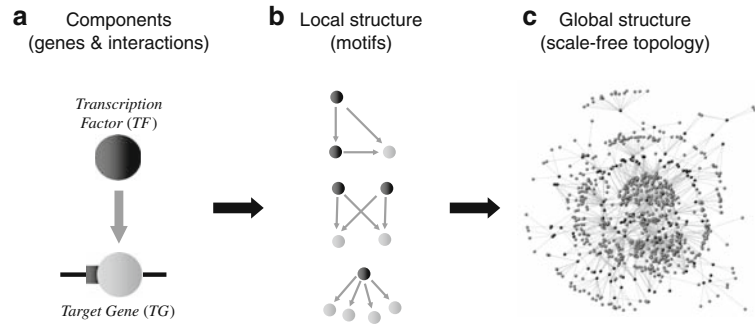
The set of genes that are predicted to have a binding site is hypothesized to be regulated by the corresponding transcription factor (20–23).

While the methods mentioned above exploit three different principles, there have been considerable efforts to develop a combined approach to predict regulatory interactions with a higher degree of confidence (10, 24–28). For instance, while analyzing microarray expression data, initially determined sets of co-regulated genes can be refined by investigating whether or not the same transcription factor actually binds to all of them by predicting the presence or absence of a binding site in the promoter regions of these genes. In this way, we can distinguish directly regulated genes from the ones that are regulated through more complicated network motifs or even genes that just randomly happen to show a similar expression profile.

In this chapter, we will primarily focus in detail on the template-based method (12). In order to know more about the other methods discussed, the reader is asked to refer to other chapters in this book, which explicitly deal with network reconstruction procedures using gene expression data and binding site data. For further information, the reader might visit <http://www.mrc-lmb.cam.ac.uk/genomes/madanm/blang/methods/>. In this companion website, we have provided a comprehensive review of additional published work that exploits these different methods.

2. Methods

The set of all transcriptional regulatory interactions within a cell can be conceptualized as a network, which is best modeled as a graph (10, 11). In such a network, nodes represent genes that are transcription factors or targets and edges represent direct transcriptional regulatory interaction. A number of recent studies on transcriptional networks in prokaryotes and eukaryotes have shown that the structure of such networks can be differentiated into three distinct levels of organization (10). At the most basic level, the network consists of a single regulatory interaction between a transcription factor and its target gene (Fig. 8.1a). At the intermediate level of organization, studies have uncovered that the basic unit is organized into fundamental building blocks of transcriptional regulation, called network motifs (Fig. 8.1b). These motifs are discrete functional units and are defined as small patterns of interconnections that are seen in several different contexts within the network (6, 29). Finally, at the global level of organization, the set of all transcriptional regulatory interactions in a cell forms the global structure, which has been shown to have a



Structure of the transcriptional regulatory network

Fig. 8.1. Organization of the transcriptional regulatory network into (a) components, (b) local structure, and (c) global structure. *Black and gray circles* represent transcription factors (TFs) and target genes (TGs) and an *arrow* represents a direct regulatory interaction.

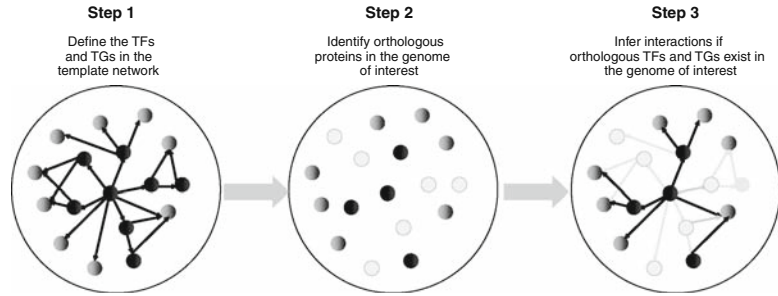
hierarchical and a scale-free topology (30–32). In other words, the global structure is characterized by the presence of many transcription factors, which regulate few genes, and the presence of a few transcription factors, the regulatory hubs, which regulate many genes (Fig. 8.1c).

In the following section, we describe the network reconstruction procedure to reconstruct conserved transcriptional regulatory interactions in a genome of interest using a template network (12). Having obtained the reconstructed networks, we then describe methods to analyze the networks at different levels of organization and methods to assess significance of the evolutionary conservation. Finally, we will also describe methods to correlate the conserved network structure with the lifestyle of the organism in order to obtain insights into interactions that are particularly important for the organism of interest. Throughout this section, we will be describing the methods by using the *E. coli* transcriptional network as the template network.

2.1. Procedure to Reconstruct the Transcriptional Network in a Genome of Interest Using a Template Network

2.1.1. Network Reconstruction Procedure

1. The transcriptional regulatory network for *E. coli* was used as the basis to reconstruct networks for other genomes simply because this is one of the best characterized bacterial networks that are currently available. Information about regulatory interactions was obtained from RegulonDB (8). Thus the template network consisted of 1,295 transcriptional interactions involving 755 proteins (112 transcription factors).
2. Orthologous proteins were identified in the genome of interest using the method described below. If orthologs were identified for an interacting transcription factor and target gene, then an interaction was inferred to be present in the genome of interest (Fig. 8.2). Note that this method can be readily extended to any starting template network.

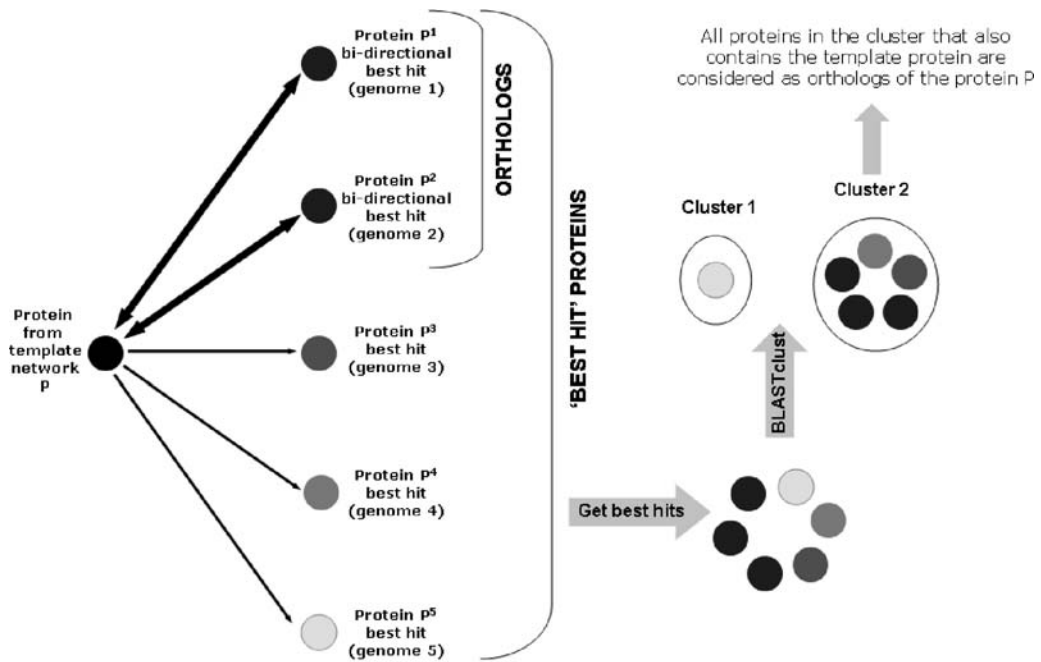


Template based method to reconstruct transcriptional network

Fig. 8.2. Method to reconstruct the transcriptional network for a genome of interest starting from an experimentally characterized template regulatory network. *Black circles* represent transcription factor proteins in the network, *gray circles* represent target genes, and *black arrows* represent direct regulatory interaction. *Light gray circles* represent proteins that are absent in the genome of interest for which an interaction is known in the template network.

2.1.2. Orthology Detection Procedure

Detecting orthology is a non-trivial exercise and can be confounded by paralogs or sequence divergence in a genome of interest. After testing various orthology detection procedures (bidirectional best hit, best hits with defined e-value cutoffs, etc), we arrived at a hybrid procedure that was used to reliably identify orthologous proteins in a genome of interest (*see Note 1 and Fig. 8.3*).



Method to detect orthologs from genomes of interest

Fig. 8.3. Hybrid method to detect orthologous proteins from the genomes of interest.

2.1.2.1. Bidirectional Best-Hit Procedure

1. For each protein \mathbf{P} in the template network, a BLAST search was performed against the genome of interest (\mathbf{x}).
2. The best hit, sequence $\mathbf{P}^{\mathbf{x}}$ from genome \mathbf{x} , was then used as a query and a BLAST search was carried out against the *E. coli* genome.
3. If the best hit using $\mathbf{P}^{\mathbf{x}}$ as the query happens to be \mathbf{P} in the template genome, then \mathbf{P} and $\mathbf{P}^{\mathbf{x}}$ were considered as orthologous proteins.
4. If however $\mathbf{P}^{\mathbf{x}}$ does not pick up \mathbf{P} from the template genome as its best hit, then a BLASTclust procedure was adopted.

2.1.2.2. BLASTclust Procedure

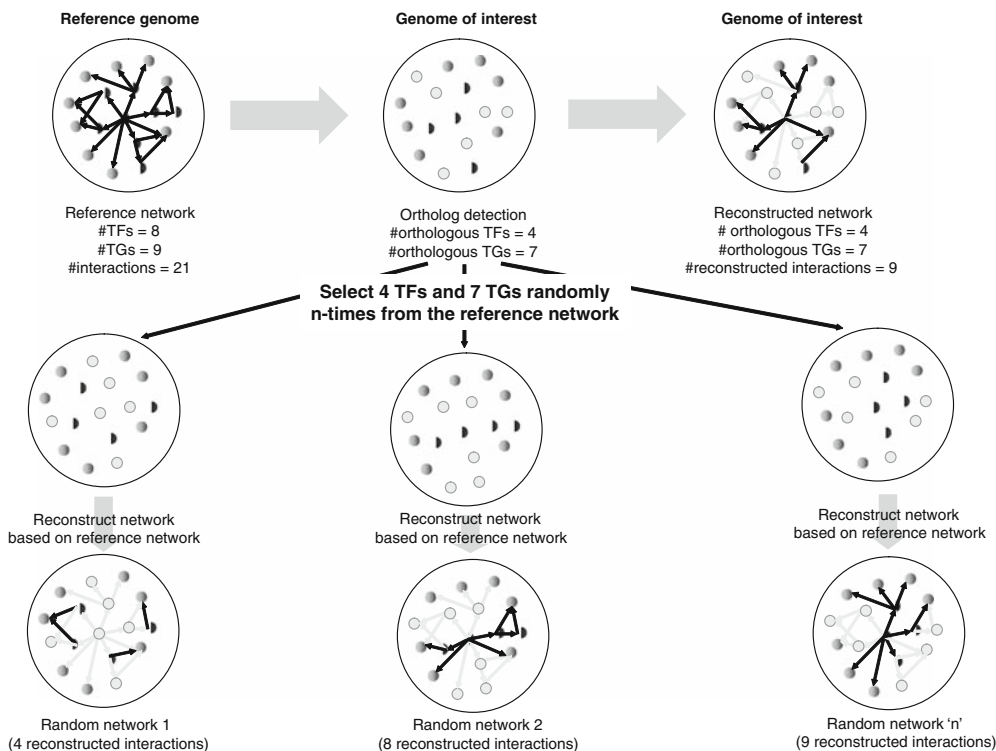
1. For each of the proteins \mathbf{P} in the template network for which the above-mentioned procedure did not pick up orthologous proteins, the best-hit sequences $\mathbf{P}^{\mathbf{x}}$ (using \mathbf{P} as the query against genome \mathbf{x}) for each genome was obtained. Thus, for every protein \mathbf{P} , this procedure gave us a set of proteins, which were the best hits from genomes where the bidirectional best-hit procedure failed.
2. The set of sequences thus obtained along with the query protein was taken through a BLASTclust (33) procedure using length conservation (L) of 60% and a score density (S) of 60% (see **Notes 2, 3, 4**).
3. All sequences belonging to the cluster that also contains the query protein \mathbf{P} from the template network were considered as orthologs (**Fig. 8.3**).

However, it should be kept in mind that these automated procedures for ortholog identification always yield a certain number of false positives.

2.1.3. Method to Create Random Networks to Assess the Significance of Trends Observed in Reconstructed Networks

To assess the significance of trends observed in real network, it is essential to ensure that the observed trends are meaningful and are not something that is expected by chance. To this end, the generation of random networks provides a good way of assessing the statistical significance of the trend. The method below (**Fig. 8.4**) details the procedure to generate several random networks similar to what is seen in the reconstructed network. The random networks generated will be used in the next sections to explain how statistical significance is computed.

1. The number of transcription factors and target genes were defined in the reference network.
2. Orthologs of transcription factors and target genes were detected in the genome of interest and the numbers of TFs, x and TGs, y are noted.
3. To generate the random network, x TFs and y TGs were randomly chosen and the network was reconstructed based on the randomly chosen x TFs and y TGs.



Generating ‘n’ random networks of similar size (as seen in genome of interest)

Fig. 8.4. Procedure to generate random networks to assess statistical significance.

2.2. Methods to Analyze Genes and Regulatory Interactions

2.2.1. Procedure to Analyze the Conservation of Genes and Regulatory Interactions

1. Interactions in the template network were ordered and indexed. For every genome, the reconstructed network was represented as a vector of 1 and 0. 1 represents the presence of an interaction and 0 represents the absence of an interaction. Note that a similar vector can be generated to create a transcription factor presence/absence profile for all the genomes of interest.
2. Having constructed the vector for every genome, the distance between the vectors (which represents the similarity in the interactions conserved between a pair of genomes) was calculated (Fig. 8.5). A tree representing the similarity of interactions (or genes) conserved in the different genomes was obtained using the distance matrix.
3. Alternatively, the vectors can be clustered using standard clustering programs such as Cluster (34) and visualized using Matrix2png (35). This provides a visual representation of the interactions conserved in the genomes of interest (see Notes 5, 6, 7).
4. It should be noted that a similar exercise performed on the presence/absence profile for transcription factors would allow us to group genomes based on similar transcription factor content.

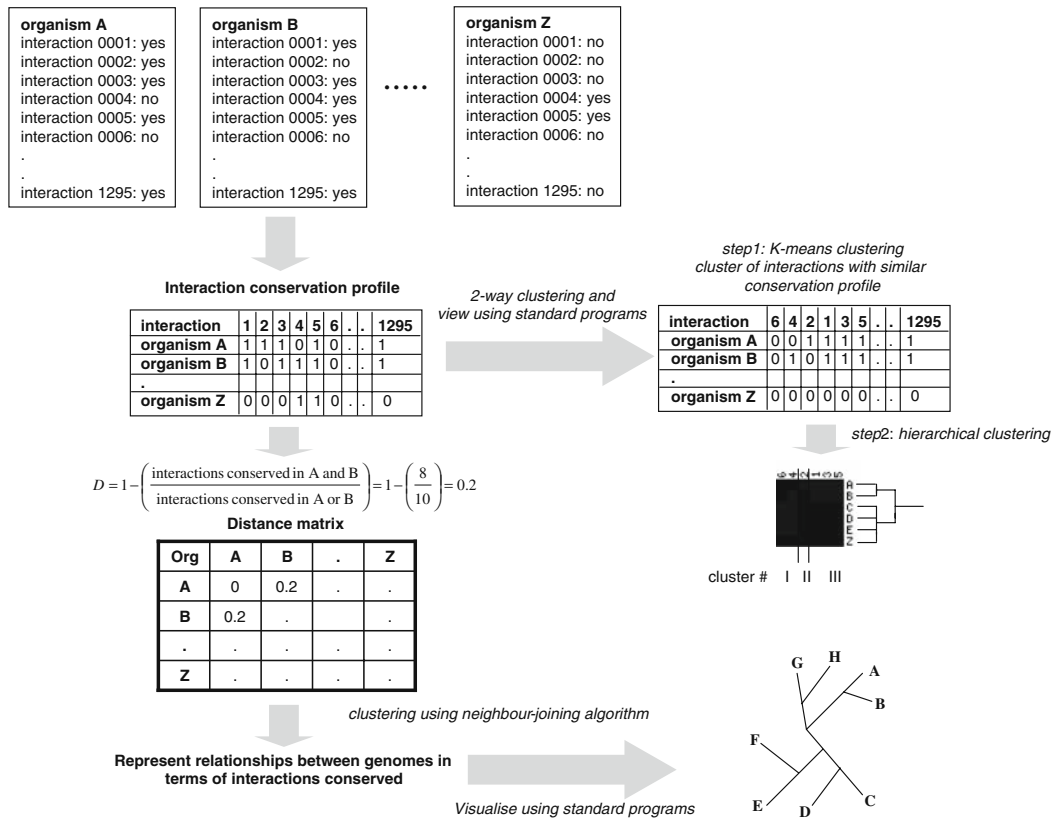


Fig. 8.5. Method to analyze interactions and genes conserved in the genomes of interest.

2.2.2. Procedure to Analyze the Significance of Conservation of Genes and Interactions

To assess if the genome of an organism living in a particular environment has conserved regulatory interactions differently from what would be expected by chance, we employ the following procedure.

1. For each of the genome of interest, we generated 10,000 random networks as described in point 1 of **Section 3.1**.
2. For each of the genomes of interest, the mean m and standard deviation s of the fraction of interactions (or genes) conserved for the 10,000 random networks and the reconstructed network were obtained.
3. The P -value, a measure of statistical significance, was calculated as the fraction of the runs where the fractional conservation was greater than or equal to the observed value for a genome of interest.
4. The Z -score, a measure of how significantly the value deviates from the expected value, was calculated as $Z = (m^{obs} - m^{mean})/s$.

2.3. Methods to Analyze Local Network Structure

Studies on the local level of transcriptional regulatory network have elucidated the presence of small patterns of interconnections called network motifs. It is now generally accepted that three kinds

of network motifs dominate these networks in prokaryotes and eukaryotes. These are the (i) feed forward motif, (ii) single input motif, and (iii) multiple input motif. In the following section, we describe the procedure to analyze the conservation of network motifs in the genomes of interest.

2.3.1. Procedure to Analyze the Conservation of Network Motifs

1. Network motifs in the template network were identified using the Mfinder program.
2. All identified motifs in the template network were ordered and indexed.
3. A motif was considered to be absolutely conserved from a template to a target genome if all the genes constituting the motif in the template network were found to be conserved in the genome of interest. If some genes were missing, the fraction of conserved interactions in the motif was noted (Fig. 8.6).
4. Thus for each genome, an ordered n -dimensional vector (motif conservation profile) is created, where n is the number of motifs considered. The values represent the fraction of the interactions forming the motifs that are conserved (Fig. 8.6).

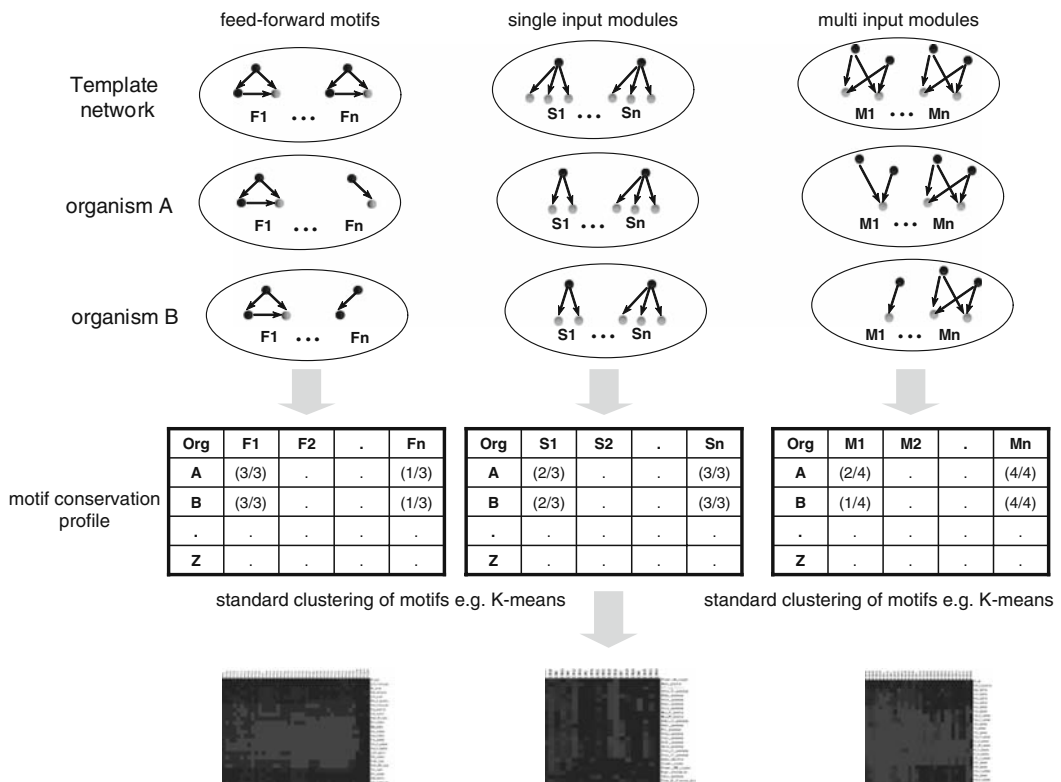


Fig. 8.6. Procedure to analyze conservation of network motifs in the genomes of interest.

5. This matrix was then subjected to the procedure explained in point 1 of **Section 3.2** to identify organisms that have a similar motif conservation profile.

2.3.2. Procedure to Analyze the Significance of Conservation of Network Motifs

1. In order to assess if interactions in motifs are selectively conserved, it becomes important to evaluate whether interactions in a motif are more conserved than any interaction in the network. We introduce a term called conservation index (C.I.) that allows us to assess this trend.

$$\text{C.I.}_{\text{genome } X} = \log_2 \left(\frac{R_{\text{motif}}}{R_{\text{all}}} \right)$$

$$R_{\text{motif}} = \frac{I_{\text{genome } X}^{\text{motif}}}{I_{\text{template}}^{\text{motif}}} \quad \text{and} \quad R_{\text{all}} = \frac{I_{\text{genome } X}^{\text{all}}}{I_{\text{template}}^{\text{all}}}$$

In this definition, $I_{\text{genome } X}^{\text{motif}}$ is the number of interactions that forms a motif in the template network, which has been conserved in genome X . $I_{\text{template}}^{\text{motif}}$ is the number of interactions in a motif in the template network. $I_{\text{genome } X}^{\text{all}}$ is the total number of interactions that have been conserved in genome X and $I_{\text{template}}^{\text{all}}$ is the total number of interactions in the template network (*see Note 8*).

2. To assess if the C.I. value could be obtained by chance, the same value was calculated for 10,000 random networks generated using the procedure described in point 3 of **Section 3.1**.
3. For each of the genomes of interest, the mean m and standard deviation s of the C.I. value for the 10,000 random networks were obtained.
4. The P -value, a measure of statistical significance, was calculated as the fraction of the runs where the value of C.I. was greater than or equal to the observed value for a genome of interest.
5. The Z -score, a measure of how significantly the value deviates from the expected value, was calculated as $Z = (m^{\text{obs}} - m^{\text{mean}})/s$.

2.4. Methods to Analyze Global Network Structure

2.4.1. Procedure to Analyze Global Network Structure

The distribution of outgoing connectivity is a parameter that is indicative of the large-scale structure (topology) of a network. It is well established that the outgoing connectivity for the *E. coli* network follows a scale-free behaviour, i.e., the distribution is best approximated by a power-law function $\mathbf{T} = a\mathbf{K}^{-b}$ where \mathbf{T} is the number of transcription factors with \mathbf{K} connections. To evaluate the distribution for the reconstructed networks we describe the following procedure:

1. For each of the reconstructed networks of the genomes of interest, the distribution was approximated by a linear function ($T = a + bK$), exponential function ($T = ae^{-Kb}$; $\log T = \log a - Kb \log e$) and a power-law function ($T = aK^{-b}$; $\log T = \log a - b \log K$).
2. The function that best approximates the observed distribution is identified as the one that has the lowest standard error.

2.4.2. Procedure to Analyze the Significance of Conservation of Global Network Structure

To identify the trend in random networks the following procedure was carried out:

1. For each of the genomes of interest, we created 10,000 random networks as described in point 3 of **Section 3.1**.
2. The procedure explained above (point 1 of **Section 3.4**) was executed on each of the 10,000 networks to get the function that best approximates the distribution for the random networks.
3. For each of the genomes of interest, the mean m and standard deviation s for the power-law exponent over all 10,000 random networks were computed.
4. As before, the P -value was calculated as the fraction of the runs where the value for the exponent was greater than or equal to the observed value.
5. As before, the Z -score was calculated as $Z = (m^{\text{obs}} - m^{\text{mean}})/s$.

2.5. Method to Co-relate Lifestyle Data with the Conservation of Regulatory Interactions and Network Motifs

2.5.1. Lifestyle-Based Network Similarity Index (LSI)

1. For each organism studied, lifestyle information was collected from the literature and from various sources, including the NCBI genome information website, Brocks Manual of Microbiology, and Bergey's Manual of Determinative and Systematic Bacteriology.
2. The following attributes were used to define the lifestyle class of an organism (*see Note 9*):
 - a. Oxygen requirement (aerobic, anaerobic, facultative, microaerophilic)
 - b. Optimal growth temperature (hyperthermophilic, thermophilic, mesophilic)
 - c. Environmental condition (aquatic, host-associated, multiple, specialized, terrestrial)
 - d. Pathogen (yes, no)
3. The lifestyle (LS) of an organism is defined as a combination of the above four properties. For example, *E. coli* would be classified as "facultative:mesophilic:host-associated:no".

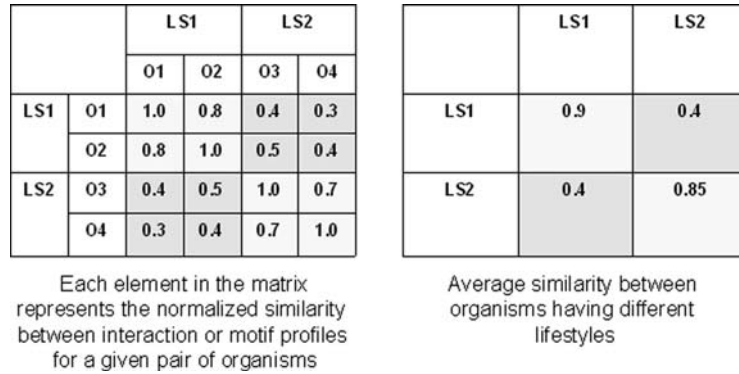


Fig. 8.7. Lifestyle and similarity in the interactions conserved. LS: lifestyle class, 0: organism.

4. Similarity measure between any two organisms was defined as the similarity in the “interaction conservation profile” or the “network motif conservation profile”.
5. Normalized similarity based on the interaction and motif conservation is calculated for each pair of lifestyle classes (Fig. 8.7).
6. Lifestyle-based network similarity index (LSI), which is an indication of how often organisms with similar lifestyle conserve similar interactions or network motifs (*see Note 9*), was calculated as:

$$\begin{aligned}
 \text{LSI} &= \frac{\text{Average similarity between organisms belonging to the same lifestyle}}{\text{Average similarity between organisms belonging to different lifestyles}} \\
 &= \frac{\sum \text{Diagonal elements}}{\sum \text{Off diagonal elements}}
 \end{aligned}$$

2.5.2. Procedure to Assess Significance of the Observed LSI Values

To test the significance of LSI values, we perform randomization experiments.

1. First, 176 random networks of the size similar to each genome studied were generated.
2. Next, the LSI value was calculated for the random network using the definition of the lifestyle class defined in point 1 of **Section 3.5**.
3. This procedure was carried out 1,000 times, and the *P*-value was calculated as the number of times the LSI values in the simulation were greater than the observed value.

4. The *Z*-score was calculated as the ratio of the difference between the observed and the average LSI value to the standard deviation in the observed distribution of LSI values for the 10,000 random networks.

3. Materials

3.1. Hardware Requirements

1. Personal Computer with at least 512 MB memory, 10 GB hard-disk space, and a processor of at least 1 GHz or better.
2. Access to a Linux or a UNIX workstation.
3. Stable connection to the Internet.

3.2. Software Requirements

1. A recent version of PERL installed on a Linux or UNIX environment (freely available for download at: <http://www.perl.com>)
2. A versatile Windows text editor such as TextPad for windows (available at <http://www.textpad.com>) or nedit for Linux
3. A recent version of the NCBI BLAST suite of programs (36) for Linux (freely available from the NCBI website at: <http://www.ncbi.nlm.nih.gov/ftp/>)
4. A recent version of the motif finding program Mfinder (37) for Windows (freely available from the Weizmann Institute at: <http://www.weizmann.ac.il/mcb/UriAlon/groupNetworkMotifSW.html>).

3.3. Template Transcriptional Regulatory Network

Information on transcriptional regulation is available for several model organisms. The following websites provide a list of regulatory interactions in different organisms. The networks have been manually curated in several cases and contain regulatory interactions inferred from large-scale functional genomics experiments in the case of yeast. For the method described in this chapter, we use only the *E. coli* regulatory network as the template. It should be noted at this point that any network can be potentially used as a template network.

1. *Escherichia coli*: RegulonDB (8) (<http://regulondb.ccg.unam.mx/index.html>)
2. *Bacillus subtilis*: DBTBS (38) (<http://dbtbs.hgc.jp/>)
3. *Corynebacterium* species: Coryneregnet (9, 39) (<https://www.cebitec.uni-bielefeld.de/groups/gi/software/coryneregnet/>)

4. *Saccharomyces cerevisiae*: A curation of regulatory interactions from several different small-scale and large-scale studies (40, 41) (<http://www.mrc-lmb.cam.ac.uk/genomes/madanm/tfcomb/tnet.txt>).

3.4. Complete Genome Sequences and Lifestyle Information for Organisms of Interest

1. The complete genome sequence and the predicted proteome of several prokaryotic and eukaryotic genomes can be obtained from the NCBI genomes website at: <http://www.ncbi.nlm.nih.gov/genomes/lproks.cgi>
2. Detailed and systematic information about the lifestyle of different completely sequenced genomes can be obtained from the same website by clicking on the “organism info” tab. If this information is not available, the reader is suggested to obtain it from the publication describing the genome sequence or refer to the Brock’s Biology of Microorganisms or Bergey’s Manual of Determinative and Systematic Bacteriology (available at: <http://www.cme.msu.edu/Bergeys/>)

4. Notes



1. Bidirectional best hit is a very conservative approach to detect orthologs. It performs best for closely related organisms and may fail to pick up orthologs from distantly related organisms. The best-hit method using specific cutoffs is too liberal, and may result in false-positive hits when the genomes compared are distantly related or when there are many closely related paralogs in the genome of interest. So our hybrid orthology detection method uses a combination of both methods as described above.
2. The BLASTclust procedure first carries out an all-against-all sequence comparison and produces clusters of sequences using the single linkage-clustering algorithm. This will ensure that orthologous proteins in distantly related organisms would still be picked up reliably through the sequences from intermediately distant genomes.
3. Manual analysis of the clusters obtained using various combinations of values for the parameters revealed that the parameters score density $S = 0.6$ and length overlap $L = 0.6$ perform best, with an optimal coverage and the lowest false-positive rate.
4. In the BLASTclust algorithm, score density (S) is defined as the ratio of the number of identical residues in the alignment to the length of the alignment. Detailed documentation for BLASTclust is available at: <ftp://ftp.ncbi.nlm.nih.gov/blast/documents/blastclust.html>.

5. Cluster is a versatile program that allows users to cluster vectors representing the conserved network. It provides several clustering methods such as (i) hierarchical clustering, (ii) K-means clustering, and (iii) self-organizing maps. Hierarchical clustering can be done using (a) single linkage, (b) multiple linkage, (c) centroid linkage, and (iv) average linkage methods. The program also allows the use of different distance measures to cluster vectors such as Pearson's correlation coefficient, Euclidean distance, Spearman's rank correlation, etc. In our experience, we find that either K-means clustering or hierarchical clustering using the single linkage method and Pearson's correlation coefficient distance measure gives the best results. The cluster software can be downloaded from: <http://bonsai.ims.u-tokyo.ac.jp/~mdehoon/software/cluster/>.
6. Matrix2png is a simple and powerful program that can be used to visualize the vector representation of the conserved networks in the genomes of interest. It generates PNG format images from tab-delimited text files of vector data. This software can be downloaded from: <http://www.bioinformatics.ubc.ca/matrix2png/>.
7. The distance matrix representing the similarity between the vectors representing the conserved networks can be visualized as a tree by using the treeview package. Treeview can be downloaded from: <http://taxonomy.zoology.gla.ac.uk/rod/treeview.html>.
8. In the calculation of the conservation index (C.I.) for network motifs, the \log_2 of the ratio ensures that selection for and against are represented symmetrically in the graph. For example, if $R_{\text{motif}} = 0.9$ and $R_{\text{all}} = 0.6$, C.I. can be calculated as $\log_2(0.9/0.6) = 0.58$. Thus if interactions in motifs are selected for, then the C.I. value will be greater than 0; if not, the value will be less than 0.
9. It should be noted that other features, such as salinity, pressure, tolerance to damaging radiation, can also be used as additional attributes to define lifestyle class. It is worth mentioning our observation that organisms with similar lifestyle can be very distantly related and organisms that are close evolutionary relatives very often tend to colonize different ecological niches.
10. In the example described in point 1 of **Section 3.5**, LSI can be calculated as: $[(0.90 + 0.85)/2]/[(0.4 + 0.4)/2] = 2.18$ (i.e., the ration of the average of the diagonal elements to the average of the off-diagonal elements). In other words, if the organisms with a similar lifestyle have higher similarity in motif or interaction content than the organisms with dissimilar lifestyle, then the LSI should be greater than 1.

11. From our analysis of the reconstructed regulatory networks, we observed that transcription factors are typically less conserved than their target genes and evolved independently of them, with different organisms evolving distinct repertoires of the transcription factors responding to specific signals. We identified that prokaryotic transcriptional regulatory networks have evolved principally through widespread tinkering of transcriptional interactions at the local level by embedding orthologous genes in different types of regulatory motifs. Different transcription factors appear to have emerged independently as dominant regulatory hubs in various organisms, suggesting that they have convergently acquired similar network structures approximating a scale-free topology. We also noted that organisms with similar lifestyles across a wide phylogenetics range tend to conserve equivalent interactions and network motifs. Thus, it appears that organism-specific optimal network designs have evolved due to the selection for specific transcription factors and transcriptional interactions that allowed responses to prevalent environmental stimuli. The methods for biological network analysis introduced here can be applied generally to study other networks, and the predictions available in the supporting website (<http://www.mrc-lmb.cam.ac.uk/genomes/madanm/evdy/>) can be used to guide specific experiments.

Supplementary Information

See <http://www.mrc-lmb.cam.ac.uk/genomes/madanm/blang/methods/> for an overview of literature in the field of transcriptional network reconstruction.

See <http://www.mrc-lmb.cam.ac.uk/genomes/madanm/evdy/> for detailed supplementary material and reconstructed networks for 175 prokaryotic genomes.

Acknowledgments

MMB acknowledges the Medical Research Council, UK, for financial support. LA acknowledges the Intramural Research Program of the NIH, NLM, NCBI, USA, for funding. We thank Arthur Wuster for critically reading this manuscript.

References

- Steinmetz LM, Davis RW. Maximizing the potential of functional genomics. *Nat Rev Genet* 2004, 5:190–201.
- Quackenbush J. Computational analysis of microarray data. *Nat Rev Genet* 2001, 2:418–27.
- Young RA. Biomedical discovery with DNA arrays. *Cell* 2000, 102:9–15.
- Bulyk ML. DNA microarray technologies for measuring protein-DNA interactions. *Curr Opin Biotechnol* 2006, 17:422–30.
- Harbison CT, Gordon DB, Lee TI, et al. Transcriptional regulatory code of a eukaryotic genome. *Nature* 2004, 431:99–104.
- Lee TI, Rinaldi NJ, Robert F, et al. Transcriptional regulatory networks in *Saccharomyces cerevisiae*. *Science* 2002, 298:799–804.
- Horak CE, Luscombe NM, Qian J, et al. Complex transcriptional circuitry at the G1/S transition in *Saccharomyces cerevisiae*. *Genes Dev* 2002, 16:3017–33.
- Salgado H, Gama-Castro S, Peralta-Gil M, et al. RegulonDB (version 5.0): *Escherichia coli* K-12 transcriptional regulatory network, operon organization, and growth conditions. *Nucl Acids Res* 2006, 34:D394–97.
- Baumbach J, Brinkrolf K, Czaja LF, Rahmann S, Tauch A. CoryneRegNet: An ontology-based data warehouse of corynebacterial transcription factors and regulatory networks. *BMC Genomics* 2006, 7:24.
- Babu MM, Luscombe NM, Aravind L, Gerstein M, Teichmann SA. Structure and evolution of transcriptional regulatory networks. *Curr Opin Struct Biol* 2004, 14:283–91.
- Barabasi AL, Oltvai ZN. Network biology: Understanding the cell's functional organization. *Nat Rev Genet* 2004, 5:101–13.
- Madan Babu M, Teichmann SA, Aravind L. Evolutionary dynamics of prokaryotic transcriptional regulatory networks. *J Mol Biol* 2006, 358:614–33.
- Yu H, Luscombe NM, Lu HX, et al. Annotation transfer between genomes: Protein-protein interologs and protein-DNA regulogs. *Genome Res* 2004, 14:1107–18.
- Lozada-Chavez I, Janga SC, Collado-Vides J. Bacterial regulatory networks are extremely flexible in evolution. *Nucl Acids Res* 2006, 34:3434–45.
- Segal E, Shapira M, Regev A, et al. Module networks: Identifying regulatory modules and their condition-specific regulators from gene expression data. *Nat Genet* 2003, 34:166–76.
- Gardner TS, di Bernardo D, Lorenz D, Collins JJ. Inferring genetic networks and identifying compound mode of action via expression profiling. *Science* 2003, 301:102–05.
- Basso K, Margolin AA, Stolovitzky G, Klein U, Dalla-Favera R, Califano A. Reverse engineering of regulatory networks in human B cells. *Nat Genet* 2005, 37:382–90.
- Wang SC. Reconstructing genetic networks from time ordered gene expression data using Bayesian method with global search algorithm. *J Bioinform Comput Biol* 2004, 2:441–58.
- Qian J, Lin J, Luscombe NM, Yu H, Gerstein M. Prediction of regulatory networks: Genome-wide identification of transcription factor targets from gene expression data. *Bioinformatics* 2003, 19:1917–26.
- Alkema WB, Lenhard B, Wasserman WW. Regulog analysis: Detection of conserved regulatory networks across bacteria: Application to *Staphylococcus aureus*. *Genome Res* 2004, 14:1362–73.
- Wang T, Stormo GD. Identifying the conserved network of cis-regulatory sites of a eukaryotic genome. *Proc Natl Acad Sci USA* 2005, 102:17400–05.
- Rodionov DA, Dubchak I, Arkin A, Alm E, Gelfand MS. Reconstruction of regulatory and metabolic pathways in metal-reducing delta-proteobacteria. *Genome Biol* 2004, 5:R90.
- Bar-Joseph Z, Gerber GK, Lee TI, et al. Computational discovery of gene modules and regulatory networks. *Nat Biotechnol* 2003, 21:1337–42.
- Xing B, van der Laan MJ. A statistical method for constructing transcriptional regulatory networks using gene expression and sequence data. *J Comput Biol* 2005, 12:229–46.
- Haverty PM, Hansen U, Weng Z. Computational inference of transcriptional regulatory networks from expression profiling and transcription factor binding site identification. *Nucl Acids Res* 2004, 32:179–88.
- Gao F, Foat BC, Bussemaker HJ. Defining transcriptional networks through integrative modeling of mRNA expression and transcription factor binding data. *BMC Bioinformatics* 2004, 5:31.

27. Bussemaker HJ, Li H, Siggia ED. Regulatory element detection using correlation with expression. *Nat Genet* 2001, 27:167–71.
28. Kim H, Hu W, Kluger Y. Unraveling condition specific gene transcriptional regulatory networks in *Saccharomyces cerevisiae*. *BMC Bioinformatics* 2006, 7:165.
29. Shen-Orr SS, Milo R, Mangan S, Alon U. Network motifs in the transcriptional regulation network of *Escherichia coli*. *Nat Genet* 2002, 31:64–68.
30. Albert R. Scale-free networks in cell biology. *J Cell Sci* 2005, 118:4947–57.
31. Teichmann SA, Babu MM. Gene regulatory network growth by duplication. *Nat Genet* 2004, 36:492–96.
32. Guelzim N, Bottani S, Bourgine P, Kepes F. Topological and causal structure of the yeast transcriptional regulatory network. *Nat Genet* 2002, 31:60–63.
33. Lespinet O, Wolf YI, Koonin EV, Aravind L. The role of lineage-specific gene family expansion in the evolution of eukaryotes. *Genome Res* 2002, 12:1048–59.
34. de Hoon MJ, Imoto S, Nolan J, Miyano S. Open source clustering software. *Bioinformatics* 2004, 20:1453–54.
35. Pavlidis P, Noble WS. Matrix2png: A utility for visualizing matrix data. *Bioinformatics* 2003, 19:295–96.
36. Altschul SF, Madden TL, Schaffer AA, et al. Gapped BLAST and PSI-BLAST: A new generation of protein database search programs. *Nucl Acids Res* 1997, 25:3389–402.
37. Kashtan N, Itzkovitz S, Milo R, Alon U. Efficient sampling algorithm for estimating subgraph concentrations and detecting network motifs. *Bioinformatics* 2004, 20:1746–58.
38. Makita Y, Nakao M, Ogasawara N, Nakai K. DBTBS: Database of transcriptional regulation in *Bacillus subtilis* and its contribution to comparative genomics. *Nucl Acids Res* 2004, 32:D75–77.
39. Baumbach J, Wittkop T, Rademacher K, Rahmann S, Brinkrolf K, Tauch A. CoryneRegNet 3.0-An interactive systems biology platform for the analysis of gene regulatory networks in corynebacteria and *Escherichia coli*. *J Biotechnol* 2007 Apr 30;129(2):279–89.
40. Balaji S, Babu MM, Iyer LM, Luscombe NM, Aravind L. Comprehensive analysis of combinatorial regulation using the transcriptional regulatory network of yeast. *J Mol Biol* 2006, 360:213–27.
41. Balaji S, Iyer LM, Aravind L, Babu MM. Uncovering a hidden distributed architecture behind scale-free transcriptional regulatory networks. *J Mol Biol* 2006, 360:204–12.

Chapter 9

Learning Global Models of Transcriptional Regulatory Networks from Data

Aviv Madar and Richard Bonneau

Abstract

Organisms must continually adapt to changing cellular and environmental factors (e.g., oxygen levels) by altering their gene expression patterns. At the same time, all organisms must have stable gene expression patterns that are robust to small fluctuations in environmental factors and genetic variation. Learning and characterizing the structure and dynamics of Regulatory Networks (*RNs*), on a whole-genome scale, is a key problem in systems biology. Here, we review the challenges associated with inferring *RNs* in a solely data-driven manner, concisely discuss the implications and contingencies of possible procedures that can be used, specifically focusing on one such procedure, the *Inferelator*. Importantly, the *Inferelator* explicitly models the temporal component of regulation, can learn the interactions between transcription factors and environmental factors, and attaches a statistically meaningful weight to every edge. The result of the *Inferelator* is a dynamical model of the *RN* that can be used to model the time-evolution of cell state.

Key words: Network inference, biclustering, network reconstruction, microarray, data-integration, cMonkey, archaea, the Inferelator.

1. Introduction

1.1. Introduction and Structure of this Chapter

In this chapter we discuss the methods for learning *Transcriptional Regulatory Networks (TRNs)* from data in a cost-effective manner. There are numerous mechanisms by which cells regulate the activity and relative concentrations of different gene products including, but not limited to, chromatin structure, transcriptional regulation, translational regulation, post-translational modification, and degradation. In the end, mechanisms for biological regulation define the control of a wide number of possible cell states: developmental, physiological, etc. It is well agreed that transcription factors (*TFs*) determine a major part of the gene

expression profile within each cell type. *TFs*, in turn, are controlled by the cell's environment. Taken together, *TFs* or regulators and their effect on their target genes constitute a *TRN*. In this review, we will (1) define the problem of *TRN* inference; (2) shortly review the relevant experiments and data types that we consider relevant for *TRN* inference, and discuss their implications, limitations, and contingencies; (3) discuss the design principles that underlie *TRNs* and the cognate challenges associated with inferring *TRNs*; (4) generally outline, what we consider to be, the requirements of a good solution for *TRN* inference; (5) introduce Bayesian networks and their possible use for *TRN* inference followed by a description of one previously described method of our own construction; and, lastly, (6) summarize the chapter and conclude that for prokaryotic systems, we can already learn predictive dynamical models from sequence, expression, and *ChIP-chip* data. We do not aim to provide the reader, herein, with an encyclopedic knowledge of all methods for *TRNs* inference, rather intend to introduce a complete set of foundational concepts and then end with a complete description of one method, the *Inferelator* (1).

1.2. Classical and Modern Study of *TRNs*

The study of genetic regulatory networks has evolved greatly in the past 10 years due to the influx of high-throughput technologies, but has been central to molecular biology since the inception of classical biochemistry and genetics. Jacob and Monod's groundbreaking work on the Lac operon (**Note 1**) (2) provided an initial paradigm for genetic control in all cells, and helped to establish the concept that certain proteins can interact with DNA regulatory sequences and small molecules, and that their interactions can bring about adaptive changes in cellular state. Following their work, several other small-scale *TRNs* were resolved to a great extent, such as the development of the sea urchin embryo (3, 4) and the progression of cell cycle in yeast (5). However, the experimental methods available at the time limited their scope, while also requiring a labor- and time-intensive study over several decades for their construction. Today, with many genome projects completed, and emerging genomic techniques enabling system-wide measurement of key mRNA, protein, and protein–DNA affinity, a significant volume of research is directed at developing learning methods – computational, machine-learning, and statistical methods – that use genome-wide data to produce models of *TRNs* which can predict the dynamical behavior of cells in diverse cell states. The introduction of microarrays – assays that measure global mRNA levels – initially spurred a large effort to improve our ability to elucidate *TRNs* directly from data. Other technologies – including proteomics, metabolomics, and studies of protein post-translational modification – further increase the need for methods that learn directly from data.

2. Review of Functional- Genomics Data Types with Respect to TRNs

2.1. Microarrays

We start with a short review of the most common system-wide techniques currently available for researchers looking at global regulation and their implications for network inference.

Microarrays are predominantly used to explore the profile of gene mRNA expression pattern of thousands of genes simultaneously. Briefly, on a substrate, a dense two-dimensional grid is created by either chemically synthesizing or depositing short DNA oligonucleotides at specific locations. Each short oligonucleotide and its location on the grid can be called a feature, and many features can be contained on one array spanning the genome of interest. Depending on the technology used, each array can have several thousand to approximately one million features. Each probe is designed to recognize a specific gene by the process of complementary hybridization. By extracting and labeling mRNAs from experimental samples, and then hybridizing those purified mRNAs to the array, the amount of labeled mRNA specifically binding to each complementary feature can be estimated at each feature, by measuring the fluorescence intensity at each feature's location. The overall process enables a global measurement of the expression level in a given sample. The necessity to perform biological replicates is undisputed as the measurement of the biological repeats allows us to assess the level of measurement variability and biological variance, thus enabling statistically sound downstream analysis. Multiple features on the chip can correspond to the same gene, thus providing a form of replicate measurement commonly referred to as a technical replicate. Alternatively, repeating the experiment wholesale on similarly prepared but separate biological sample can be referred to as a biological replicate. Both types of replicates are necessary for high-quality data. Depending on the biological question, and the experimental design, microarray data preprocessing – e.g., normalization (**Note 2**) and transformation (**Note 3**) – is commonly done with the purpose of reducing the inherent noise and facilitating further analysis of the expression profile.

A wide range of methods for microarray data analysis have evolved, ranging from simple fold-change approaches that test for differential gene expression between two experimental conditions to more complex and computationally demanding techniques. Many early techniques cluster genes – i.e., grouping genes with correlated expression over all different experimental conditions. However, when analyzing a large data set, composed of hundreds to thousands of experimental conditions, a large portion of the genes may be co-expressed under a subset of the experimental conditions in the data set. Pre-grouping co-expressed

genes under subsets of relevant conditions is termed “biclustering” (**Note 24**), and several algorithms that do so have been described (6–12). As clustering is a key first step prior to or in conjunction to *TRN* inference, it receives a thorough discussion at the Notes section.

To date mRNA measurements have played an essential role in learning *TRNs*, due to their availability, cost-effectiveness, and reliability when used properly. Global measurements of mRNA (by Microarrays, LYNX MPSS (13), SAGE (14) or as of yet unimagined technologies) will continue to be a central technology supporting *TRN* inference, as changes in transcription (of mRNA) are the direct effects of transcriptional control by any mechanism. Microarrays are not, however, in-and-of-themselves sufficient for elucidating *TRNs*; *TFs* are active at the protein level, not the mRNA level.

Limitations and contingencies. (1) Methods should be designed to tolerate error expected with microarray measurements. (2) A large number of experimental repeats are needed to allow a later, rigorous statistical analysis, bringing the cost up. (3) There is no standard analysis method for microarrays, making the comparison of results obtained from different groups problematic. We must, therefore, combine the data obtained from microarrays with additional complementary technologies as detailed below.

2.2. Direct Assays of Protein–DNA Interactions

Direct measurement of possible regulatory factors binding sites in vivo (and at different cell states) has recently become possible on a genome-wide scale with the emergence of *Chromatin immunoprecipitation (Note 4) on-chip (ChIP-chip)* (15–17). The purpose of *ChIP-chip* is to determine whether a protein, most commonly a *TF*, binds to particular regions in the genome – e.g., promoters and enhancers that the *TF* recognizes. Briefly, DNA-binding proteins are covalently bound to the DNA on which they are situated, via in vivo cross-linking. Once the proteins have been immobilized on the chromatin, DNA is broken to short segments (0.2–1 Kb) by sonication, or some other method of fragmentation. The protein–DNA complexes are then immunoprecipitated using an affinity purification method (the protein of interest being “tagged” prior). In the original protocol this was achieved by adding a multi-protein repeat to the termini of the proteins of interest and then using an antibody to this “tag” to purify only the specific protein of interest. The result is an enrichment of the DNA sequence fragments that were bound by the *TF* under examination. The fraction of DNA/protein is then separated, the cross-links are reversed, and the resulting DNA sequence (the target of the *TF*) is fluorescently labeled. At this stage the isolated DNA is hybridized to a microarray that represents the intergenic-regulatory sequences of the organism. The resulting analysis ideally

pinpoints the positions on the DNA where the *TF* is binding and the sequences it recognizes. The regulatory regions can then be mapped to the genome and suggest which genes are being regulated by the *TF*. Results from several *ChIP-chIP* experiments can be combined to produce a putative protein–DNA interaction network.

Limitations and contingencies: (1) The method is labor-intensive – requiring an experiment per protein of interest, per cell state of interest. (2) It measures protein–DNA interactions, but does not reveal whether the binding is functional, non-functional, activator, repressor, part of a multi-protein complex, etc. (3) Moreover, the results can suffer from a large percent of false positives (**Note 5**) and false negatives (**Note 6**). Thus, integration with expression and protein measurements is required for using this data as part of a *TRN* inference procedure.

2.3. Functional Annotations (Constraints from Prior Functional Information)

For illustration purposes we focus here on the *gene ontology* (*GO*) database (18); however, other databases for functional annotations are available, to name a few: *KEGG* and *EcoCyc* (19, 20) for metabolic pathways, *TRANSEAC* (21) for protein–DNA interactions and *SWISS-PROT* (22) that provides high level of annotation such as the protein function, its domain structure, and post-translational modifications. The *gene ontology* encodes a consistent description of function, localization, and cellular processes of gene products. The *GO* project has three major vocabularies (ontologies) that describe gene products in terms of their associated biological processes, cellular components, and molecular functions in a species-independent manner. The ontologies are structured as directed acyclic graphs, which are similar to hierarchies but differ in that a child, a more specialized term, can have multiple parents. Commonly, researchers check for co-function of genes in order to decide if a group of genes (e.g., a cluster) shares a function and then use that information to guide clustering or classification of functional modules prior to inference of *TRNs*. A commonly accepted assumption is that genes that share the same function – e.g., *GO* molecular function – are likely to be co-regulated. Even in primarily data-driven methods we can use co-association of functional terms as an overall sanity check after running the method in question, i.e., this step is usually done after the biclusters have been assigned and is used as overall check of the validity of the results.

Limitations and contingencies: (1) Out of all the annotated genes, only a small portion has been experimentally determined. (2) Annotations for non-model organisms had been determined based on homology to experimentally verified functional annotations; thus the annotations for small or model organisms will tend to be much more useful than for humans, for example. (3) A given gene may not share an exact term with another gene but may be

connected to it via a common parent; there is no standard protocol for deciding how near in the hierarchy the parent should be to designate the genes as sharing function. Therefore, although *GO* terms can be useful, they too should be considered indicative of gene function, and preferably integrated with the analyses of microarrays and other data types.

2.4. Protein–protein interactions

Yeast Two-Hybrid (Y2H) and Mass Spectrometry (MS): Multi-protein complexes are commonplace in the cellular environment and critical to nearly all biological processes. It has been observed that often these molecular machines are co-regulated and share *TF* binding sites in their regulatory region. Conversely, if proteins are shown to interact, this can suggest that they are co-regulated. Y2H (23) and MS (24) are tools that suggest physical interactions between proteins; although these methods have no direct relevance for regulatory network inference, they are often used to constrain learning of regulatory modules (12, 25). We will not focus heavily on the use of physical interactions in constraining *TRNs* inference herein.

2.5. Systems Biology Data Quality

The data types discussed above are relatively novel, and a complete discussion of the dramatically different data qualities one can expect from different applications of these technologies is beyond the scope of this chapter. Factors affecting data quality range from those associated specifically with different technologies (different platforms, different methods) to factors associated with experimental design (such as sampling rate, number of replicates used per experiment) to the error associated with random (or non-systematic) noise. One of the most important concepts in designing and carrying out a biological assay is to enforce strict quality control measures. For example, microarrays explore the mRNA levels of thousands of genes in a single experiment, and thus have many possible factors that can bias the results.

Therefore, it is important that variable factors be well controlled and well accounted for if the desire in a given experiment is to probe the effect at the transcript level of a single factor. Combinatorial design – i.e., varying multiple genetic and/or environmental factors simultaneously – can also be useful if the end goal is to integrate all data for a given organism to infer global regulatory networks. Overall, correct experimental design is essential to the success of any network inference procedure (26), but is not discussed here. Several examples of the measures taken to ensure quality of microarray experiments follow: (1) the use of arrays from the same similar batch or the use of assays with some measure of continuity, this goes for all other consumables such as dyes, prep kits, etc. (all of which can have strong, sometimes sequence dependant, effects on experimental outcome); (2) All experiments need to be done with uniform lab protocols, with all

meta-data – i.e., data describing experimental setup and environmental factors – properly recorded; (3) Having a large number of biological replicates (**Note 7**) for each experiment is essential for providing a statistical backbone for the results. Lack of sufficient number (**Note 8**) of replicates may produce erroneous data that will derail future analysis including network inference; (4) Data preprocessing including normalization and transformation is often used to reduce the variability across the different experiments and allow the data collected from different groups and possibly different platforms to be used under one data set. Other data types have multiple similar considerations that must be considered, e.g., *ChIP-chip* produces a large number of false positives and hence should be used as suggestive and not necessarily recapitulating real biological relevant protein–DNA interactions.

An emerging consensus is that time series measurements, i.e., dynamical observations, are essential to learning truly causal relationships, and thus true regulatory interactions, and this need for kinetic (or time-series) data is well established in other fields, such as financial forecasting (1). We will address this experimental design consideration more below, in **Section 4**.

3. Design Principles Characteristic for TRNs

As transcriptional regulatory networks are being characterized with increasing speed, it is worthwhile to look for underlying principles in the design of these control systems (27, 28). Let us begin by considering a biological example – The induction of the Lac operon in *E. coli* – presenting common requirement for TRNs. When lactose is the only carbon source available (**Note 9**), *E. coli* induces the Lac operon, which contains LacZ, LacY, and LacA. The regulatory network is expected to (1) maintain a stationary basal level of β -galactosidase (the lacZ gene product) in the absence of lactose (**Note 10**); (2) dynamically increase the expression level when the level of lactose increases; and (3) maintain the high level of expression as long as stationary levels of lactose are present. Already in the simple example of the Lac operon three TRN design principles are illustrated, namely: (1) the regulatory network function requires the induced and the basal states to be stable; (2) the system has to operate in a manner that is robust to environmental changes; and (3) the system has to respond quickly to changes in metabolites, it has to be induced quickly in response to lactose. Stability, robustness, and responsiveness are therefore intrinsic properties of the lactose system, and are prevalent attributes of many TRNs. Four common themes seem to be intrinsic to many biological regulatory networks – depending on the function of the regulatory network – and we will go through these themes one at a time:

1. Stability: the ability of a system to return to steady state after a transient disturbance.
2. Robustness: the ability of a system's steady state to remain unchanged, or not significantly changed, when the parameter values of the system significantly changes.
3. Responsiveness: the ability of a system to settle quickly into a new steady state after an environmental change.
4. Modularity: the structure of the system regulatory network is separated into modules; each module has a function and is highly connected (contain many interactions) within, but has fewer interactions with other modules.

Importantly, robustness, or insensitivity to parameter variation, has long been used as performance criterion for *TRN*. All or several of these design principles are expected to be present in any *TRN*. Ideally, criteria based on these network properties could be used to decide between alternate regulatory network models or possibly used as priors.

4. The Challenge of Inferring *TRNs*

Deciphering *TRNs* from large genomic, proteomic, and expression data sets is one of the most interesting problems for computational biologists. It is clear that many challenges are still waiting to be met by the top researchers as well as the coming new generation of researchers in the field of computational biology.

4.1. Challenges and Possible Solutions

1. The dimensionality of the data sets used for network inference is vast, composed of thousands of genes over hundreds to thousands of conditions. Add to that the number of possible predictors and we have a very large problem space indeed. To reduce the dimensionality of this problem, with little cognate loss in predictive performance, we can perform several steps either prior to network inference or in conjunction with the network inference procedure. A common technique used is to group the genes together based on clustering or biclustering algorithms. After such a step, one can reduce the expression patterns of all the individual genes in the cluster into one expression pattern of the bicluster (one way is to take the mean expression of all the genes). Another common technique is to limit the number of predictors to well-characterized genes known to act as transcription factors.
2. As combinatorial control has been shown to be commonplace in *TRNs* – i.e., genes are regulated by combinations of *TFs* (Note 11) and different combinations account for different expression patterns – it should be incorporated in some way in

all *TRN* learning algorithms. A flawed simplification, often made, is to model elementary *TRN* in which gene expression is regulated by a single *TF* or additive combinations of *TFs*. This representation, however, fails to capture combinatorial control (logic) – “AND” cannot be modeled by additive combinations of single predictors – and an explicit interaction term is needed, to model even the simplest of regulatory logic (*see Section 6.4* for details). A biologically driven *TRN* inference procedure needs to allow for explicit modeling of combinatorial interactions between the regulators. However, future methods must strike a balance between having the flexibility to model combinatorial *TF* interactions, resulting in a more complex network, and overfitting the model to the training set, thus losing predictive power.

3. Two major types of microarray experimental designs are (1) equilibrium or steady state (**Note 12**), and (2) time series or kinetic (**Note 12**). Importantly, time series data allow us to observe the dynamics that underlie the shift from one transcription profile to a new one. For example, the genes composing the flagellar motor in *Caulobacter crescentus* are transcribed in one of four distinguishable, consecutive stages of the bacteria’s life cycle, as elegantly shown by the time series microarray experiments analyzing *Caulobacter*’s cell cycle (29). One drawback is that extensively sampled time series are expensive to collect (multiple time points per perturbation/cell state) and thus a cost-effective balance between equilibrium and time-series designs may be ideal for many inference/functional genomics projects. Both the experiment design types, when analyzed simultaneously, work synergistically, providing supporting evidence for a given regulatory model. Therefore, it is beneficial to have an algorithm that can learn from both the experiment design types simultaneously.
4. A *TRN* model should, ideally, describe the system and its response to new combinations of environmental and genetic factors – i.e., a model that can predict the transcriptional response to newly measured cell states. Global prediction of system behavior when the system is confronted with novel stimulatory factors or a novel combination of perturbations is essential, but is rarely achieved in previously described studies. Moreover, the reader should be aware that there is little agreement in the manner by which different regulatory network inference methods developed to date have been validated, i.e., the validity and strength of different methods developed to infer regulatory systems from global genomic data sets is judged in many ways, across the works published to date. One possible way to determine the validity, and also

the usefulness (**Note 14**), of the different methods is their ability to predict new expression states as functions of transcription factors (or any other predictors of transcription that are used within a given model).

5. Properties of a Good Solution for the Problem of TRN Inference

Before moving forward or discuss examples of detailed solutions to the problem of *TRN* inference, we want to, at the risk of redundancy, summarize what we consider a good solution to the problem. The following points characterize some general features of what we consider a “good solution”:

1. **Model complexity:** In spite of dramatic advances in our ability to measure mRNA and protein levels in cells, nearly all biological systems are underdetermined with respect to the problem of *TRN* inference. Also, learning complete regulatory networks is computationally demanding and is therefore constrained by the efficiency of our algorithms. Hence, the methods made for inferring and modeling of regulatory networks must strike a balance with regard to model complexity; a model must be both sufficiently complex to describe the system accurately and sufficiently constrained to allow for learning parsimonious – a simple model is less likely to overfit the model to the training set – models from limited data in reasonable time.
2. **Statistical soundness:** When inferring a regulatory network, decisions must be made in an astronomically large space of possible regulator–target interactions. It is clear that some of these interactions will have more supporting evidence than other interactions. In many cases previously developed statistical methods give us a solid framework for model selection, model constraint, and model parameterization, as well as the estimation of model confidence and expected predictive performance. In any case, a rigorous statistical method is required to approach this problem.
3. **Integration of equilibrium and kinetic data sets:** Combining both types of experiment in one data set results in a more complete representation of the system, which enables much of the regulatory network to be learned (*see Section 4* for further discussion).
4. **Allowance of combinatorial interactions between predictors (**Note 15**):** To describe the complexity underlying many biologically solved *TRNs*, a good solution must allow

combinatorial control of gene expression by *TFs*. Although, allowing such interactions increases the complexity of the model, and possibly the chance of overfitting, it is required to model a biologically relevant *TRN* (see **Section 4** for further discussion).

5. Reducing the dimensionality of the *TRN* inference problem: The amount of data that is required in order to learn regulatory networks for a whole organism is immense and hampers *TRN* inference. Any method that tries to learn a whole organism *TRN* needs to find ways to reduce the dimensionality of the problem in a way that conserves the majority of the information inherent to the system.
6. Predictive power and clear route to validation: There is a distinction made between building a model that describes the data in our training set and building a model that is also able to describe not yet known results (true predictive power). To build a predictive model, one needs to first recognize several simple truths: (1) we do not have enough data to fully describe the physiology of even the simplest of organisms; (2) our methods for learning regulatory networks are limited by our desire to strike peace between model complexity and our ability to model such complexity; hence our methods are not perfect, each has its own set of pros and cons; and (3) we need to strike a balance to avoid overfitting the data.
7. Modeling feedback loops: Feedback loops are ubiquitous in regulatory networks and have central roles enhancing, buffering, and filtering changes that occur within a global regulatory system. Negative feedback in many cases is used to stabilize the system (e.g., a thermostat) and can play a key role in homeostasis. Positive feedback can be used to enhance a certain stimulus and thus fix a transition from one stable cell state to another. Feedback loops represent multiple important regulatory motifs and should be modeled properly if we are to capture dynamical aspects of global control in our models. Modeling and learning of feedback loops, however, presents several difficult challenges (alas, several popular methods are unable to learn loops of any kind). Therefore, the ability of a method to learn feedback loops will enable the resultant model to recapitulate/predict properties of real biological *TRNs*, namely stability, robustness, and responsiveness, as discussed in **Section 3**.

Taken together, it seems that there is no single good solution that fits all the requirements currently in existence; one therefore needs to decide what properties are more important to model, given the need of specific projects/problems, and choose the methods employed to accommodate these requirements accordingly.

6. Examples of Methods

6.1. A Glance at the Myriad Methods for Modeling of TRN

All methods for *TRN* inference must face similar challenges as described in **Section 4**, and no one method can fit all requirements given the diversity and inadequacies of data types. Each method has its own set of strengths and weaknesses, and the usefulness of the method is dependent on the specifications of the problem at hand. In general, methods that try to capture the system's state in detail (e.g., ordinary differential equations) need a large amount of data, or very well-studied sub-circuits, in order to produce meaningful results, and are therefore limited to small-scale, well-studied cases. Primarily, these methods make valid contributions to the modeling of already determined circuits (as opposed to the learning of global regulatory networks), while methods that simplify the states of the system, e.g., Boolean networks, perform better when larger data sets, which are incomplete in respect to the full physiological states of the system, or high-error measurements are used. In learning from a large data set with thousands of genes and only dozens to hundreds (to possibly thousands) of conditions, it is important to incorporate some measure of significance and confidence to the results; thus approaches that have a strong statistical foundation are potentially preferable.

6.2. Ordinary Differential Equations Formalism for TRNs

Ordinary differential equations (ODEs) have been widely used in science to model dynamical systems. For the case of gene regulation, ODEs model the concentration of mRNAs $\mathbf{Y} = \{y_1, \dots, y_n\}$, using a rate equation that measures the production rate of mRNA y_i as a function of the concentration of other components $\mathbf{X} = \{x_1, x_2, \dots, x_n\}$ of the system (e.g., activating and repressing *TFs*). The mathematical form of the rate equation is

$$\frac{\partial y_i}{\partial t} = f_i(X),$$

where $\frac{\partial y_i}{\partial t}$ is the production rate of mRNA y_i , and $f_i(X)$ is a function of the system's components that affect the synthesis or degradation rate of mRNA y_i . Notice that for each y_i the chosen function $f_i(X)$ is different and is composed of the specific effectors of gene's i transcript level. This formalism is key to modeling of *TRNs* and we will encounter ODEs again in **Section 6.4**, where we will discuss the *Inferelator (I)*, an algorithm for learning *TRNs* directly from systems biology data sets.

6.3. Introduction to Bayesian Networks and their Possible use for TRN Inference

A **Bayesian network** (**Note 16**) is an efficient and intuitive representation of the joint probability distribution of a set of variables. Bayesian network approaches generally model the structure of regulatory networks (e.g., transcriptional regulatory network) as a *Directed Acyclic Graph* (*DAG*) (**Note 17**), where the nodes of the graph represent the variables and the edges represent conditional relationships between the variables. Associated with each node (variable) in the graph is a set of conditional probabilities describing the possible states of the variable, given its parents. By definition each node in a Bayesian network is independent of its non-descendants. In other words, variables in the network can only have conditional relationships with those variables that are their parents.

Let us begin with a simple example of a Bayesian network, after which we will briefly explain how these can be extended to represent *TRNs*. Consider a case where a physician needs to decide whether a patient has “Diabetes Mellitus” (DM). As our physician is a bit of a statistics buff, he chooses to use a Bayesian network that will help him evaluate the probability that a patient of his has diabetes. As he is also an expert in diabetes, he knows a number of factors that can help indicate whether or not a patient has diabetes, such as “high blood glucose” (HBG) and “urinating frequently” (UF). In addition, because some of these factors can also indicate “kidney disease” (KD), he also includes this as a possible explanation. Moreover, as none of these factors will positively determine whether or not a patient has diabetes, he assigns a probability for each patient having (or not having) diabetes given each possible combination of the factors. Finally, as both diabetes and kidney disease are present in only a certain proportion of the population, he assigns probabilities to represent what he believes to be their relative frequency within the population. **Figure 9.1** would be an example of a Bayesian network created by the physician.

Inherent to the structure of any Bayesian network is the rule that each variable is independent of its non-descendants, given its parents. For example, in **Fig. 9.1**, knowing the value of GU is necessary and sufficient to determine the probability distribution of UF (if GU is true, then the probability of UF is 0.7). The structure of this Bayesian network also contains a set of conditional independence statements (Using $I(A;B|C)$ to note that A is independent of B, given its parent C): $I(DM; KD)$, $I(HBG; KD, GU, FU | DM)$, $I(GU; HBG | DM, KD)$, and $I(FU; HBG, DM, KD | DU)$. Given these set of independence statements, the joint distribution for the network is simply: $P(DM, KD, HBG, GU, FU) = P(DM) * P(KD) * P(HBG | DM) * P(GU | DM, KD) * P(FU | DU)$ (30, 31).

We can now use the network to resolve diagnosis, for example, Does a patient have diabetes given that he has high glucose in his blood but he does not frequently urinate? What if he

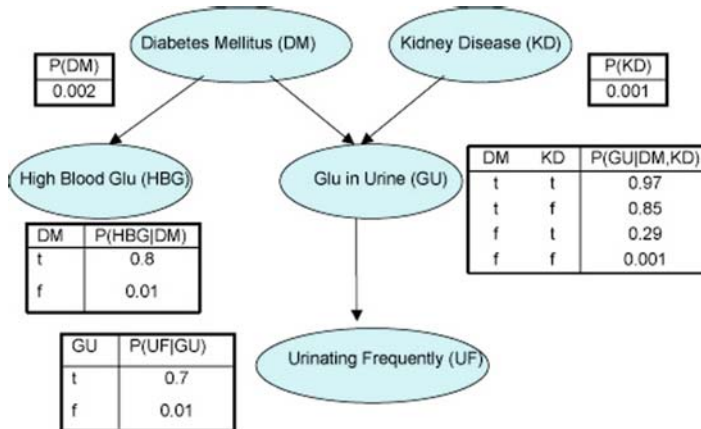


Fig. 9.1. **An example of a simplified Bayesian network.** In the figure each node (balloon) represents a variable and each edge corresponds to a conditional dependency between two variables. A table is attached to each variable, the left column shows the whole combination of states the parents of a variable can have, while the right column holds the probability of the variable being true for any combination of parental states.

frequently urinates but his blood sugar level is normal? The conditional probabilities for these cases are $P(DM | HBG, \text{not } UF)$ and $P(DM | \text{not } HBG, UF)$, respectively. Typically, we will have prior knowledge on the state of some of the variables (e.g., UF and HBG), and the evidence will iteratively change the conditional probabilities of conditionally dependent nodes – knowing that someone has high glucose in the blood will increase the chances of the variable DM being true. The crucial principle that makes our Bayesian network so useful is the rule that **each variable is independent of its non-descendants, given its parents**, resulting in a set of independencies between the variables, which then simplify the computation of the joint probability and allow us to query even complex joint probability distributions via simple operations on the Bayesian network.

Bayesian networks are particularly useful when trying to model a situation in which causality plays a role, but we have incomplete data (these methods are robust to missing data). In such a case, a probabilistic description of the possible states is naturally used. Moreover, *TRNs* are sparse, i.e., a gene is usually regulated by a small number of *TFs* that directly affect its transcription, and Bayesian networks are particularly suited for learning under such conditions. However, Bayesian networks are generally directed acyclic graphs (*DAGs*) and thus are unable to model feedback loops, which are a major control mechanism in all known *TRNs*. To extend the use of Bayesian networks for learning of *TRNs*, let us briefly review the main concepts of the algorithm developed by Friedman et al (31). The structure of a genetic regulatory system is represented as a *DAG*:

$$G = \langle V, E \rangle.$$

The vertices $i \in V, 1 \leq i \leq n$, represent the transcript level of the gene that correspond to random variables $\mathbf{X} = \{x_1, x_2, \dots, x_i, \dots, x_{10}\}$, where x_i is the expression level of the i th gene. Given a set of independent values of \mathbf{X} , the algorithm learns a network structure that fits the data based on a scoring function. More precisely, the algorithm does not learn one particular network that have the highest score, but focuses on features that are common to multiple high-scoring networks. Particularly, the algorithm searches for Markov relation and order relations between pairs of variables x_i and x_j . Markov relation occurs when x_i belongs to the minimal set of variables that shields x_j from the rest of the variables – i.e., the minimal set of a variable’s parents that appear in all of the high-scoring networks, while an order relation happens when x_i is the parent of x_j in all of the graphs in an equivalence class (**Note 18**). Knowing the Markov and order relations should be considered as an indication of causality in the network, and added with biological prior knowledge, can suggest the relationships between transcription factors and their putative targets. For a more detailed discussion of the algorithm and Bayesian networks, we refer the reader to (30–32).

6.4. The Inferelator: A Walk Through Example of An Algorithm for Learning Regulatory Networks from Systems Biology Data Sets

The *Inferelator* was built with the purpose of learning *TRNs* from systems biology data sets in a solely data-driven manner. The method is intended to be paired with an integrative-biclustering method, such as cMonkey (12). We will begin by reviewing the end result of running the *Inferelator* algorithm on a comprehensive data set of 268 microarray experiments (each derived from 16 biological and technical replicates) done for the archaeon (**Note 19**) *Halobacterium NRC-1*, followed by a step-by-step tutorial of the *Inferelator* algorithm and model formalism. At this point we believe, the reader should read Notes 24 and 25, and only then step into the *Inferelator* tutorial.

Figure 9.2a illustrates:

1. The network is learned globally with biclusters representing regulons as the basic units of response (rectangular nodes) and *TFs* and environmental factors (e.g., oxygen level) as predictors (circular nodes).
2. Edges are weighted based on a given predictor’s influence on a given bicluster.
3. Combinatorial control is commonplace in this learned network (interactions between *TFs* are shown as triangular nodes), graphically suggesting that this part of the *Inferelator* procedure is important to predictive performance.

The algorithm recovered known and novel regulatory interactions and several of the novel regulatory interactions have been subsequently verified experimentally. **Figure 9.2b**, for example, shows a novel

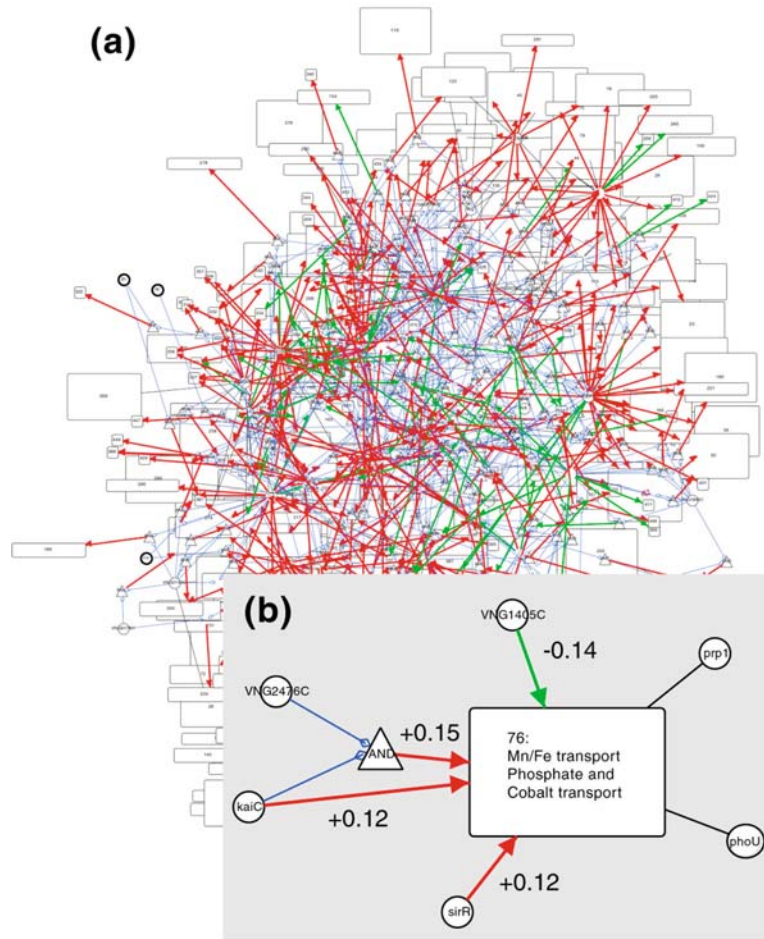


Fig. 9.2. **The inferred regulatory network of *Halobacterium* MRC-1.** The inferred regulatory network of *Halobacterium* MRC-1, visualized using Cytoscape (41) and Gaggie (42). (a) The full inferred regulatory network. (b) Example regulation of Bicluster 76. The *Inferelator* selected sirR, kaiC, VNG1405C, and VNG2476C as the most likely regulators of the genes in bicluster 76 from the set of all (82) candidate regulators. The relative weights, β , by which the regulators are predicted to combine to determine the level of expression of the genes of bicluster 76, are indicated alongside each regulation edge. The TFs VNG2476C and kaiC combine in a logical AND relationship. phoU and prp1 are the TFs belonging to bicluster 76. Taken from (1).

Key Figure 9.2:

- Bicluster – A subset of genes under a subset of experimental conditions, grouped together based on correlation in transcription pattern and other data types (see Note 24 for more details on the biclustering algorithm used in this example).
- Predictor – Transcription factors expression levels and the levels of external stimuli are treated as predictors.
- △ Interactions – The *Inferelator* allows for some combinatorial regulation in which two predictors can function together to bring about a response. Combinatorial logic such as AND OR XOR is allowed.
- \\ \\ \\ Edges – There are three types of edges in the regulatory network:
 1. *arrow head* – represents repression.
 2. *diamond head* – represents interactions (AND logic).
 3. *straight line* – represents TFs that belong to the bicluster (these TFs have a high correlation with the bicluster, but we cannot determine if they regulate it).

The thickness of the edges corresponds to the strength of the regulation as predicted by the *Inferelator* (the β parameter values, explained in Model formulation).

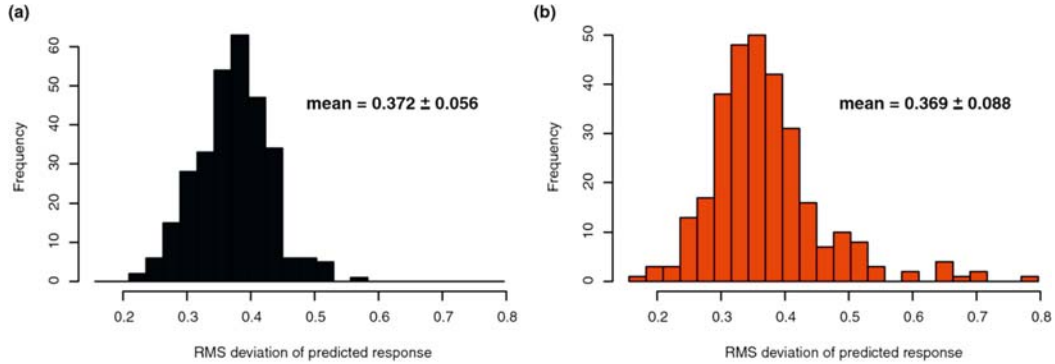


Fig. 9.3. **Predictive performance of the predicted TRN.** (a) The root mean square deviation (RMSD) error of predicted response in comparison with the true response for the 3,00 predicted biclusters evaluated over the 268 conditions of the training set. (b) The RMSD error of the same 300 biclusters evaluated on new data (24 conditions) collected after model fitting/network constructions (1).

regulatory interaction (SirR control of metal transport in *Halobacterium*) that has been validated by a combination ChIP-chip, knock out experiments, and subsequent microarray experiments (33).

Figure 9.3 illustrates the predictive performance of the inferred TRN.

Following is the tutorial on the *Inferelator* algorithm, for a more detailed discussion of the novel and known regulatory networks that were learned for *Halobacterium* NRC-1 by the *Inferelator* we refer the reader to the *Inferelator paper* (1).

6.4.1. *Inferelator* Tutorial and Formalism

A simple way to model gene transcription is using an ODE to represent the rate of production/degradation of each component of the system (in this case mRNA levels of genes and/or biclusters) as a function of the concentrations of other components, the ODE has the mathematical form:

$$\frac{\partial y_i}{\partial t} = ky_i + \gamma y_i + g(X). \quad [1]$$

Here, $X = \{x_1, \dots, x_i, \dots, x_n\}$ is the vector of all the regulatory factors, and the coefficients $k > 0$, $\gamma < 0$ are the production and degradation coefficients of gene y_i , respectively. The function $g(X)$ represents the combined effect of all the regulators on the transcript levels of gene y_i . To simplify the model, we assume that k and γ have similar values or that β (introduced shortly) can account for any difference in k and γ . Also, we name y_i – the gene of interest – simply y . The ODE can now be represented as

$$\tau \frac{\partial y}{\partial t} = -y + g(\beta \bullet X), \quad [2]$$

where τ is the time constant that represents the level of gene y in the absence of external determinants, $\hat{\beta} = (\beta_1, \beta_2, \dots, \beta_p)$ are the regression coefficients (see Notes for more details on how we choose $\hat{\beta}$) that indicate which of the possible covariates (regulatory factors) is more influential on the response, and $\beta \bullet X = \sum \beta_j x_j$ is simply the weighted sum of each regulator x_i multiplied by its corresponding coefficient β_i . In order to allow combinatorial control in which two regulators can work together to produce a response that is not simply the weighted sum of each of the two regulators alone – for example, consider a case where two *TFs* dimerize and only then become active – Eq. [2] is enhanced to the form of

$$\tau \frac{\partial y}{\partial t} = -y + g(\beta \bullet Z_{(X)}), \quad [3]$$

where $Z_{(x)} = (z_1[X_1], z_2[X_1], \dots, z_{12}[X_1, X_2])$ (hereafter notated as Z) is a set of functions of the regulatory factors X as before. However, single and multiple factors can be inputs to these functions depending on whether the functions represent single or combinatoric regulation, respectively. The coefficient β_j for $\{j = 1, 2, \dots, P\}$ now describes the influence of each element of Z , with positive coefficients corresponding to inducers of transcription and negative coefficients to transcriptional repressors. The simple choice of $z_j(x) = x_j$ amounts to the simple weighted linear combination of influencing factors $\beta \bullet Z = \sum \beta_j x_j$, as is used in Eq. [2]. To accommodate combinatorial logic for transcriptional control, we use a more general form for the function Z :

$$Z = \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \beta_4 x_4 + \beta_5 x_5 \\ + \beta_6 \min(x_i, x_j) + \beta_7 \min(x_i, x_j). \quad [4]$$

Equation [4] allows combinatorial control to be modeled, but limits the set of possible regulators to seven for each response variable (bicluster expression level), and is an example of balancing our desire to model the complexity of the problem with our ability to model it without overfitting. The combinatorial control portion of Eq. [4] is accounted for by the two last terms $\beta_6 \min(x_i, x_j) + \beta_7 \min(x_i, x_j)$. Here, for each bicluster/singleton (hereafter simply bicluster, a singleton is a gene that had not been included in any of the biclusters), we choose a limited set of predictors, five single predictors and two “interactions” (**Note 20**) – that best describes the expression level of the bicluster. The seven predictors are chosen prior to multivariate regression, in order to reduce the set of possible predictors which the *Inferelator* has to choose from. We perform a first round of shrinkage in which for each bicluster we choose the limited set of predictors that best describes it in the following manner: (1)

perform linear regression using only *one predictor* at a time in order to evaluate how well that predictor explained the expression level of the bicluster – the smaller the residual between the predicted and the real values of the bicluster expression level, the better the bicluster expression was explained; (2) similarly, perform linear regression for each *pair of predictors*, using the minimum of their expression level to evaluate how well the pair-wise predictors explained the bicluster; (3) rank all single and pair-wise predictors based on the residual values that measure how good the prediction was; and (4) for each bicluster in the response vector choose a subset of five single predictors and two “interaction” predictors that had the smallest residual.

Various functional forms can be adopted for function g . In the *Inferelator* method we adopt the following function, as it allows for simultaneous determination of β at several values of the shrinkage parameter t (see Notes):

$$g(\beta \bullet Z) = \begin{cases} \beta \bullet Z : \min(y) < \beta \bullet Z < \max(y) \\ \max(y) : \beta \bullet Z > \max(y) \\ \min(y) : \beta \bullet Z < \min(y) \end{cases} \quad [5]$$

The simplified kinetic description of **Eq. [3]** encompasses essential elements to describe gene transcription, such as control by specific *TFs*, activation kinetics, and transcript decay, while at the same time facilitating access to computationally efficient methods for searching among a combinatorial large number of possible regulators. The reason to put a constraint on the value of $g(\beta \bullet Z)$ is to limit maximum y value and minimum y value to realistic levels (look at **Eq. [3]**); y is not likely to have a larger amount of transcript than the maximum value taken from all the microarray experiments, thus the value of $g(\beta \bullet Z)$ should be constrained as described in **Eq. [5]**.

As discussed in **Section 4**, integration of kinetic and equilibrium (**Note 21**) datasets enables a better representation of the system and allows more of the *TRN* to be learned. In order to use the data from both classes, we need to combine both types of measurements into a single expression on which we learn the model parameters β and τ .

For a steady-state measurement, $\frac{dy}{dt} = 0$, **Eq. [3]** reduces to:

$$y = g(\beta \bullet Z). \quad [6]$$

For time-series measurements, taken at times $\mathbf{t} = (t_1, t_2, \dots, t_T)$, **Eq. [3]** may be approximated as follows:

$$\tau \frac{y_{m+1} - y_m}{\Delta t_m} + y_m = g\left(\sum_{j=1}^P \beta_j \cdot z_{mj}\right) \quad \text{for } m = 1, 2, \dots, T - 1, \quad [7]$$

where $\Delta t_m = t_{m+1} - t_m$ is the time interval between consecutive measurements, and y_m and z_{mj} are, respectively, the measured values of y and z_j at time t_m . With this formulation we place no requirements on the regularity of Δt_m , and can readily use data with differing time intervals between measurements (**Note 22**).

As a result, the right-hand side of both **Eqs. [6]** and **[7]** are identical – the algebraic interpretation of a scalar product is shown in **Eq. [7]** – allowing for simultaneous model fitting using equilibrium and time-series data (**Section 4** – challenge three met). Taken together all steady-state measurements and time course measurements, the left-hand sides of **Eqs. [6]** and **[7]** can be combined into a single response vector (Y), allowing β to be fit with one of the many available methodologies for multivariate regression (see Notes for discussion of LASSO).

There are two parameter types that we have to determine in our *TRN*: the set of τ s and the set of β s for each predictor surviving the first round of shrinkage. The set of β s are being fit using multivariate regression, however, both β and τ need to be determined together. To optimize tau and beta we use an iterative approach.

Beginning with an initial guess for τ , **first** we find the regression solution for β using the multivariate regression methods of LASSO; **second**, we solve for a new τ that minimizes the prediction error $S(\hat{\beta})$ between the prediction vector $\hat{\mu}$ and the training set of known bicluster expression values y .

$$S(\hat{\beta}) = \|y - \hat{\mu}\|^2 = \sum_{i=1}^n (y_i - \hat{\mu}_i)^2. \quad [8]$$

Subject to the constraint,

$$\sum_{j=1}^m |\hat{\beta}_j| \leq t|\beta_{OLS}|. \quad [9]$$

Third, we repeat the first two steps until convergence.

$|\beta_{OLS}|$ is the Ordinary Least Squares (OLS) estimate of β , and is the maximum norm β can have. Therefore, t , the shrinkage parameter can range from 0 to 1. The limit $t = 0$ amounts to selection of the null model ($y = |y|$), meaning we do not have data to learn any predictor. In the limit $t = 1$ we have the OLS estimate for β . In order to determine the optimal value for the shrinkage parameter, we minimize the prediction error, which is estimated using tenfold cross-validation (**Note 23**), as shown in **Fig. 9.4**.

For each value of the shrinkage parameter we then measure the error estimate under the ten leave-out cases:

$$\text{Err}_{CV} = \frac{\sum_{i=1}^n (y_i - \hat{\mu}_i)^2}{n}. \quad [10]$$

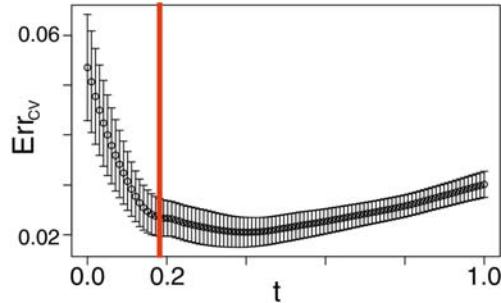


Fig. 9.4. **Selection of model using Cross-Validation (CV)**. The ordinate represents an estimate of prediction error (Err_{CV}) from tenfold CV (the mean of the error in the ten leave-out samples used is the CV error estimate). The shrinkage parameter t allows us to select subsets of predictors continuously. We evaluate our fitted model for a range of values of t (with $t = 0$ [the null model] and $t = 1$ [the ordinary least squares solution]). The error bars denote the standard error of Err_{CV} (the standard deviation of the ten leave-out samples' error estimates). The *red line* shows the value of t selected for our final model for this cluster – the most parsimonious model within one standard error of the minimum on the Err_{CV} versus t curve. Figure taken from (1).

Thereafter, the error estimate is taken to be the mean over each of the ten leave-out cases, which also allows us to measure the standard deviation of the ten individual leave-out error estimates (the error bars in **Fig. 9.4**). Remember that the smaller the values of t we choose, the more parsimonious the model is.

We find the minimum on the CV error curve, move up on the error bar for that minimum point, and then take a horizontal step toward $t = 0$ until we hit the error curve again – this is the value we choose for t – resulting in a parsimonious estimate of the shrinkage parameter t . In this manner a separate value of the shrinkage parameter t is chosen for each bicluster we attempt to model. The result is a predictive, dynamic function for each bicluster for which the method did not select the null hypothesis $t = 0$. Moreover, the values of β capture information about the influence each covariate has on the bicluster. Importantly, all data used in the procedure are normalized to have variance of one prior to network inference. Thus, for a given influence on a given bicluster, we can uniformly interpret the magnitude of β and use this value to rank the individual interactions by significance. Taken together, the set of β s and τ s for all the biclusters/genes constitute the full model for the *TRN*. Following is the outline of the method:

6.4.2. Inferelator Summary

```

For each (bicluster k) {
  For each (TF or environmental factor i){
    Update list of best single influences
    For each (TFj){
      Update list of best
      interactions list (min[ i, j] )
    }
  }
}

```

```

    }
  }
}
Select from predictors and estimate model
parameters ( $\beta$  and  $\tau$ ) using LASSO/cross
validation.
Store model for gene/bicluster k
Combine models for individual biclusters into
global network
Process network for viewing (beyond the scope
of this chapter)

```

7. Coda

Inferring regulatory networks has become increasingly more feasible with the advent of genomics and systems biology. Commonly used whole-genome techniques (e.g., microarrays), however, often suffer from large amounts of random and systematic error and high dimensionality that makes learning non-trivial. A strong, statistically based model has to satisfy three main requirements: (1) the model should be able to capture the relationships between the variables; (2) the data used for learning should have a sufficient number of replicates, sufficient sampling with respect to the complexity of the organism, and correct experimental design (including time series); and (3) the learning process should be constrained by additional data types besides expression data. Here, we described in detail one method that can infer *TRNs*, the *Inferelator (1)*, from systems biology data and from association networks. Our long-term goal is to develop methods capable of distilling physical and dynamical network models from sets of error-prone or indirect biological associations; attempting to directly learn these physical underpinnings will increase the accuracy and interpretability of the resulting validated regulatory and interaction networks. *As a field we have recently shown that, for prokaryotic systems, we can learn predictive dynamical models from sequence, expression, and chip-chip data.*

8. Notes



1. Operon: a genetic unit or cluster that consists of one or more genes that are transcribed as a unit and are expressed in a coordinated manner. This work awarded Jacob and Monod the 1965 Nobel Prize in Physiology and Medicine.

2. Normalization: the process by which microarray spot intensities are adjusted to take into account the variability across different experiments and platforms.
3. Transformation: the application of a specific mathematical function so that data are changed into a different form. The most common data transformation in microarray studies is \log_2 .
4. Immunoprecipitation: the process of precipitating a protein/protein complex from a complex mixture, using a specific antibody against a protein of interest.
5. False positive (also known as type 1 error): The rejection of a true null hypothesis, e.g., declaring that a *TF* binds a DNA sequence, when it, in fact, does not.
6. False negative: the acceptance of a false null hypothesis, e.g., declaring a *TF* does not bind a DNA sequence, when it, in fact, does.
7. Biological replicate: mRNA from different biological cases (a case is the biological unit under study – one mouse, one batch of cells, etc.) treated under the same experimental conditions is taken.
8. The number of biological replicates required is still the process of investigation.
9. The actual signal that induces lactose catabolism is allolactose, which is a catabolic intermediate of lactose.
10. Basal level of β -galactosidase is essential, as the induction of this system is done by allolactose – a catabolic intermediate – and allolactose will be present only after β -galactosidase degraded lactose. The basal amount is very small and after induction boosts up to 500 times the basal amount.
11. It is however possible that other factors may regulate the gene network, for example, other proteins such as kinases, some of which may relay the environmental affect to a molecular response.
12. Equilibrium is an assumption made by the researcher that under the experimental design the system is stable. In practice “equilibrium” or “steady state” data is simply data for which we attempt to allow the system to reach equilibrium and do not record/observe multiple time points.
13. Kinetic measurement (time series) measures the change in the system’s expression pattern from the time the stimuli (e.g., shift to lactose as carbon source) were administered until the system has established a new equilibrium point under the new constraint.
14. In addition to the contributions these works make to basic science/biology, these works will soon (in fact are already beginning to) make direct impact on a number of

economic/industrial activities, as *TRN* inference will allow for more rational engineering of bacteria (and eventually more complex systems as well). Knowing the *TRNs* of several organisms will inevitably aid both industrial and pharmaceutical biosynthetic activities as well as the efforts to use prokaryotes as platforms for bioremediation. Resolving the *TRNs* of these industry workhorse bacteria will have immense implications on the biotechnological industry.

15. Predictor: A predictor is either an experimental condition or a transcription factor. We use the same definition for a predictor as the one used in the next section describing the *Inferelator*.
16. Also commonly named a “belief network”.
17. *DAG* is a directed graph in which there is no path starting at node (i) and returning to the same node.
18. A Bayesian network structure G implies a set of independence assumptions (as explained in **Fig. 9.1**). However, more than one graph can imply exactly the same set of independencies. For example, consider graphs over two variables X and Y . The graphs $X \rightarrow Y$ and $X \leftarrow Y$ both imply the same set of independencies.
19. Many books and articles still refer to them as “Archaeobacteria;” this term has been abandoned because biochemically and genetically they are as different from bacteria as eukaryotes.
20. By interactions we refer to *TF–TF* interactions, such as dimerization, and to *TF–environmental-factor* interactions. Both kinds of interactions are allowed, as both of these factors are included in the set of predictors. To encode interactions the minimum expression value of the two predictors is taken as the expression value for the interaction, because it is the limiting factor (you cannot have more dimers than that value).
21. The experimental conditions are classified either as belonging to a steady-state experiment or a time-series experiment. In some cases, we refer to conditions as “equilibrium” or “steady state” measurements out of convenience, but cannot know whether the system, in any strict sense, is at equilibrium.
22. Note that if the interval was longer than the time scales in which possible regulatory interaction occur, those interactions will be missed.
23. A method of estimating the accuracy of a regression model. The data set is divided into several parts, with each part in turn used to test a model that was fitted to the remaining parts. The samples are partitioned into subsamples, an analysis similar to the one used on whole the sample is used on several of

the subsamples, while further sub samples are retained “blind” for subsequent use in confirming and validating our initial analysis.

24. *Biclustering as a first step to TRN inference*

Coverage: To learn a *TRN*, the data set we use should describe the behavior of the system under as many possible experimental conditions and cell states. Primarily, this means that the data set we use should be comprehensive, containing possibly hundreds to thousands of different conditions; one must also take care that the set of experiments used span different distinct cell states.

Motivation for biclustering: As we approach the data sizes needed to solve the *TRN* inference problem, a new problem emerges: as the data set grows bigger, it becomes less probable that genes will have trivial co-expression patterns across all observations. Thus, it is less likely to find clusters – i.e., groups of genes with correlated expression pattern over all the experimental conditions composing our data set. Rather, it is more likely to find biclusters – i.e., groups of genes with correlated expression pattern over a subset of the experimental conditions composing our data set – because genes are often under the control of different set of regulators when exposed to different environmental conditions. In general, there is no reason to believe that co-regulated genes will have a correlated expression pattern across all conditions in any large compendium of experiments.

Co-expression vs. co-regulation: Co-regulated genes are genes that share similar DNA sequences (i.e., *TFs* binding motifs) in their regulatory regions, which make them the target of a similar set of *TFs*. As assumed by several clustering algorithms, the genes often recognized as co-regulated are usually co-expressed. Conversely, the assumption that co-expression equals co-regulation is incorrect (34). Co-regulated genes are also often functionally (physically, spatially, genetically, and/or evolutionary) associated, and such a priori known associations can provide support for appropriately grouping genes into co-regulated groups.

Biclustering of co-regulated genes as a first step prior to TRN inference: Without biclustering, the inference algorithm has to exhaustively check which possible *TFs* control each and every gene, and also has to resolve exponentially larger numbers of causal symmetries (such as activators of activators). However, by biclustering genes, prior to inference, the *TRN* inference problem

conveniently becomes which *TFs* regulate each bicluster, rather than which *TFs* regulate each gene. Moreover, biclustering genes into co-regulated groups serves to reduce the amount of possible *TFs* to putative target bicluster interactions – and hence make the problem of learning a whole genome *TRN* realistic.

Integrative biclustering methods: A biclustering algorithm that combines microarray data with other data types (such as interaction networks and motif discovery) is termed “integrative biclustering” algorithm, and is more efficient in learning true co-regulated groups of genes, in comparison to learning merely co-expressed (that may or may not be co-regulated) groups obtained using only microarray data. An integrative biclustering algorithm will thus reduce dramatically the number of unresolvable situations, when inferring *TRNs*, caused by “false biclusters” – i.e., biclusters that although co-expressed are not co-regulated. Importantly, *TFs* binding motifs shared between genes can aid in finding true co-regulated groups. However, a large number of these motifs in the regulatory regions are still not well defined, hence integrating the de novo detection of regulatory binding motifs into the biclustering algorithm is essential even for the most well-studied organisms. Biclustering methods that use de novo motif detection – i.e., checking if co-expressed genes also share common DNA sequences in their regulatory regions – to constrain the bicluster procedure produce better results than clustering when one is interested in finding novel regulatory interactions.

cMonkey – an integrated biclustering algorithm designed to group genes based on co-regulation: Here we review an integrated biclustering algorithm, *cMonkey* (12), that groups genes and conditions into biclusters on the basis of (1) coherence in expression data across subsets of experimental conditions, (2) co-occurrence of putative *cis*-acting regulatory motifs in the regulatory regions of bicluster members, and (3) the presence of highly connected subgraphs in metabolic (35) and functional association networks (36, 37). Because *cMonkey* was designed with the goal of identifying putatively *co-regulated* gene groupings, we use it for “pre-clustering” genes prior to learning *TRNs*. Importantly, for following *TRN* inference, *cMonkey* identifies relevant conditions in which the genes within a given bicluster are co-regulated, and the inferred regulatory influences on the genes in each bicluster pertain to (and are fit using) only those conditions within each bicluster.

25. *Model selection in the context of multiple linear regression using the Lasso: Lasso, or L1-shrinkage* (38), is a method for estimating the relative value of multiple predictors and ideally selecting the most parsimonious subset of predictors with maximum predictive power. We describe the application of model shrinkage to a simple additive-model here, but these concepts can be extended to several model types. The variable of interest, \mathbf{Y} , is often called the “response variable” and is linearly dependent on the values of two or more “predictors” or “explanatory variables”, \mathbf{X} . The *Inferelator* uses transcription factors and environmental factors affecting mRNA expression as predictors. In **Section 6.4**, the *Inferelator* describes each bicluster/gene expression level as a weighted sum of the levels of its most likely predictors. Let us consider a hypothetical example that will make the concepts underlying multiple linear regression much more clear. **Table 9.1** shows the summary of ten baseline variables ($m = 10$), age, sex, and eight serum measurements that were obtained for each of $n = 450$ patients with high cholesterol and patients with normal cholesterol levels.

The goal is to construct a model that predicts response, $\mathbf{y} = \{y_1, y_2, \dots, y_n\}$ (cholesterol level), from covariates, $\mathbf{X} = \{x_1, x_2, \dots, x_m\}$ (age, sex, and eight serum measurements). Because the scales of each variable and the responses are different, we use scale transformation to make the covariates standardized with mean of zero and unit length and the response with mean of zero.

Table 9.1
Patients measurements of cholesterol level together with ten possible explanatory variables (modified from (38))

Patient	Age		Sex		Serumm Measurements						Response
	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	y
1	50	1	123	33.6	102	94.1	70	15.3	4.5	86	182
2	45	2	171	44.2	88	105.2	38	13.4	3.8	69	215
3	73	1	192	51.3	91	131.3	44	18.2	4.7	71	195
4	38	1	133	61.0	96	97.3	41	17.3	3.6	86	98
•	•	•	•	•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•	•	•	•	•
449	48	2	98	41.2	86	122.4	47	12.7	5.1	72	260
450	61	1	115	51.4	102	111.1	94	18.1	4.4	66	120

$$\sum_{i=1}^n y_i = 0, \sum_{i=1}^n x_{ij} = 0 \text{ and } \sum_{i=1}^n x_{ij}^2 = 1 \text{ for } j = 1, 2, \dots, m.$$

Again, our goal here is to predict the response based on the covariates. For illustration purposes we choose a simple linear additive model here:

$$\hat{\mu} = \sum_{j=1}^m x_j \hat{\beta}_j = X \hat{\beta}_j [X_{n \times m} = (x_1, x_2, \dots, x_m)],$$

where $\hat{\beta} = (\beta_1, \beta_2, \dots, \beta_m)$ are the regression coefficients – i.e., $\hat{\beta}$ represent which of the possible covariates is more influential on the response, and $\hat{\mu}$ is the prediction vector – i.e., our estimation of y given \mathbf{X} and $\hat{\beta}$.

We want to choose $\hat{\beta}$ values that will make our estimation, $\hat{\mu}$, of cholesterol level similar to the real measured values y .

Therefore, we search for the set $\hat{\beta}$ that will minimize the total squared error $S(\hat{\beta})$:

$$S(\hat{\beta}) = \|y - \hat{\mu}\|^2 = \sum_{i=1}^n (y_i - \hat{\mu}_i)^2.$$

The set $\hat{\beta}$ that minimizes the prediction vector is named the Ordinary Least Squares (OLS) solution $\hat{\beta}_{OLS}$.

However, minimizing $S(\hat{\beta})$ should be constrained by our desire to keep the model parsimonious. For example, it might be that three of the covariates (say: age, lipids in blood and blood glucose) are enough to give us a very good estimate of y . This estimate would never be better than the estimate we get by incorporating all of the covariates in our model, but a parsimonious model is easier to interpret and is less likely to overfit the model to the training set, hence loosing predictive performance of the model.

Therefore, let $T(\hat{\beta})$ be the absolute norm of $\hat{\beta}$, and k be the number of covariates the LASSO chooses to incorporate into our model:

$$T(\hat{\beta}) = \sum_{j=1}^k |\hat{\beta}_j|. \quad 1 \leq k \leq m,$$

$$\sum_{j=1}^k |\hat{\beta}_j| \leq t |\beta_{OLS}|. \quad 0 \leq t \leq 1,$$

where t ranges from 0 to 1 – i.e., when $t = 0$ no covariates are used in the model, and when $t = 1$ all the covariates are used and we get $|\beta_{OLS}|$ the OLS solution.

The Lasso algorithm chooses a subset of the set of $\hat{\beta}$ by minimizing $S(\hat{\beta})$ subject to bound t on $T(\hat{\beta})$:

$$\text{Minimize } S(\hat{\beta}) \text{ subject to } T(\hat{\beta}) \leq t |\beta_{OLS}|.$$

Hence:

$$0 \leq T(\hat{\beta}) \leq \sum_{j=1}^m |\hat{\beta}_j|_{OLS}$$

The so-called shrinkage parameter t essentially controls/constrains how many covariates we use in our linear model, and the parsimony property should be taken into account when choosing t . In the cholesterol example, it might be that not all of the covariates we have measured are indicative of the response, and in a case like this we would like to pick out only the covariates that are potent predictors. By constraining t we can limit the amount of covariates incorporated in our model. Choosing $t = 1$ results in the OLS solution to the problem and in most cases also causes overfitting. It is therefore important to choose a value of t carefully. In the *Inferelator* we chose the value of t using cross-validation (CV) (39) (Fig. 9.4 and discussion at Section 6.4).

An advantage of this method is that it draws upon well-developed techniques from the field of statistical learning for choosing among several possible models and for efficiently fitting the parameters of those models. For more detailed discussion we refer the reader to (38, 40).

References

1. Bonneau R, Reiss DJ, Shannon P, et al. The Inferelator: An algorithm for learning parsimonious regulatory networks from systems-biology data sets de novo. *Genome Biol* 2006, 7:R36.
2. Jacob F, Monod J, Sanchez C, Perrin D. Operon: A group of genes with the expression coordinated by an operator. *C R Hebd Seances Acad Sci* 1960, 250:1727–29.
3. Davidson EH. *Gene Activity in Early Development*. San Diego: Academic Press, 1977.
4. Samanta MP, Tongprasit W, Istrail S, et al. The transcriptome of the sea urchin embryo. *Science* 2006, 314:960–62.
5. Dynlacht BD. Regulation of transcription by proteins that control the cell cycle. *Nature* 1997, 389:149–52.
6. Cheng Y, Church GM. Biclustering of expression data. *Proc Int Conf Intell Syst Mol Biol* 2000, 8:93–103.
7. Kluger Y, Basri R, Chang JT, Gerstein M. Spectral biclustering of microarray data: Coclustering genes and conditions. *Genome Res* 2003, 13:703–16.
8. Sheng Q, Moreau Y, De Moor B. Biclustering microarray data by Gibbs sampling. *Bioinformatics* 2003, 19(Suppl 2):II196–205.
9. Tanay A, Sharan R, Kupiec M, Shamir R. Revealing modularity and organization in the yeast molecular network by integrated analysis of highly heterogeneous genome-wide data. *Proc Natl Acad Sci USA* 2004, 101:2981–86.
10. Tanay A, Sharan R, Shamir R. Discovering statistically significant biclusters in gene expression data. *Bioinformatics* 2002, 18(Suppl 1):S136–44.
11. Liu X, Sivaganesan S, Yeung KY, Guo J, Bumgarner RE, Medvedovic M. Context-specific infinite mixtures for clustering gene expression profiles across diverse microarray dataset. *Bioinformatics* 2006, 22:1737–44.
12. Reiss DJ, Baliga NS, Bonneau R. Integrated biclustering of heterogeneous genome-wide datasets for the inference of global regulatory networks. *BMC Bioinformatics* 2006, 7:280.
13. Reinartz J, Bruyns E, Lin JZ, et al. Massively parallel signature sequencing (MPSS) as a tool for in-depth quantitative gene expression profiling in all organisms. *Brief Funct Genomic Proteomic* 2002, 1:95–104.
14. Velculescu VE, Zhang L, Vogelstein B, Kinzler KW. Serial analysis of gene expression. *Science* 1995, 270:484–87.

15. Boyer LA, Lee TI, Cole MF, et al. Core transcriptional regulatory circuitry in human embryonic stem cells. *Cell* 2005, 122:947–56.
16. Harbison CT, Gordon DB, Lee TI, et al. Transcriptional regulatory code of a eukaryotic genome. *Nature* 2004, 431:99–104.
17. Lee TI, Rinaldi NJ, Robert F, et al. Transcriptional regulatory networks in *Saccharomyces cerevisiae*. *Science* 2002, 298:799–804.
18. Eilbeck K, Lewis SE, Mungall CJ, et al. The sequence ontology: A tool for the unification of genome annotations. *Genome Biol* 2005, 6:R44.
19. Keseler IM, Collado-Vides J, Gama-Castro S, et al. EcoCyc: A comprehensive database resource for *Escherichia coli*. *Nucleic Acids Res* 2005, 33:D334–37.
20. Kanehisa M, Goto S, Kawashima S, Okuno Y, Hattori M. The KEGG resource for deciphering the genome. *Nucl Acids Res* 2004, 32:D277–80.
21. Wingender E, Chen X, Hehl R, et al. TRANSFAC: An integrated system for gene expression regulation. *Nucl Acids Res* 2000, 28:316–19.
22. Boeckmann B, Bairoch A, Apweiler R, et al. The SWISS-PROT protein knowledgebase and its supplement TrEMBL in 2003. *Nucl Acids Res* 2003, 31:365–70.
23. Vidal M, Legrain P. Yeast forward and reverse 'n'-hybrid systems. *Nucl Acids Res* 1999, 27:919–29.
24. Goodlett DR, Yi EC. Proteomics without polyacrylamide: Qualitative and quantitative uses of tandem mass spectrometry in proteome analysis. *Funct Integr Genomics* 2002, 2:138–53.
25. Gunsalus KC, Ge H, Schetter AJ, et al. Predictive models of molecular machines involved in *Caenorhabditis elegans* early embryogenesis. *Nature* 2005, 436:861–65.
26. Weston AD, Baliga NS, Bonneau R, Hood L. Systems approaches applied to the study of *Saccharomyces cerevisiae* and *Halobacterium* sp. *Cold Spring Harb Symp Quant Biol* 2003, 68:345–57.
27. Savageau MA. Design principles for elementary gene circuits: Elements, methods, and examples. *Chaos* 2001, 11:142–59.
28. Wall ME, Hlavacek WS, Savageau MA. Design of gene circuits: Lessons from bacteria. *Nat Rev Genet* 2004, 5:34–42.
29. Laub MT, McAdams HH, Feldblyum T, Fraser CM, Shapiro L. Global analysis of the genetic network controlling a bacterial cell cycle. *Science* 2000, 290:2144–48.
30. Finn V, Jensen. *Bayesian Networks and Decision Graphs*. New York: Springer-Verlag, 2001.
31. Friedman N, Linial M, Nachman I, Pe'er D. Using Bayesian networks to analyze expression data. *J Comput Biol* 2000, 7:601–20.
32. Pearl J. *Probabilistic reasoning in intelligent systems: Networks of plausible inference*. San Francisco: Morgan Kaufmann Publishers Inc., 1988.
33. Bonneau R, Facciotti MT, Reiss DJ, et al. A predictive model for transcriptional control of physiology in a free living cell. *Cell* 2007; 131:1354–65.
34. Yeung KY, Medvedovic M, Bumgarner RE. From co-expression to co-regulation: How many microarray experiments do we need? *Genome Biol* 2004, 5:R48.
35. Vert JP, Kanehisa M. Extracting active pathways from gene expression data. *Bioinformatics* 2003, 19(Suppl 2):II238–44.
36. Bowers PM, Pellegrini M, Thompson MJ, Fierro J, Yeates TO, Eisenberg D. Prolinks: A database of protein functional linkages derived from coevolution. *Genome Biol* 2004, 5:R35.
37. Mellor JC, Yanai I, Clodfelter KH, Mintseris J, DeLisi C. Predictome: A database of putative functional links between proteins. *Nucleic Acids Res* 2002, 30:306–09.
38. Efron B, Hastie T, Johnstone I, Tibshirani R. Least angle regression. *Ann Statist* 2004, 32:407–99.
39. Thorsson V, Hornquist M, Siegel AF, Hood L. Reverse engineering galactose regulation in yeast through model selection. *Stat Appl Genet Mol Biol* 2005, 4:Article28.
40. Trevor H, Robert T, Jerome F. *The Elements of Statistical Learning*. New York: Springer, 2001.
41. Shannon P, Markiel A, Ozier O, et al. Cytoscape: A software environment for integrated models of biomolecular interaction networks. *Genome Res* 2003, 13:2498–504.
42. Shannon PT, Reiss DJ, Bonneau R, Baliga NS. The gaggles: An open-source software system for integrating bioinformatics software and data sources. *BMC Bioinformatics* 2006, 7:176.

Chapter 10

Inferring Molecular Interactions Pathways from eQTL Data

Imran Rashid, Jason McDermott, and Ram Samudrala

Abstract

Analysis of expression quantitative trait loci (eQTL) helps elucidate the connection between genotype, gene expression levels, and phenotype. However, standard statistical genetics can only attribute the changes in expression levels to loci on the genome, not specific genes. Each locus can contain many genes, making it very difficult to discover which gene is controlling the expression levels of other genes. Furthermore, it is even more difficult to find a pathway of molecular interactions responsible for controlling the expression levels. Here we describe a series of techniques for finding explanatory pathways by exploring the graphs of molecular interactions. We show several simple methods can find complete pathways that explain the mechanism of differential expression in eQTL data.

Key words: eQTL, pathway inference, gene regulation, signaling pathways.

1. Introduction

Recent studies on expression quantitative trait loci (eQTL) have revealed that differential gene expression is sometimes tightly linked to variation in specific chromosomal locations (1, 2). When gene expression is also associated with phenotypes such as disease, there is great interest in discovering the pathway connecting genetic variation and differential expression. However, this remains a difficult task. The chromosomal locations generally include many candidate causative genes. Genetic markers are able to narrow the genetic variation down to a region of roughly ten genes. When the expression level of a gene is strongly linked to one locus, genetic variation in one of these genes is presumably causative for the differential expression. However, linkage only cannot tell us which gene out of all the genes in the locus is causative.

Furthermore, even when the causative gene is known, it is very difficult to predict all the molecules involved in the regulatory pathway. In some situations, the differentially expressed gene falls within the locus it is linked to – in this case (*cis*-regulation), we assume that changes in the gene’s promoter, enhancer, or other *cis*-regulatory sites effect mRNA expression levels. In many cases, however, the differentially expressed gene is physically very distant from the linked locus. We assume the locus contains genes that regulate the differentially expressed gene, potentially through a long and intricate pathway (*see Fig. 10.1*). However, uncovering the precise pathway that regulates transcription remains difficult. This is because genes, proteins, and other biological molecules form highly context-dependent interaction networks, allowing for many possible paths from a causative gene in the linked locus to the differentially expressed gene. Here, we consider several approaches to solving this problem by searching graphs of known molecular interactions. Given the broad scope of this problem, we focus on differentially expressed genes that are strongly linked to exactly one locus. The methods section describes the techniques for determining linked loci, various methods to find pathways in interaction graphs, and several approaches to evaluating these methods. Finally, we discuss an evaluation of these methods and the pathways they discover, along with possible future extensions to these methods.

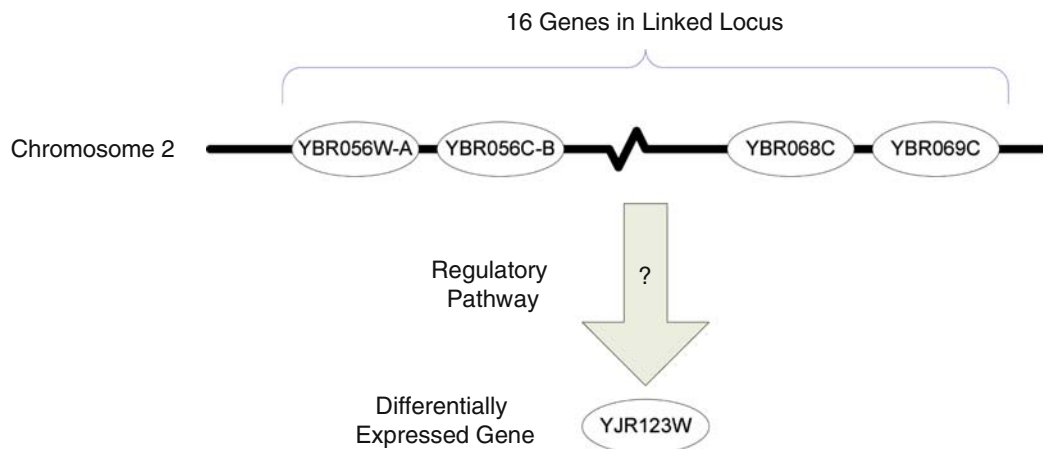


Fig. 10.1. **Finding regulatory pathways.** eQTL studies lead us to believe that genetic variation within the linked locus is responsible for the differential expression of another gene. However, we do not know which gene in the locus is responsible for regulating the expression. Furthermore, even if we did know which gene in the locus was responsible, we would still not know the regulatory pathway responsible. Here, we survey several methods for finding potential regulatory pathways from databases of known molecular interactions. This example is from eQTL studies in yeast (1), where the differentially expressed gene YJR123W has been linked to a locus with 16 genes in it.

2. Methods

Given a set of differentially expressed genes and their linked loci, we wish to find pathways from each locus to each differentially expressed gene, such that the differential expression can be explained by the pathway. This involves both discovering which gene in the locus is causing the differential expression as well as finding a coherent pathway from that gene to the differentially expressed gene, i.e., the mechanistic basis of the wiring diagram that explains the gene expression.

2.1. Finding Linked Loci

The first step in analyzing eQTL data is determining linkage. There has already been a great deal of work to find linked loci through quantitative methods. Those methods are not the focus of this chapter; here we highlight a few techniques to direct further exploration of the reader. All of these methods involve assessing the effect each locus has on the expression level of every gene. The most direct approaches use a Wilcoxon ranksum test to assign statistical significance to each locus (1, 3). More sophisticated techniques include calculating linkage by simultaneously considering multiple markers and intervals, as well as additional corrections for multiple testing (1, 4–6). Though we only focus on analysis once linkage information is in hand, we want to emphasize the strong dependence of all methods on the assessment of linkage; clearly, it is very important to determine linkage very carefully and rigorously (*see Note 1*).

2.2. Building a Molecular Interaction Graph

We wish to find pathways where each interaction is a known interaction between two molecules. This requires assembling a catalog of all interactions, to facilitate searching. Thus far, we have focused on protein–protein, transcription regulation, and phosphorylation interactions. There are several publicly available databases of this information, including many high-throughput experiments (7–10). We have also included predicted protein–protein interactions from Bioverse (<http://bioverse.compbio.washington.edu>) (11), determined through the interolog method. Note that this does not represent all known interactions; further work must explore the inclusion of metabolites as well as the effect of non-coding RNA (*see Note 2*). Related chapters discuss the techniques for reconstructing interaction graphs.

The most natural way to store these interactions is in the form of a graph, with nodes for each molecule and edges for each interaction. Due to the sparseness of these graphs, an adjacency list is the most efficient way to store the edges of the graph. The directionality of each edge should correspond to the type of interaction, e.g., physical interactions are bidirectional while transcription regulation interactions are directed from transcription factors to their targets.

2.3. Finding Pathways Between Differentially Expressed Genes and Linked Loci

With a set of differentially expressed genes and their linked loci, as well as an interaction graph, we can formulate our problem of finding plausible causative pathways. All of these methods attempt to find the causative gene and pathway in the same step. We will search all paths from the differentially expressed gene to all genes in the linked locus; the most plausible causative path should lead to the correct causative gene. In the next few sections, we will examine several methods for determining which pathways are the most plausible.

2.3.1. Finding Linear Paths Through Directed Graph Searching

The simplest formulation of a pathway is a linear sequence of interactions, from the causative gene to the differentially expressed gene. Our only obstacle is the choice of method for evaluating pathways and choosing the best one. Searching a graph for linear paths is already a well-studied problem. We explain a variety of different evaluating pathways, beginning with very simple approaches and building up to more complex procedures. The most straightforward method of evaluating a pathway is by its length. Clearly, a shorter pathway is more believable than a very long one. We can find the shortest path from genes in the linked locus to the differentially expressed gene through a standard breadth-first search (BFS). For efficiency, instead of searching for pathways from each gene in the linked locus to the differentially expressed gene, we can reverse the direction of all edges in the graph and search for paths from the differentially expressed gene to each gene in the linked locus. BFS is guaranteed to find the shortest path to every node, and is very fast and efficient.

Unfortunately, path length alone is an insufficient criterion for determining the most likely causative pathway. Often there will be many pathways of the same length. Furthermore, in many cases BFS will find pathways involving genes with uncorrelated expression levels, while a slightly longer pathway exists, which involves highly correlated genes. Using this idea, we can favor pathways with highly correlated genes, by assigning a cost to each edge and searching for the lowest cost path. Let $\text{corr}(i, j)$ be the correlation coefficient of the expression levels for genes i and j . For each edge (i, j) in the interaction graph, we compute a cost $C(i, j)$:

$$C(i, j) = -\log|\text{corr}(i, j)|$$

The absolute value of the correlation coefficient is taken because both positive and negative correlations are believable in biological pathways, corresponding to activators and repressors. The negative log is taken because our graph search finds the lowest cost paths, while we wish to favor the interactions between highly correlated genes (*see Note 3*). Finding lowest cost paths can be implemented very efficiently using a uniform cost search. While BFS uses a first-in first-out queue to order the exploration of nodes

in the graph, with a uniform cost search we explore nodes with the shortest pathway. (This is done efficiently through use of the priority queue data structure.)

Though this finds pathways with highly correlated genes, many of the pathways are still not biologically plausible. In particular, as these pathways are intended to explain gene expression level, we expect the differentially expressed gene to be immediately controlled by some transcription factor. We can make simple modifications to our graph searching methods to take advantage of the extra biological information we have on each interaction. For example, it is easy to modify a uniform cost search to ensure the first step is always to a transcription factor. Furthermore, the interaction graph could be extended to include other types of interaction data. For example, if microRNA and their targets were included in the interaction graph, we might expect microRNA to be involved in regulating expression in a manner similar to transcription factors.

2.3.2. Random Walks Across an Interaction Graph

One major shortcoming of all of the previous approaches has been that they find only linear pathways. True biological pathways involve many non-linear components, such as parallel pathways, which give biological systems added robustness, or feedback loops, which allow large responses to small stimuli. In a recent paper, Tu et al. proposed using a random walk across an interaction graph to find causative pathways (3). Their method involves finding pathways in the reverse direction, so it begins by reversing the direction of all edges in the interaction graph. Then they perform the following steps:

1. In the reversed interaction graph, start at the differentially expressed gene.
2. Randomly choose a neighbor of the current node, and go to that node.
3. If the new node is a gene in the locus, then stop the search and record the end gene.

If not, return to **step 2**.

4. Repeat the random walk 10,000 times. Choose the causative gene as the gene in the linked locus that was visited most frequently.

When choosing a neighbor to visit, nodes with highly correlated expression levels are favored. The probability of choosing a neighbor for the next step is proportional to the correlation coefficient of the expression levels. Furthermore, to prevent arbitrarily long paths, the walk is terminated after ten steps if it still has not reached a gene in the linked locus.

The choice of a causative pathway is not as clear in this scenario. Many of the nodes that were visited were along spurious paths. However, we do not only want to find the most visited

nodes along a linear path between the differentially expressed gene and the causative gene; this approach was chosen for its ability to find non-linear pathways. Thus, we count how many times all nodes were visited during the random walks. If the causative gene has been visited c times, we include all genes that have been visited some fraction c/k times (for example, $c/3$ times), and which are on a path from the differentially expressed gene to the causative gene.

Here we explain one small optimization of this approach. Instead of actually performing a random walk, we can replace it with a closed form equivalent; we will compute the probability the random walk will end at each gene in the linked locus. This improves efficiency, and more importantly it is able to exactly model the random walk without the need for many repeated trials.

If we number the nodes from 1 to n , let us denote the probability of being at node i after t steps as $P_t[i]$. Let us denote the start node. We will denote the probability of transitioning to node j from node i to be $T[i, j]$. Then we can precisely compute $P_t[i]$:

$$P_0[s] = 1 \text{ and } P_0[i] = 0 \text{ for } i \neq s.$$

$$P_t[j] = \sum_i P_{t-1}[i] T[i, j].$$

To get to node j at time t , the random walk must be in some neighbor node i at time $t - 1$, and then it must move from node i to node j .

Using these equations, the probability distribution can be efficiently calculated to any arbitrary time t by storing only two arrays of size n , one for the probabilities at time t and one for time $t - 1$.

While this approach is straightforward and has been used to discover pathways from eQTL data, several variants of this method still need to be explored. Several other distance metrics using random walks across a graph have been proposed – further testing is required to determine which method is optimal (see (12) for a review of other distance metrics). Furthermore, the interaction graphs in higher organisms are significantly larger; for large enough graphs, these methods will be impractical due to memory constraints.

2.4. Evaluation Using Gene Knockouts

It is currently difficult to assess the accuracy of inferred pathways from eQTL data because there is no good data set for validation. In virtually all cases, the true pathway is unknown. Furthermore, we cannot expect to find all previously known biological pathways – the nature of the eQTL data relies on natural genetic variation, and this variation may not perturb known pathways. Thus, there is no good “gold standard” data set (see **Note 4**).

Tu et al. proposed a solution to this problem by using gene knockouts to create a fake “gold standard” eQTL data set (3). The Rosetta compendium of gene knockouts in yeast (13) provides expression levels in over 200 gene knockout experiments. This gives the same information as in eQTL data: expression levels combined with genetic variation. In the knockout experiments, we know precisely which gene has been removed, while genetic markers from eQTL studies narrow the region down to a locus with ~5–50 genes. To simulate eQTL data, we create a locus of ten genes around the true knockout. Each of the proposed methods can then be tested with the expression data and these created loci. If the causative gene in the inferred pathway is the gene knockout, then we conclude the method was successful. We are assuming that if the predicted pathway includes the correct causative gene, then the entire pathway is correct; this assumption will not always be true.

It is very difficult to determine which genes are truly differentially expressed due to the gene knockouts. Expression data is extremely noisy; to decide which genes are differentially expressed, we must set a threshold. Also, as proposed by Tu et al., we can cluster genes by common transcription factors, and only take large clusters. Both of these methods require subjective thresholds.

These methods greatly simplify our test cases. We have found the accuracy of these methods is highly dependent on the subjective decisions for gene expression. Nonetheless, though these test cases are far from perfect, this is a good first evaluation before turning to much more expensive and time-consuming verification with laboratory experiments.

3. Discussion

We have outlined several techniques for finding pathways from eQTL data. In our experiments on gene knockouts, we have found that all of the methods above perform significantly better than random choice. For example, these methods suggested an interesting feedback cycle in the MAPKKK pathway. The pathway starts from the pheromone response element Ste2 and after many interactions it regulates Dig1; we find that Dig1 in turn regulates the transcription of Ste2 (*see Fig. 10.2*). This feedback cycle has been reported before through direct experimentation; we simply highlight this as one noteworthy pathway uncovered by these methods (14–17).

We also consider one example from using true eQTL from yeast (1). Consider the differentially expressed gene YJR123W and the set of 16 genes that are in the linked locus, as in **Fig. 10.1**. All

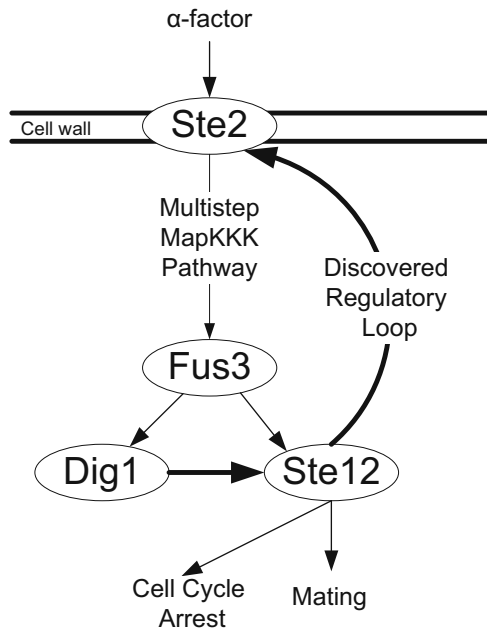


Fig. 10.2. **An example pathway found in gene knockout tests.** The MAPKKK pathway is known to start when the pheromone receptor Ste2 interacts with α -factor. After many steps, the pathway regulates Dig1 and Ste12. From knockout experiments, we found that Dig1 regulates Ste2, demonstrating a feedback cycle in the MapKKK pathway. This feedback cycle was already known, but it serves to demonstrate the types of pathways that can be uncovered through these methods.

of the methods discussed above consistently highly rank a pathway from YBR059C to YJR123W (*see* Fig. 10.3). We note that the roles we predict are consistent with the annotations in CYGD (8) and previous literature: YBR059C is a well-known kinase, and has been implicated as an important protein in signaling pathways (18); YKR092C is known to be phosphorylated, and it is believed that it may bind DNA or act as a cofactor (19); YDR174W is known to bind DNA or act as a cofactor for transcriptional regulation (20, 21). As the true pathway is unknown in this case, we cannot be certain of these results without direct experimental validation. However, the agreement between this proposed pathway and the available literature data is very promising. In total, we predict 411 explanatory pathways for eQTL in yeast, and expect many of these will be correct given the accuracy in gene knockout experiments.

In general, we have found the accuracy of all the methods are extremely similar (*see* Fig. 10.4). There is also a high degree of overlap between the pathways found by each method (*see* Fig. 10.5). However, these results are highly dependent on the set of test conditions used (the p -value cutoffs for defining differentially expressed genes from the knockout data, as well as the

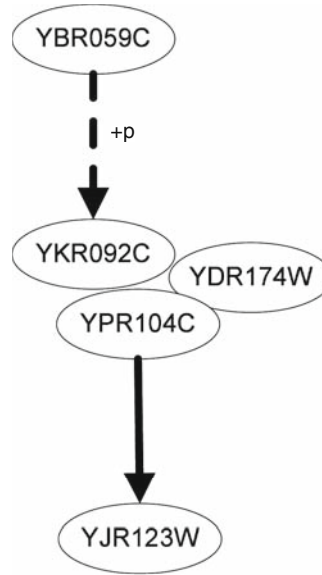


Fig. 10.3. A pathway explaining the differential expression of YJR123W in yeast eQTL experiments (see Fig. 10.1). Here the *dashed line* represents the phosphorylation of YKR092C by the kinase YBR059C, the grouped proteins are a physical complex formation, and the final *solid line* is a transcriptional regulation interaction by transcription factor YPR104C to its target YJR123W.

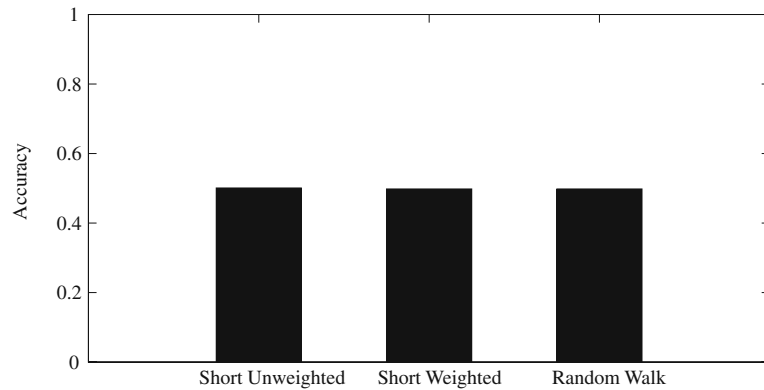


Fig. 10.4. **A comparison of the accuracy of the three methods to map pathways from eQTL.** All the methods have nearly identical accuracy on the gene knockout tests.

p-value cutoffs used in defining the interaction graph). We have found that under some conditions simply taking the shortest unweighted path outperforms other methods, while in other conditions the random walk method is the best. We are still attempting to uncover patterns that explain the differences in each method. However, we can clearly see that each method is capable of producing explanatory pathways, as seen in the examples above.

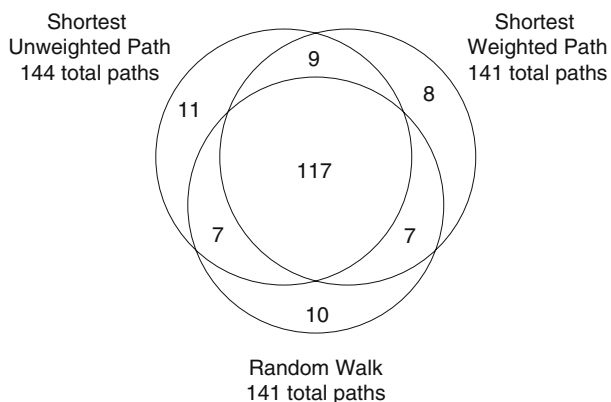


Fig. 10.5. **Overlap of pathways found by each of the three methods evaluated on the known gene knockout test cases.** Aside from minor differences, the methods are correct for the same test cases.

Our work highlights the strong dependence all of these methods have on the interaction graph. If even one interaction from the true pathway is missing from the interaction graph, none of these methods will be able to deduce the correct pathway. Furthermore, if the interaction graph contains many spurious edges, then all of these methods can easily be misled to deduce the wrong pathway, by finding a “short-circuited” pathway. We hope that this work will motivate continuing work to improve the accuracy and coverage of current molecular interaction databases (*see Note 2*).

Though these approaches have been somewhat successful, they are far from complete. They do not achieve perfect accuracy, and they cannot find explanatory pathways in many cases. Indeed, we have also eliminated a large portion of the data by considering only highly differentially expressed genes and genes that are clustered by transcription factor. Still, we are able to find a large number of novel pathways with these methods.

The approach described here also serves as an extensible framework to try additional methodology. Each of the variants described here requires only very small changes to the underlying code. With minimal effort, additional approaches can be added and compared to the existing methodology. For example, one could easily test what effect changing the interaction graph has on the predictions. This framework greatly facilitates the development of techniques for generating new candidate pathways.

We wish to emphasize that these approaches only serve for hypothesis generation. Verification of the proposed pathways requires additional work in the laboratory. Though the accuracies of these methods are not ideal, they do narrow a nearly infinite space of possibilities down to a tractable number. We cannot and do not expect these methods to be perfect, given the errors in our

interaction graph, the missing molecular interactions (e.g., metabolites), the noise in the gene expression levels, and the numerous other factors that are not modeled at all (e.g., chromatin remodeling).

One major drawback of all these approaches is that they are designed to find pathways when a gene shows strong linkage to only a single locus. This accounts for a small fraction (less than 20%) of real eQTL data – as shown by the studies of Brem and Kruglyak (1). Several studies have proposed methods for analyzing eQTL data by simultaneously considering all genes and all loci (22, 2). These approaches attempt to construct a complete network of all genes and their relationships using machine-learning techniques. However, these approaches only indicate the influence of some genes on the expression level of other genes; they do not deduce pathways of molecular interactions, which provide a mechanistic basis for the wiring diagram for gene expression.

The techniques evaluated in this manuscript and the techniques for network inference complement each other and should be integrated together to provide the most coherent explanation possible. As a first step, once the network reconstruction algorithms have uncovered the most important relationships, the techniques described here could be used to find the pathway of molecular interactions that are responsible. For example, if the network reconstruction algorithms suggest that gene A influences gene B (through some potentially long and indirect method), we could search for a pathway in the interaction graph connecting gene A to gene B. Furthermore, we feel that the interaction graph should be directly incorporated into the network reconstruction algorithms, through the use of structure priors, i.e., the network reconstruction algorithms should favor inferring relationships along pathways with very good explanations. We are currently exploring these approaches to enable the inference of a greater number of more accurate pathways.

4. Notes



1. *Importance of determining linkage.* As noted previously, all subsequent analysis is based on first using statistical tests to determine to which loci each gene is linked. There are several techniques that differ in their sophistication.
2. *Importance of the interaction graph.* All of the methods mentioned here are highly sensitive to the edges, which are in the interaction graph. If even one edge in the true pathway is missing from the graph of molecular interactions, then it will

be impossible to discover the true pathway. If the graph contains false edges, all of the graph searching methods can very easily be misled. Many of the interactions in our network come from high-throughput experiments, such as yeast two-hybrid or ChIP-chip experiments. Converting this data to a graph requires the use of subjective p -value cutoffs. This means the interaction graph will always contain some spurious interactions, and it will be missing many true interactions. We are still exploring the robustness of these methods to errors in the interaction graph.

3. *Converting correlations to distances.* When converting expression correlations to distances, large correlations must be converted into small distances and small correlations into large distances. We chose the distance to be $1 - |\text{corr}(i, j)|$; however, this is simply a convenient heuristic. Several other transformations could be applied; we have experimented with $-\log(|\text{corr}(i, j)|)$ and $1/(|\text{corr}(i, j)|)$, and found they all obtain similar results.
4. *No gold standard data set for testing.* As there is no eQTL data set for which all of the true pathways are known, it is very difficult to determine if any new methods are successful or not, without direct experimentation. The best test set that can be used currently is the gene knockout data. However, we must still decide to use some subjective criteria to decide which genes are differentially expressed. Furthermore, we do not know the true pathway between the knockout and the differentially expressed genes.

Acknowledgments

This work was supported by NSF Graduate Research Fellowship DGE0203031.

References

1. Brem RB, Storey JD, Whittle J, Kruglyak L. Genetic interactions between polymorphisms that affect gene expression in yeast. *Nature* 2005, 436:701–03.
2. Schadt EE, Monks SA, Drake TA, Lusis AJ, Che N, Colinayo V, Ruff TG, Milligan SB, Lamb JR, Cavet G, Linsley PS, Mao M, Stoughton RB, Friend SH. Genetics of gene expression surveyed in maize, mouse and man. *Nature* 2003, 422:297–302.
3. Tu Z, Wang L, Arbeitman MN, Chen T, Sun F. An integrative approach for causal gene identification and gene regulatory pathway inference. *Bioinformatics* 2006, 22:e489–96.
4. Doerge RW, Zeng ZB, Weir BS. Statistical issues in the search for genes affecting quantitative traits in experimental populations. *Stat Sci* 1997, 123:195–219.
5. Storey JM, Akey JM, Kruglyak L. Multiple locus linkage analysis of genomewide expression in yeast. *Plos Biol* 2005, 3:e267.

6. Chen L, Storey JD. Relaxed significance criteria for linkage analysis. *Genetics* 2006, 173:2371–81.
7. Lee S, Pe'er D, Dudley AM, Church GM, Koller D. Identifying regulatory mechanisms using individual variation reveals key role for chromatin modification. *PNAS* 2006, 103:14062–67.
8. Gldener U, Mnsterkter M, Kastenmller G, Strack N, van Helden J, Lemer C, Richelles J, Wodak SJ, Garca-Martnez J, Prez-Ortn JE, Michael H, Kaps A, Talla E, Dujon B, Andr V, Souciet JL, De Montigny J, Bon E, Gaillardin C, Mewes HW. Cygd: The comprehensive yeast genome database. *Nucl Acids Res* 2005, 33(Database Issue): D364–68.
9. Harbison CT, Gordon DB, Lee TI, Rinaldi NJ, Macisaac KD, Danford TW, Hannett NM, Tagne J, Reynolds DB, Yoo J, Jennings EG, Zeitlinger J, Pokholok M, Kellis DK, Rolfe PA, Takusagawa KT, Lander ES, Gifford DK, Fraenkel E, Young RA. Transcriptional regulatory code of a eukaryotic genome. *Nature* 2004, 431:99–104.
10. Ptacek J, Devgan G, Michaud G, Zhu H, Zhu X, Fasolo J, Guo H, Jona G, Breitkreutz A, Sopko R, McCartney RR, Schmidt MC, Rachidi N, Lee S, Mah S, Meng L, Stark MJR, Stern DF, De Virgilio C, Tyers M, Andrews B, Gerstein M, Schweitzer B, Predki PF, Snyder M. Global analysis of protein phosphorylation in yeast. *Nature* 2005, 438:679–84.
11. McDermott J, Bumgarner RE, Samudrala R. Functional annotation from predicted protein interaction networks. *Bioinformatics* 2005, 21:3217–26.
12. Chandra AK, Raghavan P, Ruzzo WL, Smolensky R. The electrical resistance of a graph captures its commute and cover times. In *Proceedings of the Twenty First Annual ACM Symposium on Theory of Computing*, 1989, pp. 574–86.
13. Hughes TR, Marton MJ, Jones AR, Roberts CJ, Stoughton R, Armour CD, Bennett HA, Coffey E, Dai H, He YD, Kidd MJ, King AM, Meyer MR, Slade D, Lum PY, Stepaniants SB, Shoemaker DD, Gachotte D, Chakraburty K, Simon J, Bard M, and Friend SF. Functional discovery via a compendium of expression profiles. *Cell* 2005, 102:109–26.
14. Fields S, Herskowitz I. The yeast *ste12* product is required for expression of two sets of cell-type-specific genes. *Cell* 1985, 42: 923–30.
15. Song O, Dolan J, Yuan Y, Fields S. Pheromone-dependent phosphorylation of the yeast *ste12* protein correlates with transcriptional activation. *Genes Dev* 1991, 5: 741–50.
16. Chen Q, Konopka BJ. Regulation of the g-protein-coupled alpha-factor pheromone receptor by phosphorylation. *Mol Cell Biol* 1996, 16:247–57.
17. Hartig A, Holly J, Saari G, MacKay VL. Multiple regulation of *ste2*, a mating type-specific gene of *saccharomyces cerevisiae*. *Mol Cell Biol* 1986, 6:2106–14.
18. Cope MJ, Yang S, Shang C, Drubin DG. Novel protein kinases *ark1p* and *prk1p* associate with and regulate the cortical actin cytoskeleton in budding yeast. *J Cell Biol* 1999, 144:1203–18.
19. Meier UT. Comparison of the rat nucleolar protein *nopp140* with its yeast homolog *srp40*. Differential phosphorylation in vertebrates and yeast. *J Biol Chem* 1996, 271:19376–84.
20. Lu J, Kobayashi E, Brill SJ. Characterization of a high mobility group 1/2 homolog in yeast. *J Biol Chem* 1996, 271:33678–85.
21. Gadal O, Labarre S, Boschiero C, Thuriaux P. *Hmo1*, an hmg-box protein, belongs to the yeast ribosomal DNA transcription system. *EMBO J* 2002, 21:5498–507.
22. Lee SI, Pe'er D, Dudley A, Church G, Koller D. Identifying regulatory mechanisms using individual variation reveals key role for chromatin modification. *Proc Natl Acad Sci USA* 2006, 103:14062–67.

Chapter 11

Methods for the Inference of Biological Pathways and Networks

Roger E. Bumgarner and Ka Yee Yeung

Abstract

In this chapter, we discuss a number of approaches to network inference from large-scale functional genomics data. Our goal is to describe current methods that can be used to infer predictive networks. At present, one of the most effective methods to produce networks with predictive value is the Bayesian network approach. This approach was initially instantiated by Friedman et al. and further refined by Eric Schadt and his research group. The Bayesian network approach has the virtue of identifying predictive relationships between genes from a combination of expression and eQTL data. However, the approach does not provide a mechanistic bases for predictive relationships and is ultimately hampered by an inability to model feedback. A challenge for the future is to produce networks that are both predictive and provide mechanistic understanding. To do so, the methods described in several chapters of this book will need to be integrated. Other chapters of this book describe a number of methods to identify or predict network components such as physical interactions. At the end of this chapter, we speculate that some of the approaches from other chapters could be integrated and used to “annotate” the edges of the Bayesian networks. This would take the Bayesian networks one step closer to providing mechanistic “explanations” for the relationships between the network nodes.

Key words: Networks, pathways, functional genomics, review, computational biology.

1. Definitions

Presently, the words “pathway” and “network” are used almost interchangeably. However, in a given use, the constructs these words represent can be vastly different (e.g., literature relationships, physical interactions, or coupled chemical reactions). For clarity in the subsequent discussions, it is helpful to use more

specific and robust terms and define what is meant by each term. Hence, for our purposes we will use the following definitions:

- *Molecular or biochemical pathway*: A set of coupled chemical reactions or signaling events. Nodes are molecules (often substrates) and edges represent chemical reactions. We also include conformational changes as the result in downstream signaling via other chemical reactions in this definition.
- *Physical interaction network*: A graphical representation of molecular binding interactions such as a protein–protein interaction network. Nodes are molecules; edges represent physical interactions between molecules.
- *Correlation or co-expression network*: A graphical representation that averages over-observed expression data. Nodes are molecules (typically mRNAs); edges represent correlations between expression levels of connected nodes.
- *Bayesian expression network (Bayes nets)*: A directed, graphical representation of the probabilities of one observation given another. In our use, nodes represent mRNA molecules, edges represent the probability of a particular expression value, given the expression values of the parent nodes.
- *Knowledge-based network*: A graphical representation of relationships between genes or molecules as inferred from external knowledge. An example would be a literature-based network in which the nodes represent genes and the edges represent the presence of a co-citation in a Pubmed abstract.

2. The Current State of Gene Annotation

Prior to discussing methods for pathway inference, it is useful to assess the current state of gene functional annotation. The motivation for this discussion is that one can think of placement in (a) biological pathway(s) as the ultimate form on annotation. We will briefly explore the current state of understanding gene function in the context of yeast and human genomes. The question we wish to ask is “for what fraction of the genome do we know the gene function?”

This simple question is immediately complicated when one attempts to assess the level of detail of the knowledge. For some genes, we know a great deal; for example, for many metabolic enzymes we know the details of the reaction the enzyme catalyzes, the specific substrates on which it operates, the necessary co-factors, and the surrounding context of other coupled reactions. For some genes we know very little; for example, we may be very confident that

an enzyme is a protease without knowing the substrate(s) for the enzyme or the context in which it operates. At the extreme, there are a number of genes that we can recognize as genes, but have no idea about their function. In addition, gene function is complicated by other factors such as alternative splicing, functional but non-coding RNAs, and post-translational modifications, just to name a few.

To put this in perspective, consider the knowledge we have about *Saccharomyces cerevisiae* or yeast. Yeast is one of the most highly studied organisms on the planet. Much of the current, easily accessible genome knowledge of the yeast *Saccharomyces cerevisiae* is represented in the *Saccharomyces cerevisiae* Genome Database (SGD – <http://www.yeastgenome.org/>). At present, there are 4,397 verified yeast ORFs in SGD. In order to demonstrate the variability of annotation available, we examine two extremes of annotation (e.g., very little knowledge about the function and mapped to a biochemical pathway). In SGD 267 (6.1%) verified ORFs are annotated with phrase unknown function and another 250 are annotated with phrase containing putative, hypothetical or proposed. Therefore, we have some idea of what roughly 90% of all yeast genes do at least at the level of being able to assign a crude function.

At the other extreme of functional annotation, only about 10% (460) of the verified yeast ORFs are mapped to a biochemical pathway in SGD. In the KEGG2 database (1, 2), there are a total of 6,224 yeast ORFs of which 1,305 (21%) map to a pathway in KEGG2. While these mappings are somewhat biased toward metabolic pathways and cannot be claimed to represent the sum total of pathway knowledge, they do represent what pathway mappings are presently database accessible. Regardless, it is fair to say that we do not have pathway information for the vast majority of yeast genes.

For the human genome, we used the EBI UniProt database as our source to look at the status of functional annotation. In UniProt, there are 35,715 unique proteins (includes some splice variants). Of these, 21,087 (59%) have no gene ontology annotation and 11,679 (32%) have no associated key words. For pathway mappings we looked to the KEGG2 database. The total number of human genes represented in the KEGG2 database (www.genome.jp/kegg/kegg2.html) is 26,694. Of these, there are a total of 4,373 unique genes, of just 16.3%, that have been mapped to a KEGG2 pathway.

3. Value of Pathway Information and Current Knowledge of Molecular Pathways

Without doubt, some of the most significant advances in biochemistry and molecular biology in the past 100 years have been in the development and integration of the understandings necessary to

construct a number of detailed biochemical pathways. Such pathways are tremendously important, as they provide a mechanistic basis (via the underlying chemistry of a system) for biological phenomena and, therefore, provide predictive ability. For example, the connection between a gene deletion and a given metabolic disease can be inferred from the knowledge of metabolic pathways. Over the past 40 years, a significant fraction of all Nobel Prizes in Medicine and Chemistry have been awarded to researchers for discoveries related to the understanding of specific biochemical pathways.

Biochemical pathways in graphical form are a condensed and abstract representation of knowledge that has been gained from a great deal of genetics and/or detailed chemical experimentation. Since the earliest creation of crude hand-drawn prototypes of metabolic pathways by Donald Nicholson in 1955 (*see* <http://www.tcd.ie/Biochemistry/IUBMB-Nicholson/hist.html>), there has been high demand for this type of information [as represented by a more than 40-year history of printed IUBMB-Sigma-Nicholson Metabolic Pathways wall charts from Sigma Aldrich, a similar 20+ year offering from Boehringer Mannheim GmbH and, more recently, a proliferation of public (KEGG, BioCyc, GenMapp etc) and commercial (Jubilant, Pathway Assist, etc.) databases of similar and expanded information]. Much of the driving force for the creation of public and commercial databases of pathways has been the desire to interpret functional genomics data (in particular, expression data) in the context of known biochemical pathways. Hence, in addition to these databases, a variety of software tools have been developed to display expression data on biochemical pathways and other types of biological networks (*see* for example, the cytoscape software tool at www.cytoscape.org).

As previously discussed, only a relatively small fraction of even the yeast genome has been mapped to one or more biochemical pathways. While the rate of accumulation of other types of genomic information has been rapid, the ability to assign genes to pathways has proceeded at a much slower pace. As we move forward with functional genomics experimentation that wishes to make use of pathway information, new tools and methods to more rapidly infer pathways and networks from genome-scale data are needed.

4. Approaches to Obtaining Genome-Wide Function, Network, and Pathway Information

In this section, we will discuss the experimental methods to generate genome-wide data that inform us about biological networks and biochemical pathways, and the computational methods to integrate such data types to better infer biological networks and

pathways. We feel there are two fundamental goals in biological network reconstruction: first, the network should provide, lead to, or assist in the development of an underlying mechanistic basis for any behaviors predicted from the network; and secondly, the biological network must be predictive of some behavior(s) of the system. Our interest in pathway inference is driven by the goal to more rapidly infer the predictive networks that are connected to underlying molecular interactions.

4.1. Methods to Measure and Compute Physical Interaction Networks

At present, there are three primary types of physical interactions that are measured at genome-wide scale. High-throughput protein–protein interaction measurements have been accomplished by a variety of techniques, including yeast two-hybrid, co-immunoprecipitation or co-IP (often coupled to mass spec analysis), and protein arrays (for a recent review *see* (3)). Protein–DNA interactions have been measured at large scale using the “CHIP on chip” technique (4, 5). More recently, techniques to measure protein–nucleic acid interactions have been extended to protein–RNA interactions (6, 7). We also anticipate that data from large-scale screens to characterize most RNA–protein interactions within a given organism (yeast and human, in particular) will become available in the near future. For any given organism, the amount of effort and resources required to experimentally elucidate physical interaction networks is quite large. To address this issue, a number of investigators have been developing techniques to predict physical interaction networks computationally using a variety of methods (8–12). One of the most common methods makes use of the similarity of pairs of proteins in a target organism to a database of known protein interactions covering a diverse range of organisms (13, 10, 11). For many organisms, the predicted physical interaction networks are the only genome-scale source of such information.

There are a few issues with both the predicted and the experimentally determined physical interaction networks. Many experimentally determined physical interactions may be biologically irrelevant or only relevant under very specific conditions. For example, the CHIP-on-chip data generated for NfKb shows binding to regions just upstream of approximately 10% of all the genes in the human genome (14), yet it is unlikely that NfKb is actively regulating that large a fraction of the genome. Perhaps equally troubling with regards to CHIP-on-chip data is a very large false-negative rate. We have compared much of the publicly available yeast CHIP-on-chip data to the databases of experimentally verified transcription factor (TF) regulation; we have found that current CHIP-on-chip techniques fail to find binding for about 80% of the known TF regulatory interactions (15). Finally, while the physical interaction networks (experimental and predicted) provide valuable insight into relationships between genes, such networks are generally not predictive of systems behavior. The effect of a perturbation such as a change in expression level of a given

protein or RNA is not predictable from the network; therefore, what is needed are additional experimental and computational methods to both improve the reliability of the physical interaction networks and create biological networks that are predictive of systems behavior.

4.2. Methods to Infer Regulation and/or Create Predictive Networks

4.2.1. Identification of Co-regulated Genes by Cluster Analysis

Some of the earliest work in this regard is the large-scale analysis of gene expression as a function of cell cycle in yeast (16). This work presented the results of a comprehensive series of experiments designed to identify all protein-encoding transcripts in the genome of *S. cerevisiae*, which are regulated by the cell cycle. DNA microarrays were used to analyze mRNA levels in cell cultures that had been synchronized by three independent methods. The data were analyzed by looking for genes whose expression varied in a cyclic manner during the cell cycle and also by cluster analysis (a method for discovering patterns in complex data sets). In this case, cluster analysis was used to identify genes that shared common patterns of expression across the experiments. In particular, the study focused on genes that behaved similarly to other genes that are known to be cell-cycle-regulated. A total of 800 genes, or approximately 13% of all protein-coding genes in the genome, were found to be cell-cycle-regulated.

For each of the 800 genes identified as cell-cycle-regulated, 700 bp of genomic sequence immediately upstream of the start codon was analyzed to identify the potential binding sites for known or novel factors that might control expression during the cell cycle. The majority of the genes were shown to have good matches to known cell cycle transcription factor binding sites. In addition, the distribution of these putative transcription factor binding sites and their positions relative to the start codon contain information that is predictive of the phase of the cell cycle during which a given gene is expressed.

Cluster analysis has been shown to be a powerful tool to identify genes whose expression levels are correlated across numerous experiments, and often groups of such genes share common promoters and/or functions. Hence, the use of the methods pioneered by Spellman and colleagues, e.g., the meta-analysis of massive amounts of gene expression data to identify genes that are co-expressed followed by promoter analysis (Fig. 11.1), is now fairly commonplace [e.g., (17–22)]. However, this approach of “guilt by association” (23) does have its limitations. In particular, array data can be quite noisy and cluster analysis will find patterns in noise as well as signal. Hence, genes that appear to be co-expressed in a large number of experiments may be grouped on the basis of noise. Additionally, the number of experiments that must be analyzed to be reasonably certain that co-expression in these experiments is indicative of co-regulation can be quite high. In the work recently published from our group (24), we re-analyzed large sets of gene expression data measurements

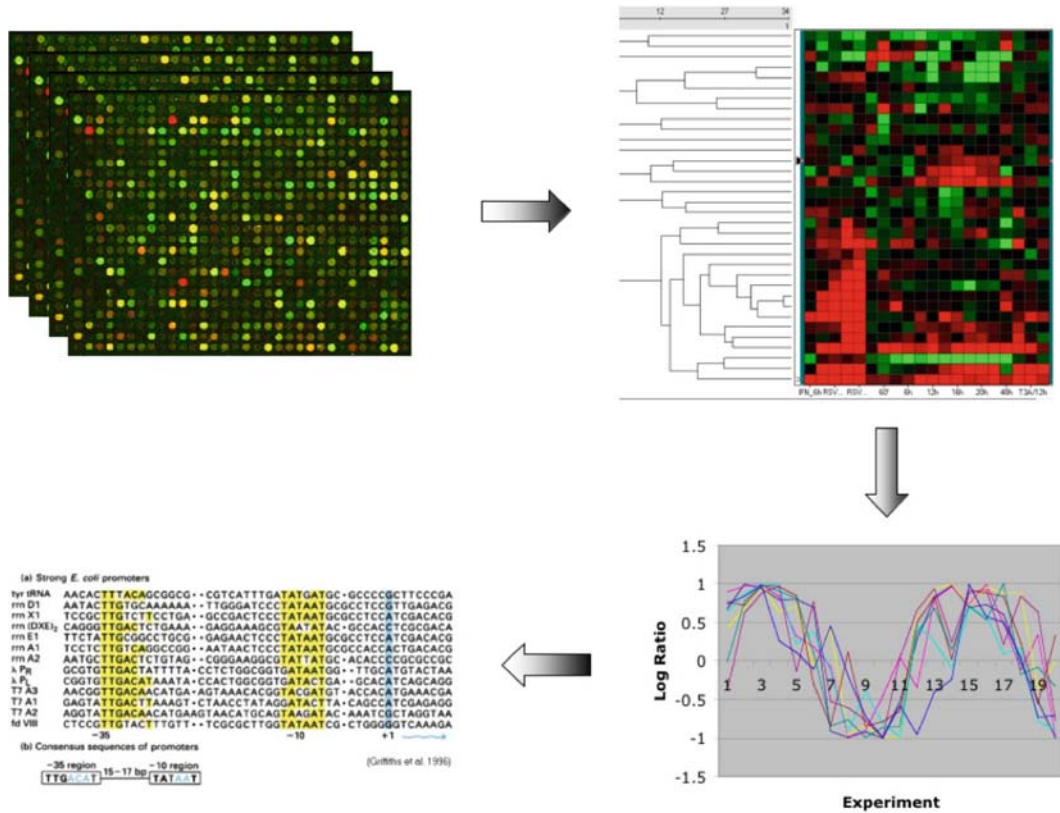


Fig. 11.1. A common strategy to identify co-regulated genes. Cluster analysis is applied to multiple array experiments to identify genes whose expression levels are correlated across all experiments (*upper 2 and lower right panels*). Analysis of the upstream regions of DNA sequence is performed to identify common motifs that are putative binding sites for transcription factors (*lower left panel*).

in yeast to estimate how many experiments are necessary for co-expression to be a reliable indicator of co-regulation. Our estimates are that roughly 50 distinct conditions (experiments) are necessary in yeast. Given that gene regulation in mammals is far more complex than it is in yeast, our work suggests that clustering across even larger numbers of array experiments will be required for this approach to reliably infer co-regulation in mammalian cells.

In addition to the fact that co-expression does not necessarily imply co-regulation, the approach outlined in **Fig. 11.1** is also hampered at the level of sequence analysis. Sequence analysis to identify transcription factor binding sites can be quite challenging, as binding site motifs are generally quite small (6–15 nucleotides) and are not 100% conserved in sequence across the motif. Therefore, any moderately long stretch of DNA sequence contains numerous potential motifs to which a known transcription factor may bind (13, 8). At present, the net result is that all putative transcription factor binding sites must be experimentally

confirmed by a method that demonstrates a direct physical interaction between the site and the factor *and/or* a direct influence of the factor on the expression of that gene in the appropriate physiological context.

4.2.2. Pathway Inference and Modeling from Array Data (Sometimes in Combination with Other Data Types)

Cluster analysis to identify co-regulated genes is not the only approach in attempting to infer pathways from array data. In 2001, Ideker et.al. published a manuscript in *Science* in which a systematic approach to systems biology was described (25). In brief, this approach consists of the following strategy to gain pathway information from array data:

- (i) Define all of the genes in the genome and the subset of genes, proteins, and other small molecules constituting the pathway of interest. If possible, define an initial model of the molecular interactions governing pathway function, drawing upon previous genetic and biochemical research.
- (ii) Perturb each pathway component through a series of genetic (e.g., gene deletions or over expression) or environmental (e.g., changes in growth conditions or temperature) manipulations. Detect and quantify the corresponding global cellular response to each perturbation with technologies for large-scale mRNA- and protein-expression measurement.
- (iii) Integrate the observed mRNA and protein responses with the current pathway-specific model and with the global network of protein–protein, protein–DNA, and other known physical interactions.
- (iv) Formulate new hypotheses to explain observations not predicted by the model. Design additional perturbation experiments to test these, and iteratively repeat Steps (ii), (iii), and (iv).

This approach was applied to an analysis of the yeast galactose-utilization pathway, one of the most studied pathways in biological literature (9, 10). Arrays were used to identify 997 mRNAs responding to 20 systematic perturbations of the yeast galactose-utilization pathway; these perturbations consisted of wild-type yeast and nine gene deletion strains grown under two different growth conditions (growth in the presence or absence of 2% galactose with 2% raffinose provided in both media). The observed responses of GAL genes were compared to the predicted behavior modeled from current knowledge of the galactose utilization pathway. In general, the observed mRNA changes were in good agreement with the qualitative changes in mRNA expression that were predicted based on the model. For example, growth of wild-type cells in +gal versus –gal media significantly induced *GAL1*, *GAL2*, *GAL7*, *GAL10*, and *GAL80* as expected, while deleting the positive regulators *GAL3* and *GAL4* led to a significant expression decrease in many of these genes. In –gal media, deletion of the repressor *GAL80* caused a

dramatic increase in GAL-enzyme expression; in +gal media, this deletion had little or no effect on these genes, presumably because they were already highly expressed.

However, several observations were not predicted by the model. In many cases, these suggested new regulatory phenomena that may be tested by hypothesis-driven approaches. For example, in the presence of galactose, *gal7* and *gal10* deletions unexpectedly reduced the expression levels of other GAL enzymes. Because the metabolite Gal-1-P is known to accumulate in cells lacking functional Gal7 and is detrimental in large quantities (11), one hypothesis is that the observed expression-level changes are dependent on the build-up of Gal-1-P or one of its metabolic derivatives. Utilizing this model, it is expected that the cell would limit metabolite accumulation by first sensing toxic levels through an unknown mechanism, then triggering a decrease in GAL-enzyme expression. To test the hypothesis that the effects of *gal7D* and *gal10D* are dependent on increased levels of Gal-1-P or a derivative molecule, the expression profile of a *gal1Dgal10D* double deletion growing in +gal conditions (relative to the *wt gal1* reference) was obtained. It was predicted that in this strain, the absence of *GAL1* activity would prevent the build-up of Gal-1-P and the changes in GAL gene expression would not occur. Conversely, if the expression changes did not depend on Gal-1-P (e.g., are caused by chromosomal interactions at the *GAL1-10-7* locus), they would also be likely to occur in the *gal1Dgal10D* strain. Consistent with the initial hypothesis, GAL-enzyme expression was not significantly affected by this perturbation, and the expression profile of *gal1Dgal10D*-affected genes was more similar overall to the profile of *gal1D* +gal than to that of *gal10D*+gal or any other perturbation. In addition to the effects of gene deletions in the Gal pathway on the expression level of other genes in the same pathway, the array data showed the effects of these perturbations on all transcripts in the genome. For each Gal gene deletion, numerous perturbations of gene expression in apparently unrelated pathways were observed.

An attempt was made to explain these more distant effects by mapping the gene expression data onto an integrated physical interaction network (Fig. 11.2) consisting of known transcription factor interactions and protein–protein interactions from global yeast two-hybrid data (12). This approach has been carried further through the creation of Cytoscape (www.cytoscape.org), an open source package of software that automates the visualization and mapping of data onto physical interaction networks (26, 27). This approach shows promise in that it provides a rapid way to develop hypotheses to explain observed effects on gene expression. For the yeast galactose data, some of the effects on gene expression in other, more distant pathways could be explained through putative protein–protein or protein–DNA interactions that were identified by this mapping.

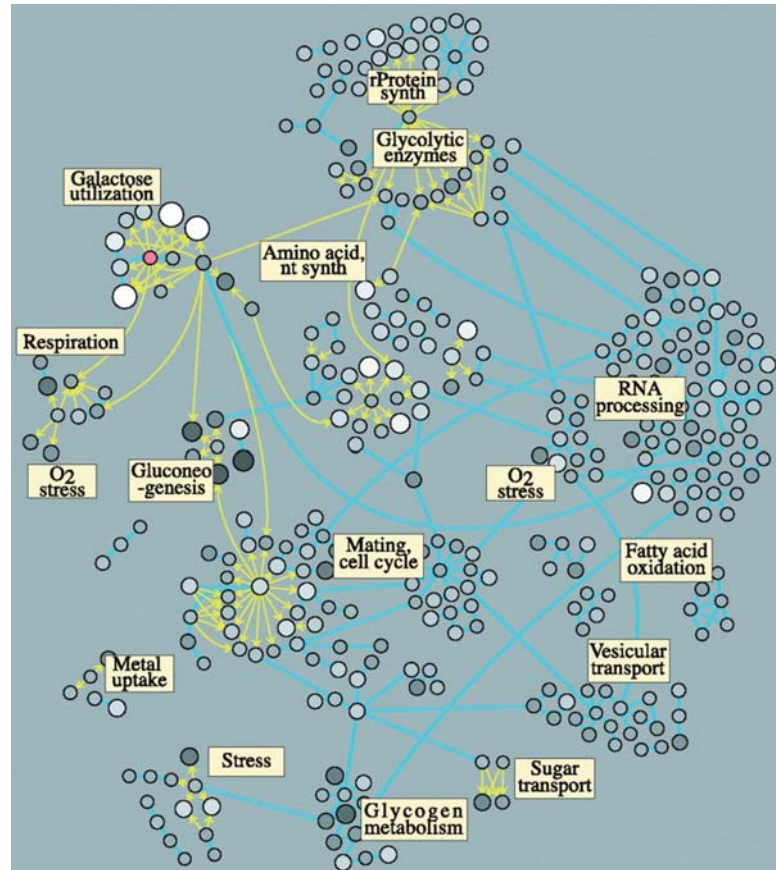


Fig. 11.2. Mapping of the yeast galactose expression data onto an integrated physical-interaction network. Nodes represent genes, *light arrows* directed from one node to another signifies that the protein encoded by the first gene can influence the transcription of the second by DNA binding, and a *dark line* between two nodes signifies that the corresponding proteins can physically interact. Effects of the gal4D +gal perturbation are superimposed on the network. GAL 4 is the hub node in the cluster of genes labeled “galactose utilization” (upper left). The gray scale intensity of other nodes represents the magnitude of changes in mRNA levels – node diameter also scales with the magnitude of change (from (25)).

There are a few recent proposals that also map expression data onto interaction networks. For example, Han et al. (28) proposed to use expression data to characterize “hubs” on an interaction network. They defined hubs as nodes (proteins) that are linked to more than five other nodes (proteins) on the network, and characterized two types of hubs: date hubs and party hubs. Party hubs are highly correlated in expression with their partners and presumably interact with them at similar times, while date hubs exhibit limited co-expression and the corresponding physical interactions usually occur at different times and/or locations. The authors also observed that the removal of date or party hubs from the network has different effects on network connectivity:

removing party hubs does not affect connectivity while removing date hubs has similar effects as attacking all hubs. In another example, Luscombe et al. (29) mapped expression data under different conditions onto a regulatory interaction network and analyzed the network dynamics exhibited by the expression data. They divided the condition-specific sub-networks into two different categories, identified network motifs (which are compact patterns of interconnection between transcription factors and targets), and regulatory hubs (which target large numbers of genes). As more global protein–DNA and protein–protein interaction data become available, we anticipate that the approach of mapping expression data to physical interaction networks will become an invaluable tool for pathway inference.

Beyond the insights into galactose metabolism in yeast and the overall approach, there is a key message that can be derived from this data: apparently simple perturbations (such as a single gene deletion) often produce complex changes in gene expression on pathways that are both closely and distantly related to the perturbation. The net result is that any single-array experiment comparing two conditions typically produces a plethora of observed changes, *only a fraction of which are directly related to the biology of interest*. Hence, it is easily possible to over-interpret one’s array data to infer relationships between the biology of interest and the observed changes that are, in fact, only distantly related to the biological phenomenon (30).

Yeang and Jaakkola (31) have recently developed a new framework for inferring models of transcriptional regulation. The models in this approach, referred to as “physical models,” are constructed using verifiable molecular attributes of the underlying biological system (Fig. 11.3). The attributes include, for example, protein–protein and protein–DNA interactions, the directionality of signal transduction in protein–protein interactions, as well as the sign of the effects of these interactions (e.g., whether an

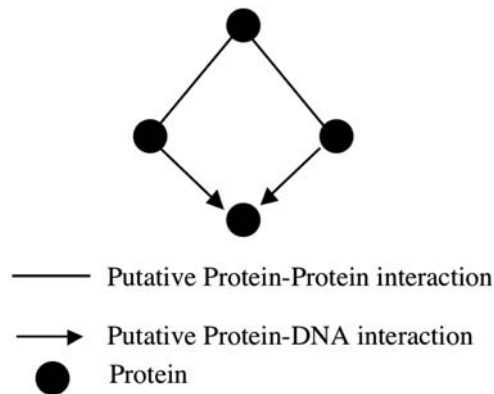


Fig. 11.3. A sample physical model (adapted from Yeang and Jaakkola (31)).

upstream gene activates or represses the downstream gene). Each attribute is defined as a variable in the model, and the variables define a collection of annotated random graphs (possible pathways). The possible graphs (or likely biological pathways) are then constrained by the available data. Protein–protein interaction data (usually from yeast two-hybrid data) and transcription factor binding data (from a variety of sources) provide constraints that are directly tied to the variables in the model. Other sources of data, such as transcriptional responses to gene deletions, provide only indirect evidence about the (physical) variables.

In their model, each effect due to a gene deletion is associated with a set of molecular cascades that could, in principle, explain the effect. The net result is a set of aggregate constraints about the physical variables in the model. The most probable model(s) is(are) then found using an approximate inference method, e.g., an iterative algorithm to efficiently sample the most likely pathway(s). Tests of this approach on datasets related to the pheromone response pathway in *S. cerevisiae* show that the resulting pathway models are consistent with previous studies. Yeang and Jaakkola show that the approach is capable of predicting gene deletion effects with a high degree of accuracy and that the method also implicates likely molecular cascades which are responsible for each observed gene deletion effect. The approaches taken in the study of Ideker et al. leveraged a great deal of pre-existing knowledge of that pathway. The approach of Yeang and Jaakkola is also highly dependent on the availability of some data to use as starting points for the construction of physical models. However, for many organisms, there is very little available physical interaction data. Hence, if we are to infer pathway information from expression data, we also need approaches that do not require large amounts of pre-existing physical interaction data.

A number of alternative approaches to pathway inference that do not require extensive pre-existing knowledge of physical interactions have been developed. For example, Soinov et al. developed a supervised learning approach to the reconstruction of gene networks from expression data and applied their method to time-series data from yeast (32). Their method is based on building decision-tree-related classifiers, which predict gene expression from the expression data of other genes. Given a gene-expression matrix X_{ij} (where $i \rightarrow$ genes and $j \rightarrow$ experiments in a time course), their method deduces

- (i) the state of the gene i in sample j from the expression values of other genes in the same sample;
- (ii) the state of the gene i in sample j from the expression values of genes from the previous sample/samples; and
- (iii) the change in the state of the gene i from the changes in states of other genes.

The classifiers were represented as simple rules defining gene interrelations. Rules among 20 genes involved in the yeast cell cycle were considered. In most cases, the rules that were recovered (without any a priori knowledge or input to the method) are consistent with existing knowledge of cell cycle gene expression. In addition, previously unknown relationships were discovered, which can be treated as new hypotheses for subsequent experimental confirmation.

We (24), along with Friedman et al. (33), have developed techniques to perform context-specific clustering, which is a technique used to identify genes that are co-regulated in only a subset of the available data. While our more recent method for context-specific clustering has some advantages over that developed by Friedman's group, they took the results a step further to create machine-learning techniques that fit a combined probability model of expression patterns and TF binding sites to detect which binding sites best characterize a co-expression cluster. In addition, Friedman's group was the first to fit Bayesian networks to gene expression data (34, 35). In these network reconstructions, the nodes represent mRNA levels and the edges the probability that a parent mRNA level influences a child's mRNA level. There are several attractive features to Bayesian network approach. First, the networks provide predictive value – that is, the effect of changing the expression level of one gene on other genes is contained within the network. Second, the posterior probabilities obtained from fitting the networks to experimental data provide estimates of certainty in the predictions. Third and perhaps most importantly, the structure of the networks obtained correlates well with previous biological knowledge; that is, groups of genes that participate in similar functions are usually found in tightly connected “hubs.” Fourth, with appropriate data, it is possible to infer and fit causal relationships within the network; edges can be converted from non-directional correlations of gene expression to directional, causal events. Eric Schadt's group has adopted the Bayesian network approach of Friedman et al. to create causal network models that integrate both co-expression and genetic data (36–39).

4.2.3. Gene Expression and Genetics

Perhaps the most exciting area of research in which array data are leading directly to pathway inference is that of expression quantitative trait locus (eQTL) mapping (40–42). In this approach, the expression level of each gene is treated as a quantitative trait and genetic mapping is used to identify inherited loci that correlate with the expression level of each gene. This approach takes advantage of the fact that there are large numbers of differences in gene expression between individuals that are due to DNA polymorphisms, and uses genetics to identify the causative loci for each difference.

For example, in the work of Schadt et al., two standard inbred mouse strains, C57BL/6 J and DBA/2 J, were bred to produce 111 F₂s (second-generation offspring). Genome-wide genetic mapping was used to identify which alleles were inherited in each individual and gene expression measurements were performed on the liver tissue from each individual. In the parental strains, a significant fraction of the genome (7861/23,574 genes or 33%) is differentially expressed between the two strains. The F₂s are semi-random admixtures of the genomes of both strains. Through this combination of genetic mapping and expression profiling, it is possible to identify allelic variants of loci that correlate with increased or decreased expression of a gene. This approach is closely related to the perturbation approach outlined above in 4.2.2 (25), with the exception that the perturbations are accomplished through natural genetic mixing and the linear combination of multiple perturbations is present in each individual in the study. In effect, genetics is used to perturb the expression levels of many genes simultaneously, and genetic mapping is then used to correlate these effects with causative regions of the genome.

When eQTL mapping was applied to the C57BL/6 J x DBA/2 J cross, Schadt and colleagues found 4,339 eQTLs over 3,701 genes with LOD scores greater than 4.3, and more than 11,000 genes with at least one eQTL with an LOD score greater than 3.0 (LOD = log of the odds ratio and represents the probability that the locus influences the gene expression level by more than random chance). For many of these genes, the locus with the highest LOD score contained the gene itself. This is neither surprising nor particularly valuable, since one might expect that a polymorphism in the gene's promoter sequence would affect its expression level. In addition, polymorphisms in splice site junctions can also appear as differential expression depending on the region(s) of the gene that is interrogated on the array.

In similar work in yeast, Brem et al. crossed two strains [a lab strain (BY) and a wild strain isolated from a California vineyard, (RM)] to create 112 segregants. Each segregant was genotyped using Affymetrix arrays and, for each, expression analysis was performed using two-color ORF arrays. Initial analysis of this data set under a single locus model yielded eQTLs for 570 genes, each of which was linked to one or more different loci, with most expression levels showing complex inheritance patterns. Subsequent work by Brem et al. has re-analyzed this data under a more complex model to identify a large number of genetic interactions (43). At present, several research groups around the country are beginning to generate eQTL data predominately for human and mouse.

An interesting outcome of this work is that eQTL do not appear to be significantly enriched for the transcription factors of the linked genes; in retrospect, this is perhaps not too surprising.

For example, consider the following hypothetical biochemical pathway $A \rightarrow B \rightarrow C \rightarrow D \rightarrow E \rightarrow F$ in which A-F represent substrates that are operated on by enzymes (AtoB-ase, BtoC-ase, etc.) that convert each substrate to the next one down the line. If all five enzymes that are implied in this pathway were regulated by a single transcription factor, and if the activity of that transcription factor was in turn regulated by the concentration of the substrate F, then we would anticipate that mutations in any of the five enzymes would be detected as an eQTL for the others. In such a case, the mutation in the said enzyme would appear as “causal” for the altered expression level in the other enzymes in a reconstructed Bayesian network. However, in the actual biological pathway, the expression level or activity of the transcription factor is actually causal for the expression level of the enzyme mRNAs in the pathway.

5. Moving Forward

At present, Bayesian networks can be fit to expression data and, as discussed above, a variety of other data types (e.g., eQTL data) can be used to provide constraints that allow causal inferences to be made. *Unlike many other network reconstructions, Bayesian expression networks have predictive value.* That is, one can predict the changes in the mRNA expression level of other genes in the network, which will be caused by impacting the mRNA expression level of a target gene. Hence, these networks have tremendous practical value. For example, if one has a drug that reduces the expression level of a given gene, one can infer the effect of the drug on other genes from Bayesian network. However, while these reconstructed Bayesian network may be very predictive, as discussed briefly above, they may not accurately represent the underlying biological pathway. In addition, while these networks are strongly related to the underlying biochemical pathways, they are not “annotated” with the relevant physical and chemical interactions that are responsible for creating the data that were used to infer the pathways. So how do we move forward?

5.1. Using TF–Gene Interactions as Constraints to Bayesian Networks

As more and more data become available such that we can infer the direct targets of transcription factors, we need to use TF–gene regulation as constraints in the Bayesian networks. There are a number of data sets under development that will aid in better inference of TF–gene interactions. As discussed above, CHIP-on-chip data is providing a wealth of information regarding the binding of transcription factors to the given DNA targets. In addition, techniques have been developed within Marth Bulyk’s

lab to use dsDNA arrays to more carefully define the binding motifs of DNA-binding proteins (44–46) and her lab is working to characterize all the binding motifs of all known yeast transcription factors. It is important to note that CHIP-on-chip and DNA binding array data sets define the binding of TF factors to DNA and that this is a necessary but not sufficient condition for the regulation of nearby genes. Hence, these data sets need to be combined with other information to more accurately infer regulatory TF–gene interactions.

Toward this end, we are currently developing a probabilistic approach to predict direct gene targets for transcription factors (TFs), and prospectively validating our predictions in the lab. While there are a number of different data types that contain information related to gene regulation (expression, sequence, eQTL, CHIP-on-Chip, protein–protein interactions), each data type has its own biases and sources of noise. Significant progress toward the inference of gene regulation would require a sensible plan for integrating these various data types. Our predictive model incorporates information from multiple data types by systematically assigning weights to each data source such that relatively noisy data sources are assigned relatively low weights. Due to the availability of large expression, ChIP and eQTL datasets in yeast, we are testing our method in yeast. Our preliminary results show that leveraging multiple data types improves prediction accuracy in our cross-validation study.

5.2. Integration of Physical Interaction Networks with Bayesian Networks

As discussed in Chapter 10, we need to develop methods to search through measured and predicted physical interaction networks to identify possible “explanations” for given edges within the Bayesian networks. A simple example of the kind of explanatory physical interactions we might find is shown in **Fig. 11.4**. In this example, we show a transcription factor that has a physical interaction with a protein coded by “gene1”. If we imagine that the protein1 \leftrightarrow TF-1 interaction reduces the activity of TF-1, then we would anticipate that other genes (say genes 2, 3, and 4), which are also regulated by TF-1, would have mRNA levels that are correlated with the mRNA level for gene 1. Hence in a Bayesian

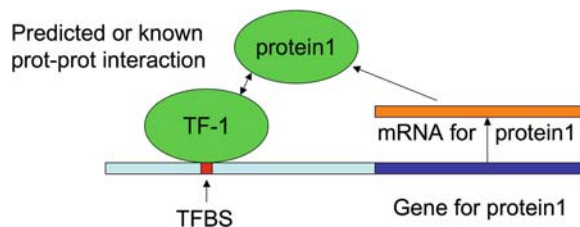


Fig. 11.4. A simple regulatory loop. TF-1 represents a transcription factor for Gene1. In this loop, the protein product of gene 1 binds to its own transcription factor.

network reconstruction we would anticipate that genes 1, 2, 3, and 4 are connected and that this loop within the physical interaction network could be used as annotation on the edges between gene 1 and other genes.

5.3. Network Reconstructions Involving Feedback Loops

While Bayesian network reconstruction is presently the most informative method for producing predictive networks, it is hampered by a fundamental flaw. By their very nature, Bayesian networks are directed *acyclic* graphs. Hence, standard Bayesian networks cannot represent the feedback loops that are so common in biology. In the long run, this important unsolved problem must be tackled as we go forward. There are several possible approaches to incorporating feedback into networks. One approach is to model expression using Dynamic Bayesian networks (DBNs). DBNs model the stochastic evolution of a set of random variables over time. In comparison with BNs, discrete time is introduced and conditional distributions for each variable are related to the values of parent variables in the previous time point. In this formalism, there is no need for the graph to be acyclic and hence feedback loops can be introduced. In the past 4 or so years, a number of groups (47–66) have been developing DBN methods for network inference from gene expression data and we anticipate that this methodology will come into more frequent use as larger data sets become available.

Two alternative approaches to modeling network dynamics involve directly modeling the chemical reactions through either solutions to differential equations (*see* Chapter 13) or stochastic simulation methods (*see* Chapter 14). As these methods are discussed in detail in other chapters in this volume, they will not be discussed in detail here. However, it is noted that both methods require both extensive knowledge of the likely chemical reactions present in the system of interest AND a fairly complete set of measurements on the relative concentrations of the players (proteins, substrates, etc.) in said reactions. In many cases, such extensive data are not generally available.

6. Summary

In this chapter we briefly discussed a number of approaches to network inference using large-scale functional genomics data. Our goal was to describe methods that can currently be used to infer networks that have predictive value. At present, one of the most effective methods to produce networks with predictive value is the Bayesian network approach as initially instantiated in this domain

by Friedman et al. (67) and further refined by Eric Schadt and his research group (36–39, 42, 68). This method has the virtue of identifying predictive relationships between genes from a combination of expression and eQTL data. However, it is lacking in providing a mechanistic bases for the predictive relationships and is ultimately hampered by an inability to model feedback.

In **Section 1**, a number of methods were described to identify or predict network components such as physical interactions. Ultimately, some of these approaches to identify network components could be integrated with and used as “annotation” for the edges in Bayesian networks. This would take the Bayesian networks one step closer to providing mechanistic “explanations” for the relationships between the nodes. In **Section 3**, methods to model network dynamics are discussed in detail. These methods typically have the benefit of an ability to model feedback loops. In addition, these methods often use the actual chemical reactions that produce the data as an underlying representation (and hence provide a strong mechanistic basis for the model). A challenge for the future is to produce networks that are both predictive and provide mechanistic understanding. To do so, methods to move more freely between and integrate the approaches described in **Sections 1, 2, and 3** will be required.

References

1. www.genome.ad.jp/kegg.
2. Kanehisa, M., S. Goto, S. Kawashima, and A. Nakaya, The KEGG databases at GenomeNet. *Nucleic Acids Res.*, 2002. 30(1): 42–6.
3. Chen, C.S. and H. Zhu, Protein microarrays. *Biotechniques*, 2006. 40: 423, 425, 427.
4. Ren, B., et al., Genome-wide location and function of DNA binding proteins. *Science*, 2000. 290: 2306–9.
5. Lee, T.I., et al., Transcriptional regulatory networks in *S. cerevisiae*. *Science*, 2002. 298: 799–804.
6. Serikawa, KA, X.L. Xu, V.L., MacKay, G.L. Law,, Q. Zong, L.P. Zhao, R.E. Bumgarner,, and D.R. Morris, The transcriptome and its translation during recovery from cell cycle arrest in *S. cerevisiae*. *Mol Cell Proteomics*, 2003. 2:191–204.
7. Ule, J., K. Jensen.K., A. Mele, and R.B. Darnell, CLIP: a method for identifying protein-RNA interaction sites in living cells. *Methods*, 2005. 37: 376–86.
8. Levy, S. and S. Hannenhalli, Identification of transcription factor binding sites in the human genome sequence. *Mamm Genome*, 2002. 13(9): 510–4.
9. Lohr, D., P. Venkov, and J. Zlatanova, Transcriptional regulation in the yeast GAL gene family: a complex genetic network. *Faseb J*, 1995. 9(9): 777–87.
10. Douglas, H.C. and D.C. Hawthorne, enzymatic expression and genetic linkage of genes controlling galactose utilization in *saccharomyces*. *Genetics*, 1964. 49: 837–44.
11. Lai, K. and L.J. Elsas, Overexpression of human UDP-glucose pyrophosphorylase rescues galactose-1-phosphate uridyl-transferase-deficient yeast. *Biochem Biophys Res Commun*, 2000. 271(2): 392–400.
12. Schwikowski, B., P. Uetz, and S. Fields, A network of protein-protein interactions in yeast. *Nat Biotechnol*, 2000. 18(12): 1257–61.
13. Sinha, S. and M. Tompa, YMF: A program for discovery of novel transcription factor binding sites by statistical overrepresentation. *Nucleic Acids Res*, 2003. 31(13): 3586–8.

14. Martone, R, E.G., P. Bertone, S. Hartman, T.E. Royce, N.M. Luscombe, J.L. Rinn, F.K. Nelson, P. Miller, M. Gerstein, S. Weissman, and M. Snyder, Distribution of NF-kappaB-binding sites across human chromosome 22. *Proc Natl Acad Sci*, 2003. 100: 12247–52.
15. Yeung, K.Y., M. Medvedovic, and R.E. Bumgarner, From co-expression to co-regulation: how many microarray experiments do we need? *Genome Biol*, 2004. 5(7): R48.
16. Spellman, P.T., G. Sherlock, M.Q. Zhang, V.R. Iyer, K. Anders, M.B. Eisen, P.O. Brown, D. Botstein, and B. Futcher, Comprehensive identification of cell cycle-regulated genes of the yeast *Saccharomyces cerevisiae* by microarray hybridization. *Mol Biol Cell*, 1998. 9(12): 3273–97.
17. Shah, N.H., D.C. King, P.N. Shah, and N.V. Fedoroff, A tool-kit for cDNA microarray and promoter analysis. *Bioinformatics*, 2003. 19(14): 1846–8.
18. Qiu, P., Recent advances in computational promoter analysis in understanding the transcriptional regulatory network. *Biochem Biophys Res Commun*, 2003. 309(3): 495–501.
19. Tavazoie, S., J.D. Huges, M.J. Campbell, R.J. Cho, and G.M. Church, Systematic determination of genetic network architecture. *Nat Genet*, 1999. 22: 281–5.
20. Wolfsberg, T.G., A.E. Gabrielian, M.J. Campbell, R.J. Cho, J.L. Spouge, and D. Landsman, Candidate regulatory sequence elements for cell cycle-dependent transcription in *Saccharomyces cerevisiae*. *Genome Res*, 1999. 9(8): 775–92.
21. Jelinsky, S.A., P. Estep, G.M. Church, and L.D. Samson, Regulatory networks revealed by transcriptional profiling of damaged *Saccharomyces cerevisiae* cells: Rpn4 links base excision repair with proteasomes. *Mol Cell Biol*, 2000. 20(21): 8157–67.
22. Ohler, U. and H. Niemann, Identification and analysis of eukaryotic promoters: recent computational approaches. *Trends Genet*, 2001. 17(2): 56–60.
23. Quackenbush, J., Genomics. Microarrays – guilt by association. *Science*, 2003. 302(5643): 240–1.
24. Liu, X., S. Sivaganesan, K.Y. Yeung, J. Guo, R.E. Bumgarner, and M. Medvedovic, Context-specific infinite mixtures for clustering gene expression profiles across diverse microarray dataset. *Bioinformatics*, 2006. e-pub.
25. Ideker, T., V. Thorsson, J.A. Ranish, R. Christmas, J. Buhler, J.K. Eng, R. Bumgarner, D.R. Goodlett, R. Aebersold, and L. Hood, Integrated genomic and proteomic analyses of a systematically perturbed metabolic network. *Science*, 2001. 292(5518): 929–34.
26. Ideker, T., O. Ozier, B. Schwikowski, and A.F. Siegel, Discovering regulatory and signalling circuits in molecular interaction networks. *Bioinformatics*, 2002. 18 Suppl 1: S233–40.
27. Shannon, P., A. Markiel, O. Ozier, N.S. Baliga, J.T. Wang, D. Ramage, N. Amin, B. Schwikowski, and T. Ideker, Cytoscape: a software environment for integrated models of biomolecular interaction networks. *Genome Res*, 2003. 13(11): 2498–504.
28. Han, J.D., N. Bertin, T. Hao, D.S. Goldberg, G.F. Berriz, L.V. Zhang, D. Dupuy, A.J. Walhout, M.E. Cusick, F.P. Roth, and M. Vidal, Evidence for dynamically organized modularity in the yeast protein-protein interaction network. *Nature*, 2004. 430(6995): 88–93.
29. Luscombe, N.M., M.M. Babu, H. Yu, M. Snyder, S.A. Teichmann, and M. Gerstein, Genomic analysis of regulatory network dynamics reveals large topological changes. *Nature*, 2004. 431(7006): 308–12.
30. Bumgarner, R. and M. Jones, Are we playing a "Kevin Bacon" game with microarray data? *Genome Technol*, 2001. 10: 104.
31. Yeang, C.H. and T.S. Jaakkola, Physical network models and multi-source data integration. in *The Seventh Annual International Conference on Research in Computational Molecular Biology*. 2003.
32. Soinov, L.A., M.A. Krestyaninova, and A. Brazma, Towards reconstruction of gene networks from expression data by supervised learning. *Genome Biol*, 2003. 4(1): R6.
33. Barash, Y and N. Friedman, Context-specific Bayesian clustering for gene expression data. *J Comput Biol*, 2002. 9: 169–91.
34. Friedman N, M. Linial I. Nachman, and D. Peer, Using Bayesian networks to analyze expression data. *J Comput Biol*, 2000. 7: 601–20.
35. Pe'er, D, A. Regev, G. Elidan, and N. Friedman, Inferring subnetworks from perturbed expression profiles. *Bioinformatics*, 2001. 17 Suppl 1: S215–24.
36. Schadt, E.E. and P.Y. Lum, Reverse engineering gene networks to identify key drivers of complex disease phenotypes. *J Lipid Res*, 2006. 47(12): 2601–13.

37. Ghazalpour, A., S. Doss, B. Zhang, S. Wang, C. Plaisier, R. Castellanos, A. Brozell, E.E. Schadt, T.A. Drake, A.J. Lusis, and S. Horvath, Integrating genetic and network analysis to characterize genes related to mouse weight. *PLoS Genet*, 2006. 2(8).
38. Schadt, E.E., Novel integrative genomics strategies to identify genes for complex traits. *Anim Genet*, 2006. 37 Suppl 1: 18–23.
39. Drake, T.A., E.E. Schadt, and A.J. Lusis, Integrating genetic and gene expression data: application to cardiovascular and metabolic traits in mice. *Mamm Genome*, 2006. 17(6): 466–79.
40. Jansen, R.C. and J.P. Nap, Genetical genomics: the added value from segregation. *Trends Genet*, 2001. 17(7): 388–91.
41. Brem, R.B., G. Yvert, R. Clinton, and L. Kruglyak, Genetic dissection of transcriptional regulation in budding yeast. *Science*, 2002. 296(5568): 752–5.
42. Schadt, E.E., S.A. Monks, T.A. Drake, A.J. Lusis, N. Che, V. Colinayo, T.G. Ruff, S.B. Milligan, J.R. Lamb, G. Cavet, P.S. Linsley, M. Mao, R.B. Stoughton, and S.H. Friend, Genetics of gene expression surveyed in maize, mouse and man. *Nature*, 2003. 422(6929): 297–302.
43. Brem, R.B. and L. Kruglyak, The landscape of genetic complexity across 5,700 gene expression traits in yeast. *Proc Natl Acad Sci U S A*, 2005. 102(5): 1572–7.
44. Bulyk, M.L., Analysis of sequence specificities of DNA-binding proteins with protein binding microarrays. *Methods Enzymol*, 2006. 410: 279–99.
45. Bulyk, M.L., DNA microarray technologies for measuring protein-DNA interactions. *Curr Opin Biotechnol*, 2006. 17(4): 422–30.
46. Mukherjee, S., M.F. Berger, G. Jona, X.S. Wang, D. Muzzey, M. Snyder, R.A. Young, and M.L. Bulyk, Rapid analysis of the DNA-binding specificities of transcription factors with DNA microarrays. *Nat Genet*, 2004. 36(12): 1331–9.
47. Geier, F., J. Timmer, and C. Fleck, Reconstructing gene-regulatory networks from time series, knock-out data, and prior knowledge. *BMC Syst Biol*, 2007. 1(1): 11.
48. Zhao, W., E. Serpedin, and E.R. Dougherty, Inferring gene regulatory networks from time series data using the minimum description length principle. *Bioinformatics*, 2006. 22(17): 2129–35.
49. Missal, K., M.A. Cross, and D. Drasdo, Gene network inference from incomplete expression data: transcriptional control of hematopoietic commitment. *Bioinformatics*, 2006. 22(6): 731–8.
50. Liebermeister, W. and E. Klipp, Bringing metabolic networks to life: integration of kinetic, metabolic, and proteomic data. *Theor Biol Med Model*, 2006. 3: 42.
51. Imoto, S., Y. Tamada, H. Araki, K. Yasuda, C.G. Print, S.D. Charnock-Jones, D. Sanders, C.J. Savoie, K. Tashiro, S. Kuhara, and S. Miyano, Computational strategy for discovering druggable gene networks from genome-wide RNA expression profiles. *Pac Symp Biocomput*, 2006: 11, 559–71.
52. Huang, Y., D. Liu, and H. Wu, Hierarchical Bayesian methods for estimation of parameters in a longitudinal HIV dynamic system. *Biometrics*, 2006. 62(2): 413–23.
53. Gupta, R., P. Auvinen, A. Thomas, and E. Arjas, Bayesian hierarchical model for correcting signal saturation in microarrays using pixel intensities. *Stat Appl Genet Mol Biol*, 2006. 5: Article20.
54. Golightly, A. and D.J. Wilkinson, Bayesian sequential inference for stochastic kinetic biochemical network models. *J Comput Biol*, 2006. 13(3): 838–51.
55. Dojer, N., A. Gambin, A. Mizera, B. Wilczynski, and J. Tiuryn, Applying dynamic Bayesian networks to perturbed gene expression data. *BMC Bioinformatics*, 2006. 7: 249.
56. Zou, M. and S.D. Conzen, A new dynamic Bayesian network (DBN) approach for identifying gene regulatory networks from time course microarray data. *Bioinformatics*, 2005. 21(1): 71–9.
57. Yoshida, R., S. Imoto, and T. Higuchi, Estimating time-dependent gene networks from time series microarray data by dynamic linear models with Markov switching. *Proc IEEE Comput Syst Bioinform Conf*, 2005: 289–98.
58. Liebermeister, W. and E. Klipp, Biochemical networks with uncertain parameters. *Syst Biol (Stevenage)*, 2005. 152(3): 97–107.
59. Liang, Y., B. Tayo, X. Cai, and A. Kelemen, Differential and trajectory methods for time course gene expression data. *Bioinformatics*, 2005. 21(13): 3009–16.
60. Ferrazzi, F., P. Magni, and R. Bellazzi, Random walk models for bayesian clustering of gene expression profiles. *Appl Bioinformatics*, 2005. 4(4): 263–76.
61. Di, Camillo, B., F. Sanchez-Cabo, G. Toffolo, S.K. Nair, Z. Trajanoski, and C. Cobelli, A quantization method based on threshold optimization for

- microarray short time series. *BMC Bioinformatics*, 2005. 6 Suppl 4: S11.
62. Beal, M.J., F. Falciani, Z. Ghahramani, C. Rangel, and D.L. Wild, A Bayesian approach to reconstructing genetic regulatory networks with hidden factors. *Bioinformatics*, 2005. 21(3): 349–56.
 63. Yu, J., V.A. Smith, P.P. Wang, A.J. Hartemink, and E.D. Jarvis, Advances to Bayesian network inference for generating causal networks from observational biological data. *Bioinformatics*, 2004. 20(18): 3594–603.
 64. Wu, F.X., W.J. Zhang, and A.J. Kusalik, Modeling gene expression from microarray expression data with state-space equations. *Pac Symp Biocomput*, 2004: 581–92.
 65. Wang, S.C., Reconstructing genetic networks from time ordered gene expression data using Bayesian method with global search algorithm. *J Bioinform Comput Biol*, 2004. 2(3): 441–58.
 66. Sugimoto, N. and H. Iba, Inference of gene regulatory networks by means of dynamic differential Bayesian networks and nonparametric regression. *Genome Inform*, 2004. 15(2): 121–30.
 67. Pe'er, D., A. Regev, G. Elidan, and N. Friedman, Inferring subnetworks from perturbed expression profiles. *Bioinformatics*, 2001. 17 Suppl 1: S215–24.
 68. GuhaThakurta, D., T. Xie, M. Anand, S.W. Edwards, G. Li, S.S. Wang, and E.E. Schadt, Cis-regulatory variations: a study of SNPs around genes showing cis-linkage in segregating mouse populations. *BMC Genomics*, 2006. 7: 235.

Chapter 12

Exploring Pathways from Gene Co-expression to Network Dynamics

Huai Li, Yu Sun, and Ming Zhan

Abstract

One of the major challenges in post-genomic research is to understand how physiological and pathological phenotypes arise from the networks or connectivity of expressed genes. In addressing this issue, we have developed two computational algorithms, CoExMiner and PathwayPro, to explore static features of gene co-expression and dynamic behaviors of gene networks. CoExMiner is based on B-spline approximation followed by the coefficient of determination (CoD) estimation for modeling gene co-expression patterns. The algorithm allows the exploration of transcriptional responses that involve coordinated expression of genes encoding proteins which work in concert in the cell. PathwayPro is based on a finite-state Markov chain model for mimicking dynamic behaviors of a transcriptional network. The algorithm allows quantitative assessment of a wide range of network responses, including susceptibility to disease, potential usefulness of a given drug, and consequences of such external stimuli as pharmacological interventions or caloric restriction. We demonstrated the applications of CoExMiner and PathwayPro by examining gene expression profiles of ligands and receptors in cancerous and non-cancerous cells and network dynamics of the leukemia-associated BCR–ABL pathway. The examinations disclosed both linear and nonlinear relationships of ligand–receptor interactions associated with cancer development, identified disease and drug targets of leukemia, and provided new insights into biology of the diseases. The analysis using these newly developed algorithms show the great usefulness of computational systems biology approaches for biological and medical research.

Key words: Systems biology, co-expression, pathway dynamics, network modeling, coefficient of determination (CoD), Markov chain, transcriptional intervention.

1. Introduction

One of the major challenges in post-genomic research and computational systems biology is to understand how physiological and pathological phenotypes arise from the networks or connectivity of

expressed genes (1, 2). The utilization of high-throughput data generated by microarrays and other methodologies provides scientists with a first step toward the goal of system-level analyses of biological networks (3). Systems biology has shown promise in many areas of biology, particularly for identifying diagnostic biomarkers and drug-affected genes or drug targets (4, 5). In this chapter, we identify two cruxes in the study of biological networks, i.e., static features of gene co-expression and dynamic behaviors of networks, and describe how to decipher network or pathway information using computational systems biology approaches based on gene expression data.

1.1. Gene Co-expression

The study of gene co-expression allows the discovery of transcriptional responses that involve coordinated expression of genes which likely work in concert in the cell. With recent interest in biological networks, the use of gene co-expression measured across large number of experiments has emerged as a novel holistic approach for microarray data analysis (6–9). Typically, the metric of co-expression that has been used is Pearson's correlation coefficient (6, 7, 10, 11). This linear-model-based correlation coefficient provides a good first approximation of co-expression, but is also associated with certain pitfalls. When the relationship between log-expression levels of two genes is nonlinear, the degree of co-expression is underestimated (12). Since the correlation coefficient is a symmetrical measurement, it cannot provide evidence of a directional relationship in which one gene is upstream of another (13). Similarly, mutual information is also not suitable for modeling directional relationship, although it has been applied in various co-expression studies (11, 14).

Recently, we proposed a new algorithm, CoExMiner, which provides a more biologically meaningful and comprehensive model for gene co-expression, functional relationships, and network structure (15). The new algorithm is based on B-spline approximation followed by CoD estimation. The algorithm is capable of uncovering both linear and nonlinear relationships of co-expression and suggesting the directionality. It is thus particularly useful in the prediction analysis of gene expression, the determination of connectivity in a pathway, and network inference. The computation by the new algorithm requires no quantization of microarray data, thus avoiding significant loss or misrepresentation of biological information, which would otherwise occur in the conventional application of CoD (16, 17). In this chapter, we describe the basics of CoExMiner algorithm. We also show the application of the algorithm in modeling the co-expression patterns and exploring biological information from the microarray data of different cancers. The algorithm allowed the correct identification of co-expressed ligand–receptor pairs specific to cancerous tissues and provided new insight into the understanding of cancer development.

1.2. Network Dynamics

Biological networks or pathways behave only under controlled manners in response to disease development, changing cellular conditions, or external stimuli (18). By characterizing the dynamic behavior of biological pathways, we aim to identify how disease or cellular phenotypes arise from the connectivity or networks of genes and their products. Various algorithms have been employed in examining the dynamic behaviors of biological networks in silico, including the Markov chain (19) and probabilistic Boolean network (20). In silico simulation has been particularly important in network analysis since network activity is constrained by various complex forms of interactions (21, 22).

Recently, we developed a new algorithm, PathwayPro, to mimic the complex behavior of a biological pathway through a series of perturbations made in silico to each gene or gene combination (23). The inputs to the algorithm are the topologies of pathways and gene expression data. The outputs are the estimated probabilities of network transition across different cellular conditions under each transcriptional perturbation. The algorithm can provide answers to two questions. First, whether or how much a gene or external perturbation contributes to the dynamic behavior of a pathway in instances such as disease development or recovery, aging processes, and cell differentiation. Second, in what specific ways is this contribution manifested. PathwayPro analysis is particularly valuable in its ability to simulate in silico pathway behaviors that may not be easy to create in vitro. The hypotheses subsequently derived can then be tested via independent experiments. The analysis thus facilitates the development of systematic approaches to effective preventive and therapeutic intervention in disease. The potential clinical impact of such analysis is tremendous as the type of intervention analysis not only open up a window on the biological behavior of an organism and the disease progression but also translate into accurate diagnosis, target identification, drug development, and treatment. We demonstrate the application of PathwayPro by analyzing the leukemia-related BCR-ABL proteins and the pathway. The analysis correctly identified drug targets for leukemia and shed light on the understanding of the disease.

2. Basics of Algorithms

2.1. Computational Model for Gene Co-expression of Mixed Patterns

The algorithm we present here is based on CoExMiner (15). The algorithm uses a B-spline approximation to predict the expression value of the target gene g_y using the predictor gene g_x , followed by CoD estimation of co-expression of g_x and g_y . The algorithm allows measurement of both linear and nonlinear patterns and directionality of co-expression.

2.1.1. B-Spline Approximation

The B-spline is a set of piecewise polynomial functions (24). It can be defined as follows:

$$\bar{\mathbf{P}}(t) = \sum_{j=1}^{n+1} B_{j,k}(t) \bar{\mathbf{P}}_j, \quad t_{\min} \leq t < t_{\max} \quad [1]$$

In Eq. [1], $\bar{\mathbf{P}}_1, \bar{\mathbf{P}}_2, \dots, \bar{\mathbf{P}}_{n+1}$ are $n+1$ control points. The $B_{j,k}$ basis function is of order k . k must be at least 2, and can be no more than $n+1$. Equation [1] defines a piecewise continuous function. A knot vector, $t_1, t_2, \dots, t_{k+(n+1)}$, must be specified for a given number of control points $n+1$ and B-spline order k . It is necessary that $t_j \leq t_{j+1}, \forall j$. The basis function $B_{j,k}$ depends only on the value of k and the values in the knot vector. $B_{j,k}$ is defined recursively as

$$B_{j,1}(t) = \begin{cases} 1, & t_j \leq t < t_{j+1} \\ 0, & \text{otherwise} \end{cases} \quad [2]$$

$$B_{j,k}(t) = \frac{t - t_j}{t_{j+k-1} - t_j} B_{j,k-1}(t) + \frac{t_{j+k} - t}{t_{j+k} - t_{j+1}} B_{j+1,k-1}(t)$$

By viewing the co-expression pattern as a two-dimensional scatter plot for a given pair of genes g_x and g_y with expression values $\{(x_i, y_i), i = 1, \dots, N\}$, the plot pattern can be modeled by Eq. [1]. To construct a 2D B-spline curve requires that $\bar{\mathbf{P}}(t)$ and $\bar{\mathbf{P}}_j$ in Eq. [1] are written as $\bar{\mathbf{P}}(t) = \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} f(t) \\ g(t) \end{pmatrix}$ and $\bar{\mathbf{P}}_j = \begin{pmatrix} \tilde{x}_j \\ \tilde{y}_j \end{pmatrix}$. Here $f(t)$ and $g(t)$ are the x and y components of a point on the curve. $\{(\tilde{x}_j, \tilde{y}_j), j = 1, \dots, n+1\}$ are the control points selected from $\{(x_i, y_i), i = 1, \dots, N\}$ where $n+1 \leq N$.

2.1.2. CoD Estimation

CoD is the ratio of the explained variation to the total variation and denotes the strength of the association between predictor genes and target gene. Specifically, for any feature set X , CoD relative to the target variable Y is defined as $\text{CoD}_{X \rightarrow Y} = \frac{\varepsilon_0 - \varepsilon_X}{\varepsilon_0}$, where ε_0 is the prediction error in the absence of predictor and ε_X is the error for the optimal predictors (16). For the purpose of exploring co-expression patterns, we consider only a pair of genes g_x and g_y , where g_y is the target gene that is predicted by the predictor gene g_x . The errors are estimated based on the available samples for simplicity.

In specific, given a pair of genes g_x and g_y with expression values x_i and y_i , $i = 1, \dots, N$, where N is the number of samples, CoD can be computed according to the definition.

$$\text{CoD}_{g_x \rightarrow g_y} = \frac{\varepsilon_0 - \varepsilon_X}{\varepsilon_0} = \frac{\sum_{i=1}^N (y_i - \bar{y})^2 - \sum_{i=1}^N (y_i - \hat{y}_i)^2}{\sum_{i=1}^N (y_i - \bar{y})^2} \quad [3]$$

The key point for computing CoD from **Eq. [3]** is to find the optimal estimator \hat{y}_i from continuous values of data samples (x_i, y_i) . Motivated by the spirit of B-spline, an algorithm is formulated to estimate the CoD from continuous data of gene expression. The proposed algorithm is summarized as follows.

Input

- A pair of genes g_x and g_y with expression values x_i and y_i , $i = 1, \dots, N$. N is the number of samples.
- M intervals of control points. By given N and M , the number of control points ($n+1$) is determined as $n = \lfloor \frac{N}{M} \rfloor$, where $\lfloor \cdot \rfloor$ is the floor function.
- Spline order k .

Output

- CoD of gene g_y predicted by gene g_x .

Algorithm

- Fit B-spline curve $\bar{\mathbf{P}}(t) = \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} f(t) \\ g(t) \end{pmatrix}$ based on control points $\bar{\mathbf{P}}_1, \bar{\mathbf{P}}_2, \dots, \bar{\mathbf{P}}_{n+1}$, a knot vector, $t_1, t_2, \dots, t_{k+(n+1)}$, and the order of k .
 - Find indices of $\left\{ \begin{pmatrix} x'_i \\ y'_i \end{pmatrix}, i=1, \dots, N \right\}$, where $(x'_1 \leq x'_2 \leq \dots \leq x'_N)$ are ordered as monotonic increasing based on (x_1, x_2, \dots, x_N) , y'_i is the value with the same index as x'_i .
 - Assign ($n+1$) control points as:

$$\bar{\mathbf{P}}_j = \left\{ \begin{pmatrix} \tilde{x}_j \\ \tilde{y}_j \end{pmatrix} = \begin{pmatrix} x'_{1+(j-1) \times M} \\ y'_{1+(j-1) \times M} \end{pmatrix}, j = 1, \dots, n \right\} \text{ and}$$

$$\bar{\mathbf{P}}_{n+1} = \left\{ \begin{pmatrix} \tilde{x}_{n+1} \\ \tilde{y}_{n+1} \end{pmatrix} = \begin{pmatrix} x'_N \\ y'_N \end{pmatrix} \right\}.$$
 - Compute the $B_{j,k}(t)$ basis functions recursively from **Eq. [2]**.
 - Formulate $\bar{\mathbf{P}}(t) = \sum_{j=1}^{n+1} B_{j,k}(t) \begin{pmatrix} \tilde{x}_j \\ \tilde{y}_j \end{pmatrix}$ based on **Eq. [1]**.
- Calculate CoD of gene g_y predicted by gene g_x
 - Compute mean expression value of g_y without predictors according to $\bar{y} = \frac{\sum_{i=1}^N y_i}{N}$.
 - For $i = 1, \dots, N$, find $\hat{y}'_i = F(x'_i)$ by eliminating t between $x = f(t)$ and $y = g(t)$. First find $t_i = \arg \left\{ \min_t |f(t) - x'_i| \right\}$. Then compute $\hat{y}'_i = g(t_i)$.
 - Calculate CoD from **Eq. [3]** based on the ordered $\left\{ \begin{pmatrix} x'_i \\ y'_i \end{pmatrix}, i = 1, \dots, N \right\}$. Refer to **Eq. [3]**, CoD value is the same as calculated based on $\left\{ \begin{pmatrix} x_i \\ y_i \end{pmatrix}, i = 1, \dots, N \right\}$.

Including the special cases, we have (i) $\varepsilon_0 > 0$, if $\varepsilon_0 \geq \varepsilon_X$, compute CoD from **Eq. [3]**; else set CoD to 0. (ii) $\varepsilon_0 = 0$, if $\varepsilon_X = 0$, set CoD to 1; else set CoD to 0.

2.1.3. Statistical Significance

For a given CoD value estimated on the basis of B-spline approximation (referred as CoD-B in the following), the probability of obtaining a larger CoD-B by randomly shuffling one of the expression profiles (P_{shuffle}) is calculated by Monte Carlo simulation. In the simulation, random datasets can be created by shuffling the expression profiles of the predictor gene A and the target gene B, and then CoD-B is determined based on the random dataset. P_{shuffle} of CoD-B from the real data could be determined according to the derived probability distribution of CoD-B from the simulation.

2.2. Computational Model for Pathway Dynamics

The algorithm we present here is based on PathwayPro (23). In this algorithm, a finite-state Markov chain model is constructed with the gene expression profile and network topology. The probability of network transition is determined based on state-dependent multivariate conditional probabilities between gene expression levels.

2.2.1. Model Construction

The proposed computational model contains n selected genes. Each gene has a ternary expression value, which is assigned as either over-expressed (1), equivalently-expressed (0), or under-expressed (-1), depending whether the expression level is significantly lower than, similar to, or greater than the respective control threshold. For capturing the dynamics of the network, we use the state of predictor genes at step t and the corresponding conditional probabilities, which are estimated from observed data, to derive the state of the target gene at step $t + 1$. **Equation [4]** shows the definition of transition between gene states at step t and the state at step $t + 1$, which can be represented as a Markov chain (19).

$$S^{(t)} =: (\mathcal{G}_1^{(t)} \mathcal{G}_2^{(t)} \dots \mathcal{G}_n^{(t)}) \longrightarrow S^{(t+1)} =: (\mathcal{G}_1^{(t+1)} \mathcal{G}_2^{(t+1)} \dots \mathcal{G}_n^{(t+1)}) \quad [4]$$

Here, we generalize the model which allows any number of predictor genes for each target gene based on the topology of the network. If the network topology shows there are no predictors as inputs to predict a gene in the next step, the current gene value is kept. The transition rule for $S^{(t)} \rightarrow S^{(t+1)}$ is depicted in **Fig. 12.1** and characterized by **Eq. [5]**.

$$\mathcal{G}_i^{(t+1)} = \begin{cases} -1: & \text{with } C_i^{-1}(\mathcal{G}_{i_1}^{(t)} \mathcal{G}_{i_2}^{(t)} \dots \mathcal{G}_{i_k}^{(t)}) = p(\mathcal{G}_i^{(t+1)} = -1 | \mathcal{G}_{i_1}^{(t)} \mathcal{G}_{i_2}^{(t)} \dots \mathcal{G}_{i_k}^{(t)}) \\ 0: & \text{with } C_i^0(\mathcal{G}_{i_1}^{(t)} \mathcal{G}_{i_2}^{(t)} \dots \mathcal{G}_{i_k}^{(t)}) = p(\mathcal{G}_i^{(t+1)} = 0 | \mathcal{G}_{i_1}^{(t)} \mathcal{G}_{i_2}^{(t)} \dots \mathcal{G}_{i_k}^{(t)}) \\ 1: & \text{with } C_i^1(\mathcal{G}_{i_1}^{(t)} \mathcal{G}_{i_2}^{(t)} \dots \mathcal{G}_{i_k}^{(t)}) = p(\mathcal{G}_i^{(t+1)} = 1 | \mathcal{G}_{i_1}^{(t)} \mathcal{G}_{i_2}^{(t)} \dots \mathcal{G}_{i_k}^{(t)}) \end{cases} \quad [5]$$

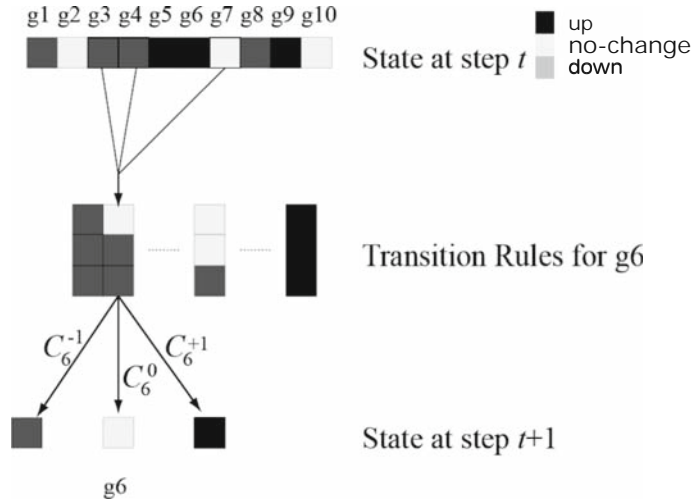


Fig. 12.1. Illustration of transition rules for target genes in the Markov chain model. In this example, target gene g_6 has three predictor genes g_3 , g_4 , and g_7 . The value of g_6 at step $(t+1)$ is determined by the conditional probabilities under the condition $g_3 = 0$, $g_4 = -1$, and $g_7 = -1$ at step (t) .

where $i_1, i_2, \dots, i_k, l \in \{1, 2, \dots, n\}$ and k is the number of predictor genes. C_l^{-1} , C_l^0 , and C_l^1 are conditional probabilities that depend on the states of the predictor genes and satisfy $C_l^{-1} + C_l^0 + C_l^1 = 1$ in **Eq. [5]**. For example, if there are three predictor genes for a target gene with a ternary value, there are $3^3 = 27$ possible states observable. The conditional probabilities C_l^{-1} , C_l^0 , and C_l^1 are estimated from the data. Since the number of experiments (data) in microarray studies is often limited, there may be some states not observed in the data. In such case, we assign $\Pr(g_l = -1)$, $\Pr(g_l = 0)$, and $\Pr(g_l = 1)$ for C_l^{-1} , C_l^0 , and C_l^1 , respectively. Based on the transition rule, we can compute the transition probability between any two arbitrary states of the Markov chain as follows:

$$\Pr\{S^{(t)} \rightarrow S^{(t+1)}\} = \prod_{l=1}^n C_l^{g_l^{(t+1)}} \quad [6]$$

In the simulation, a small but sufficient perturbation is added to guarantee a steady-state distribution exists and the chain converges to the steady-state distribution. With a perturbation, the entire Markov chain is ergodic and every state will eventually be visited. Considering gene perturbation, the transition probability **Eq. [6]** can be generalized as (19) **Eq. [7]**:

$$\begin{aligned} \Pr\{S^{(t)} \rightarrow S^{(t+1)}\} &= \left(\prod_{l=1}^n C_l^{g_l^{(t+1)}} \right) \times (1-p)^n \\ &+ p^{n_0} (1-p)^{n-n_0} p_0^{n_0} \times \mathbf{1}_{[S^{(t)} \neq S^{(t+1)}]} \end{aligned} \quad [7]$$

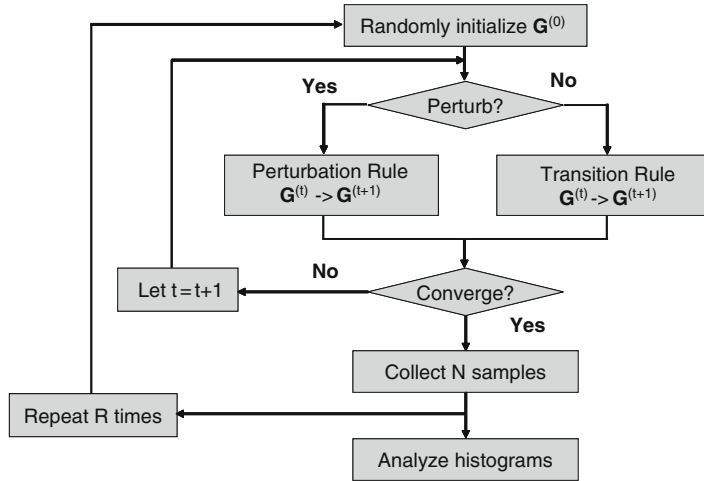


Fig. 12.2. Simulation algorithm for steady-state analysis. The algorithm starts from a random initial state and repeats R times before collecting samples from steady-state distribution. In the simulation, a small but measurable perturbation is added to guarantee a steady-state distribution exists and the chain converges to the steady-state distribution.

where p is the perturbation probability for each gene, $n_0 = \sum_{l=1}^n \mathbf{1}_{[g_l^{(t)} \neq g_l^{(t+1)}]}$ is the number of genes to be perturbed, and $p_0 = 1/(q-1)$. In the ternary case, $q = 3$, so p_0 is equal to 0.5. The simulation algorithm used in this study is summarized in Fig. 12.2.

2.2.2. Intervention Analysis by Markov Chain Model

The ability of the current model to enhance our understanding of biological systems should be further investigated by exploring another common biological system feature, the ability to readily switch from one relatively stable state to another in response to a simple stimulus. To a certain extent, this study can also verify how well the model mimics biological systems. Basically, one question may be interesting to ask: Given a desired target state and an initial state, with which genes in network should we intervene by simultaneously flipping their status so that the probability that the network will reach the desired target state is greatest? We could address this question by finding the best candidate genes for intervention based on first-passage time (20, 25). The first-passage time provides a natural way to capture the goals of intervention in the sense that we wish to transit to certain states (or avoid certain states) as quickly as possible, or, alternatively, by maximizing the probability of reaching such states before a certain time. So it can be used as a tool for deciding which genes are the best candidates for intervention. The first passage time from state x to state y can be defined as follows: with the probability $F_k(x, y)$ that, starting in state x , the first time the network reach a given state y will be at step

k . It is easy to see that for $k = 1$, $F_1(x, y) = A(x, y)$, which is just the transition probability from x to y . For $k \geq 2$, $F_k(x, y)$ satisfies (25)

$$F_k(x, y) = \sum_{z \in [-1, 0, 1]^n - \{y\}} A(x, z) F_{k-1}(z, y) \quad [8]$$

In Eq. [8], each element $A(x, y)$ of the transition matrix A can be computed using Eq. [7]. For a fixed K , a $3^n \times K$ matrix F can be created in which each column contains the probability $F_k(x, y)$ from all possible starting states x to a given target state y at k steps. We can then use $H_K(x, y) = \sum_{k=1}^K F_k(x, y)$ as a measurement index. Because the events that the first passage time from x to y will be at step k are disjoint for different k , the sum of their probabilities for $k = 1, \dots, K$ is equal to the probability that the network, starting in state x , will visit state y before step K . Since the chain is ergodic with perturbation probability p , when $K = \infty$, $H_\infty(x, y)$ is equal to the probability that the chain ever visits the state y , which is equal to 1.

Using the above measurement tool, we construct the intervention information matrix H at a fixed $K = 3$. In this matrix, each row $H_3(x, \cdot)$ represents the probability that the network, from a starting state x , will visit all desired ending states before step $K = 3$. Each column $H_3(\cdot, y)$ represents the probability that the network, starting in all possible intervened states, will visit state y before step $K = 3$. For simulating simple stimuli, we mathematically change the expression level of one gene, two genes, and three genes each time and keep the rest of the genes unchanged for a starting state x . For a ternary expression, that will generate $C_n^3 \times 3^3$ intervened states for changing one, two, and three genes which include the original state x .

3. Biological Applications

We implemented both CoExMiner and PathwayPro algorithms in Java-based interactive computational tools (15, 23). With the software tools, we analyzed ligand and receptor expression profiles in cancerous and normal tissues, and examined the leukemia-related ABL-BCR pathway.

3.1. Co-expression of Ligand–Receptor Pairs

In this study, we used CoExMiner to analyze the co-expression of ligands and their corresponding receptors in dissected tissues of lung cancer, prostate cancer, acute myeloid leukemia (AML), and their normal tissue counterparts. The ligand–receptor cognate pair data were obtained from the Database of Ligand–Receptor

Partners (DLRP) (10). The gene expression data are downloaded from the GEO database (accession numbers GSE1987, GSE1431, and GSE995, respectively). The array data, initially obtained using Affymetrix microarrays, were normalized by the Robust Multi-Array Analysis (RMA) method (26).

Significantly co-expressed ligand and receptor pairs were identified in the cancer and normal tissue groups at thresholds of R^2 and CoD-B of 0.50 and P_{shuffle} of 0.05. From these, differentially co-expressed pairs between cancerous and normal tissues were selected. **Table 12.1** lists the differentially co-expressed genes between cancerous and normal tissues. About 12 ligand–receptor pairs were differentially co-expressed between lung cancer and normal tissues (CoD-B difference >0.40 , **Table 12.1A**). The

Table 12.1
List of ligand–receptor pairs which showed differential co-expression between cancers and normal tissue. (A) Lung cancer; (B) Prostate cancer; (C) Acute myeloid leukemia (AML)

Ligand	Receptor	CoD-B		P_{shuffle}	
		Cancer	Normal	Cancer	Normal
(A) Lung cancer					
<u>BMP7</u>	<u>ACVR2B</u>	0.76	0.00	0.028	0.58
<u>EFNA3</u>	<u>EPHA5</u>	0.84	0.00	6.7E-06	0.69
<u>FGF8</u>	<u>FGFR2</u>	0.55	0.00	1.5E-07	0.66
<u>IL16</u>	<u>CD4</u>	0.62	0.031	2.7E-06	0.68
<u>CCL23</u>	<u>CCR1</u>	0.00	0.85	0.73	2.1E-09
<u>IL1RN</u>	<u>IL1R1</u>	0.23	0.83	0.077	8.4E-07
<u>IL18</u>	<u>IL18R1</u>	0.18	0.71	0.097	4.5E-06
<u>IL13</u>	<u>IL13RA2</u>	0.00	0.69	0.62	1.5E-04
<u>BMP5</u>	<u>BMPR2</u>	0.00	0.61	0.69	1.7E-04
(B) Prostate cancer					
<u>BMP6</u>	<u>ACVR2B</u>	0.63	0.081	0.0011	0.44
<u>BTC</u>	<u>EGFR</u>	0.75	0.00	1.7E-11	0.28
<u>TGFB2</u>	<u>TGFBR2</u>	0.79	0.00	3.5E-04	0.49
<u>INHA</u>	<u>ACVR2A</u>	0.59	0.019	1.1E-06	0.45
<u>CCL23</u>	<u>CCR1</u>	0.00	0.85	0.43	3.2E-09

(continued)

Table 12.1 (continued)

Ligand	Receptor	CoD-B		P _{shuffle}	
		Cancer	Normal	Cancer	Normal
<u>IL1RN</u>	<u>IL1R1</u>	0.00	0.82	0.32	3.1E-07
<u>TNFSF8</u>	<u>TNFRSF8</u>	0.00	0.76	0.36	1.5E-06
<u>IL18</u>	<u>IL18R1</u>	0.00	0.70	0.39	2.1E-07
<u>FIGF</u>	<u>KDR</u>	0.00	0.57	0.26	0.0023
<u>CXCL5</u>	<u>IL8RB</u>	0.00	0.58	0.41	1.1E-04
(C) Acute myeloid leukemia					
<u>FASLG</u>	<u>FAS</u>	0.90	0.14	3.6E-05	0.34
<u>BMP7</u>	<u>BMPRI1B</u>	0.82	0.00	7.7E-04	0.59
<u>EFNA5</u>	<u>EPHA1</u>	0.85	0.00	2.5E-04	0.71
<u>FGF3</u>	<u>FGFR2</u>	0.81	0.00	7.4E-06	0.66
<u>FGF13</u>	<u>FGFR4</u>	0.75	0.059	0.0097	0.47
<u>NRG1</u>	<u>ERBB3</u>	0.95	0.00	1.7E-05	0.28
<u>CCL4</u>	<u>CCBP2</u>	0.99	0.24	9.6E-06	0.062
<u>CCL7</u>	<u>CCR5</u>	0.97	0.29	0.00476	0.41
<u>IFNA8</u>	<u>IFNAR2</u>	0.88	0.00	2.9E-05	0.70
<u>IFNG</u>	<u>IFNGR1</u>	0.87	0.00	3.4E-04	0.68
<u>IL13</u>	<u>IL4R</u>	0.82	0.00	0.0041	0.70
<u>INHBB</u>	<u>ACVR2B</u>	0.82	0.23	1.5E-04	0.11
<u>AMH</u>	<u>AMHR2</u>	0.00	0.78	0.63	4.7E-05
<u>CD40LG</u>	<u>CD40</u>	0.00	0.97	0.33	8.6E-05
<u>TNFSF7</u>	<u>TNFRSF7</u>	0.39	0.97	0.043	8.2E-05
<u>EFNA1</u>	<u>EPHA4</u>	0.065	0.86	0.59	1.6E-06
<u>FGF1</u>	<u>FGFR4</u>	0.00	0.93	0.32	1.6E-06
<u>CXCL2</u>	<u>IL8RB</u>	0.25	0.84	0.33	3.3E-06
<u>FGF17</u>	<u>FGFR3</u>	0.17	0.70	0.17	3.0E-04
<u>DLK1</u>	<u>NOTCH4</u>	0.00	0.89	0.55	2.5E-07
<u>TNFSF4</u>	<u>TNFRSF4</u>	0.00	0.92	0.67	3.3E-04
<u>CXCL9</u>	<u>CXCR3</u>	0.30	0.98	0.054	1.5E-04
<u>TGFB1</u>	<u>TGFBR1</u>	0.00	0.71	0.62	6.8E-05

ligand BMP7 (bone morphogenetic protein 7), related to cancer development (27, 28), was one of the differentially co-expressed genes. For BMP7 and its receptor ACVR2B (activin receptor IIB), the CoD-B was 0.76 ($P_{\text{shuffle}} < 0.028$) in the lung cancer and 0.00 ($P_{\text{shuffle}} < 0.58$) in the normal tissue, and the R^2 value was 0.042 (cancer) or 0.0012 (normal tissue) (Table 12.1A). Therefore, BMP7 and ACVR2B show a nonlinear co-expression in the lung cancer but no co-expression in the normal tissue. The co-expression profile (Fig. 12.3A) further showed that the two genes displayed approximately the nonlinear pattern (piecewise pattern) of co-expression, and BMP7 was over-expressed in the lung cancer as compared with the normal tissue. These results are suggestive of a certain level of negative feedback involved in the interaction between BMP7 and ACVR2B.

The ligand CCL23 (chemokine ligand 23) and its receptor CCR1 (chemokine receptor 1), on the other hand, exhibited a high linear co-expression in the normal lung tissue but no co-expression in cancerous lung samples. As shown in Table 12.1A, the CoD-B value of the gene pair was 0.85 in the normal tissue while 0.00 in the lung cancer, and the R^2 value was 0.91 in the normal tissue and 0.054 in the lung cancer, suggesting a linear co-expression between the genes. The linear co-expression pattern is further profiled in Fig. 12.3B. Similarly, CCL23 and CCR1 were also highly co-expressed in the normal prostate samples (CoD-B = 0.85) but not co-expressed in the cancerous prostate samples (CoD-B = 0.0) (Table 12.1B). However, CCL23 and CCR1 were not co-expressed in either normal (CoD-B = 0.0) or AML samples (CoD-B = 0.0). The results suggest that CCL23 and CCR1 show differential co-expression not only between cancerous

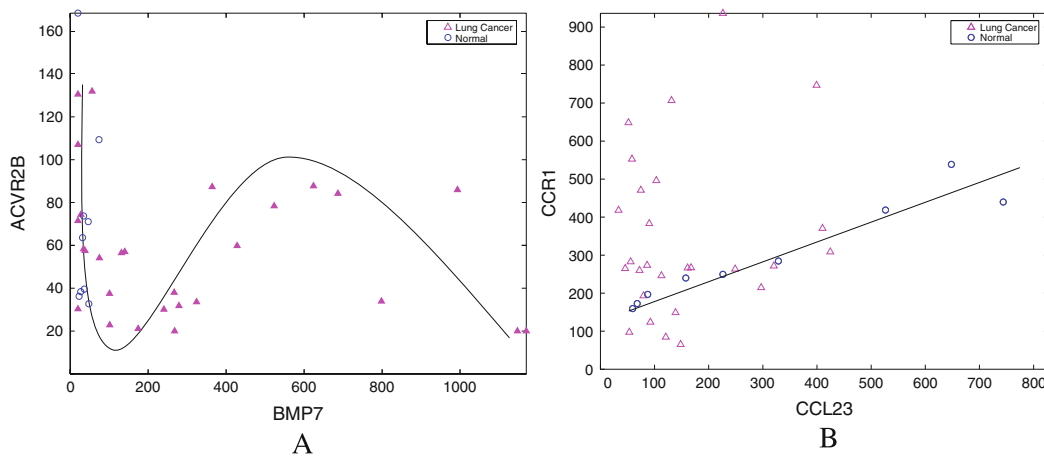


Fig. 12.3. Co-expression profiles of two representative ligand-receptor pairs in lung cancer cells and normal cells. (A) BMP7 and ACVR2B in lung cancer samples ($P_{\text{shuffle}} < 0.028$) and normal samples ($P_{\text{shuffle}} < 0.58$); (B) CCL23 and CCR1 in lung cancer samples ($P_{\text{shuffle}} < 0.73$) and normal samples ($P_{\text{shuffle}} < 2.1\text{E-}09$).

and normal tissues but also among different cancers. It has been reported that chemokine members and their receptors contribute to tumor proliferation, mobility, and invasiveness (29). Some chemokines help to enhance immunity against tumor implantation, while others promote tumor proliferation (30). Our results suggest that a tight interaction between CCL23 and CCR1 is absent in lung and prostate cancer samples but present in AML samples.

Many ligands and receptors showed different patterns of co-expression in cancer and normal tissues. In the lung cancer, for example, 11 ligand–receptor pairs showed a linear co-expression pattern, which were significant in both CoD-B and R^2 , while 28 pairs showed a nonlinear pattern, which were significant only in CoD-B. In the counterpart normal tissue, however, 35 ligand–receptor pairs showed a linear co-expression pattern, while 6 pairs showed a nonlinear pattern. Such differences in the co-expression pattern were not identified in previous co-expression studies based on the correlation coefficient (10). The findings of nonlinear co-expressed pairs of ligand–receptor by CoExMiner provide novel candidates for further study in cancer biology.

3.2. Identification of Disease Genes and Drug Targets of Leukemia

We conducted an analysis of the leukemia-related BCR-ABL pathway using PathwayPro. The analysis profiled the dynamic behavior of the pathway in response to leukemia development and identified possible disease genes and drug targets. Affymetrix array data from chronic myeloid leukemia (CML) and normal white blood cells (31, 32) were downloaded from the GEO database (accession numbers GSE2535 and GSE995). We discretized gene expression values into three categories: over-expressed (1), equivalently-expressed (0), and under-expressed (−1), depending whether the expression level is significantly lower than, similar to, or greater than the respective control threshold. Since some genes have small natural ranges of variation, we used z-transformation to normalize the expression levels of genes across experiments, so that relative expression levels of all genes have the same mean and standard derivation. We then conducted data quantization with the control threshold set to be one standard derivation.

Figure 12.4 shows the network topology of the ABL-BCR pathway (33–35). BCR and ABL are linked to the cytoplasm as part of a large signaling complex with a variety of cellular substrates, related to the development of chronic myeloid leukemia (CML) (33–35). In silico simulation was conducted by mathematical perturbation on the expression value of each gene (referred to as single-gene intervention), each combination of two genes (double-gene intervention), and each combination of three genes (triple-gene intervention). In each perturbation, the observed expression of a gene was altered to the opposite direction or remained unchanged. We measured the transition probability of

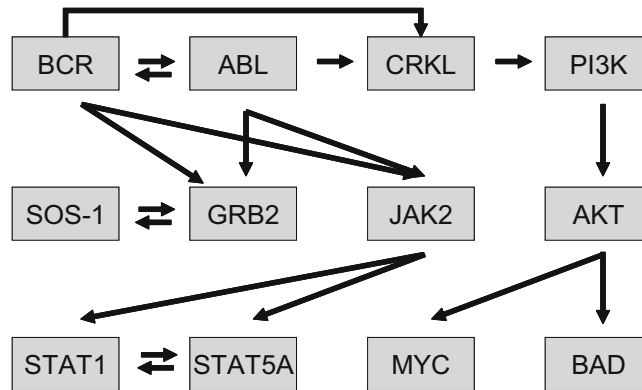


Fig. 12.4. The topology of leukemia-related BCR-ABL pathway. The *arrows* represent the directions of the causal relationships among genes. BCR and ABL are linked to the cytoplasm as a part of a large signaling complex with a variety of cellular substrates, related to the development of leukemia. The drug Gleevec is a selective BCR-ABL inhibitor in this pathway.

the ABL-BCR pathway between the normal condition and the leukemia state under a series of transcriptional interventions. The probability of the network transitioning from normal to leukemia states reveals disease susceptibility of genes involved. The higher the probability is, the more likely a gene or gene combination under a certain intervention is responsible for the development of the disease. On the other hand, the probability of the transition from leukemia to normal states is a measure of the potential usefulness of a drug or therapeutic intervention.

Our analysis first showed that more genes and gene combinations had higher probabilities to contribute to network transitions from normal to leukemia states than from leukemia to normal states (Table 12.2). The result suggests that the chance is higher for a human to develop leukemia than to recover from the disease. As illustrated in Table 12.2, in the double-gene intervention, changes directly involving the genes BCR and ABL yielded the highest probability (0.01) for a normal-to-leukemia transition. The interventions on ABL/AKT1 and BCR/ABL led to the highest transition probabilities (0.002 and 0.001, respectively) for a leukemia-to-normal transition, although they remained nearly 100 times lower than those for normal-to-leukemia transitions. In the triple-gene intervention (Table 12.2), the triplets BCR/ABL/BAD and BCR/ABL/MYC showed a highest probability (0.01) for normal-to-leukemia transition, while the BCR/ABL/AKT combination appeared to have the highest probability (0.007) for leukemia-to-normal transitions. The importance of BCR and ABL to the network transition was further illustrated by the single-gene intervention, where both BCR and ABL were

Table 12.2
Probabilities of network transition by serial interventions on genes in the ABL-BCR pathway of human

Gene	Transcriptional Intervention	Transition pProbability
(A) Transition from normal to CML states by single-gene intervention ^a		
BCR	0 => -1 => 1	0.00639
(B) Transition from CML to normal states by single-gene intervention ^b		
ABL1	1 => 0 => -1	0.000299
(C) Transition from the normal to CML states by double-gene intervention ^c		
BCR ABL1	0 -1 => 1 1 => 1 1	0.0109
BCR BAD	0 1 => -1 0 => 1 0	0.00639
BCR MYC	0 -1 => -1 0 => 1 0	0.00639
BCR BAD	0 1 => -1 -1 => 1 0	0.00639
BCR MYC	0 -1 => -1 1 => 1 0	0.00639
BCR STAT5A	0 1 => -1 -1 => 1 1	0.00639
BCR STAT5A	0 1 => -1 0 => 1 1	0.00639
BCR STAT1	0 0 => -1 1 => 1 0	0.00639
BCR STAT1	0 0 => -1 -1 => 1 0	0.00639
BCR CRKL	0 -1 => -1 1 => 1 0	0.00539
BCR CRKL	0 -1 => -1 0 => 1 0	0.00399
BCR PIK3CG	0 -1 => -1 0 => 1 -1	0.00384
BCR JAK2	0 0 => -1 1 => 1 0	0.00224
BCR AKT1	0 0 => -1 -1 => 1 0	0.00107
(D) Transition from the CML to normal states by double-gene intervention ^d		
ABL1 AKT1	1 0 => 0 1 => -1 0	0.00185
ABL1 AKT1	1 0 => 0 -1 => -1 0	0.00179
BCR ABL1	1 1 => 0 -1 => 0 -1	0.00111
(E) Transition from normal to CML states by triple-gene intervention ^e		
BCR ABL1 BAD	0 -1 1 => 1 1 0 => 1 1 0	0.010936
BCR ABL1 MYC	0 -1 -1 => 1 1 0 => 1 1 0	0.010936
BCR ABL1 BAD	0 -1 1 => 1 1 -1 => 1 1 0	0.010933
BCR ABL1 MYC	0 -1 -1 => 1 1 1 => 1 1 0	0.010933

(continued)

Table 12.2 (continued)

Gene	Transcriptional intervention	Transition pProbability
BCR ABL1 STAT5A	0 -1 1 => 1 1 0 => 1 1 1	0.010933
BCR ABL1 STAT5A	0 -1 1 => 1 1 -1 => 1 1 1	0.010933
BCR ABL1 STAT1	0 -1 0 => 1 1 -1 => 1 1 0	0.010933
BCR ABL1 STAT1	0 -1 0 => 1 1 1 => 1 1 0	0.010933
(F) Transition from CML to normal states by triple-gene intervention ^f		
BCR ABL1 AKT1	1 1 0 => 0 -1 1 => 0 -1 0	0.00684
BCR ABL1 AKT1	1 1 0 => 0 -1 -1 => 0 -1 0	0.00662
ABL1 CRKL AKT1	1 0 0 => 0 -1 1 => -1 -1 0	0.00297
ABL1 CRKL AKT1	1 0 0 => 0 -1 -1 => -1 -1 0	0.00288
BCR ABL1 AKT1	1 1 0 => -1 -1 1 => 0 -1 0	0.00274
BCR ABL1 AKT1	1 1 0 => -1 -1 -1 => 0 -1 0	0.00265
ABL1 CRKL AKT1	1 0 0 => 0 1 1 => -1 -1 0	0.00250
ABL1 CRKL AKT1	1 0 0 => 0 1 -1 => -1 -1 0	0.00242

^aProbability cutoff 1E-4.

^bProbability cutoff 1E-4.

^cProbability cutoff 1E-3.

^dProbability cutoff 1E-3.

^eProbability cutoff 1E-2.

^fProbability cutoff 2E-3.

The gene expression profile of each state is presented as: initial state (e.g., normal state) => state after intervened => end state (e.g., disease state). Transcriptional intervention is presented as: initial state (e.g., normal state) => state after intervened => end state (e.g., disease state). In each state, expression levels of each gene are presented by ternary values.

associated with the highest transition probability (**Table 12.2**). Moreover, BCR and ABL showed high frequencies in all of their partnerships with other genes in the double or triple interventions positive for network transition. As illustrated in **Fig. 12.5**, BCR and ABL were on the top by the frequency of partnership with other genes in the normal-to-leukemia transition, while BCR and ABL, along with AKT and CRKL, were on the top in the leukemia-to-normal transition in the triple-gene invention. These results suggest that BCR and ABL are the most contributive to the network behavior transition between the normal condition and the leukemia state, and therefore the most susceptible for the development of CML leukemia as well as the recovery of the disease to a normal condition. The two genes can thus serve as good drug targets for the treatment of CML

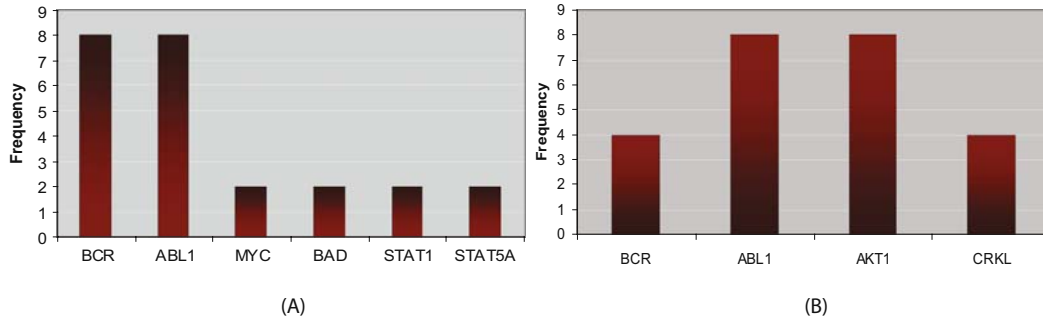


Fig. 12.5. Frequency of partnership of each gene with other genes in the triple-gene interventions on the ABL-BCR associated pathway. The frequency is calculated as the number of occurrence of each gene above a certain transition probability cutoff, after ranking the transition probabilities under the triple-gene interventions. (A) Transition from normal to CML states (transition probability cutoff: 0.01); (B) Transition from CML to normal states (transition probability cutoff: 0.001). BCR and ABL are the most contributive to the network behavior transition from the normal condition to the leukemia state.

leukemia. This result, reached independently by the computational analysis, is in agreement with the conclusion by previous laboratory-based studies. It has been shown that CML is associated in most cases with the fusion of the genes ABL and BCR, and the activation of BCR-ABL represses apoptosis and allows transformed cells to divide, resulting in the development of CML (33–35). The drug Gleevec is a selective BCR-ABL inhibitor, effective in the treatment of CML (36). The PathwayPro analysis not only correctly identified the drug targets but further indicated that BAD and MYC played critical roles in the leukemia development while AKT appeared important in the leukemia recovery to normal. The results provide new insights into our understanding of the leukemia disease.

4. Conclusions

The two algorithms described in this chapter, CoExMiner and PathwayPro, help to decipher biological information from static features of gene co-expression and dynamic behaviors of gene networks. The systems biology analyses allow one to determine how genes interact with each other to perform specific biological processes or functions, and how disease or cellular phenotypes arise from the connectivity or network of genes and their products. The algorithms and software developed for computational systems biology greatly facilitate drug discovery, sensitive diagnostic biomarker identification, and basic investigations in many aspects of biology.

5. Notes



1. We have implemented Java-based interactive computational tools for the CoExMiner and PathwayPro algorithms that we have developed. The software tools are available upon request to the authors.
2. The current version of CoExMiner deals with a pair of genes g_x and g_y , where g_y is the target gene that is predicted by the predictor gene g_x . In the future, we would extend our algorithm to explore multivariate gene relations as well.
3. The current version of PathwayPro allows self-regulation and feedback loops exist in the topologies of pathways. Due to the limitation of computational power, it is feasible for PathwayPro to tackle with the network with 10–20 nodes.

Acknowledgments

This study was supported by the Intramural Research Program, National Institute on Aging, NIH.

References

1. Kitano H. Computational systems biology. *Nature* 2002, 420(6912):206–10.
2. Ideker T, Galitski T, Hood L. A new approach to decoding life: Systems biology. *Annu Rev Genomics Hum Genet* 2001, 2:343–72.
3. Schulze A, Downward J. Navigating gene expression using microarrays – A technology review. *Nat Cell Biol* 2002, 3:E190–E195.
4. Savoie CJ, Aburatani S, Watanabe S, et al. Use of gene networks from full genome microarray libraries to identify functionally relevant drug-affected genes and gene regulation cascades. *DNA Res* 2003, 10(1):19–25.
5. Imoto S, Savoie CJ, Aburatani S, et al. Use of gene networks for identifying and validating drug targets. *J Bioinform Comput Biol* 2003, 1(3):459–74.
6. Stuart JM, Segal E, Koller D, Kim SK. A gene-coexpression network for global discovery of conserved genetic modules. *Science* 2003, 302(5643):249–55.
7. Lee HK, Hsu AK, Sajdak J, Qin J, Pavlidis P. Coexpression analysis of human genes across many microarray data sets. *Genome Res* 2004, 14(6):1085–94.
8. van Noort V, Snel B, Huynen MA. The yeast coexpression network has a small-world, scale-free architecture and can be explained by a simple model. *EMBO Rep* 2004, 5(3): 280–84.
9. Carter SL, Brechbuhler CM, Griffin M, Bond AT. Gene co-expression network topology provides a framework for molecular characterization of cellular state. *Bioinformatics* 2004, 20(14):2242–50.
10. Graeber TG, Eisenberg D. Bioinformatic identification of potential autocrine signaling loops in cancers from gene expression profiles. *Nat Genet* 2001, 29(3):295–300.
11. Butte AJ, Kohane IS. Mutual information relevance networks: functional genomic clustering using pairwise entropy measurements. *Pac Symp Biocomput* 2000:418–29.
12. Herrgard MJ, C overt MW, Palsson BO. Reconciling gene expression data with known genome-scale regulatory network structures. *Genome Res* 2003, 13(11):2423–34.
13. Imoto S, Goto T, Miyano S. Estimation of genetic networks and functional structures

- between genes by using Bayesian networks and nonparametric regression. *Pac Symp Biocomput* 2002:175–86.
14. Zhou X, Wang X, Dougherty ER. Construction of genomic networks using mutual-information clustering and reversible-jump Markov-Chain Monte-Carlo predictor design. *Signal Processing* 2003, 83(4):745–61.
 15. Li H, Sun Y, Zhan M. Analysis of gene coexpression by B-spline based CoD estimation. *EURASIP J Bioinform Syst Biol* 2007, 2007:Article ID 49478, 10 pages.
 16. Dougherty ER, Kim S, Chen Y. Coefficient of determination in nonlinear signal processing. *Signal Processing* 2000, 80:2219–35.
 17. Hashimoto R, Kim S, Shmulevich I, Zhang W, Bittner ML, Dougherty ER. Growing genetic regulatory networks from seed genes. *Bioinformatics* 2004, 20:1241–47.
 18. Huang S. Genomics, complexity and drug discovery: Insights from Boolean network models of cellular regulation. *Pharmacogenomics* 2001, 2(3):203–22.
 19. Kim S, Li H, Dougherty ER, et al. Can Markov chain models mimic biological regulation? *J Biol Syst* 2002, 10(4):337–57.
 20. Shmulevich I, Dougherty ER, Zhang W. Gene perturbation and intervention in probabilistic Boolean networks. *Bioinformatics* 2002, 18(10):1319–31.
 21. de Jong H. Modeling and simulation of genetic regulatory systems: A literature review. *J Comput Biol* 2002, 9(1):67–103.
 22. Smolen P, Baxter DA, Byrne JH. Modeling transcriptional control in gene networks – methods, recent results, and future directions. *Bull Math Biol* 2000, 62(2):247–92.
 23. Li H, Zhan M. Systematic intervention of transcription for identifying network response to disease and cellular phenotypes. *Bioinformatics* 2006, 22(1):96–102.
 24. Prautzsch H, Boehm W, Paluszny M. Bézier and B-spline techniques. Berlin, New York: Springer, 2002.
 25. Cinlar E. Introduction to Stochastic Processes. New Jersey: Prentice Hall, 1975.
 26. Irizarry RA, Bolstad BM, Collin F, Cope LM, Hobbs B, Speed TP. Summaries of Affymetrix GeneChip probe level data. *Nucl Acids Res* 2003, 31(4):e15.
 27. Brubaker KD, Corey E, Brown LG, Vessella RL. Bone morphogenetic protein signaling in prostate cancer cell lines. *J Cell Biochem* 2004, 91(1):151–60.
 28. Yang S, Zhong C, Frenkel B, Reddi AH, Roy-Burman P. Diverse biological effect and Smad signaling of bone morphogenetic protein 7 in prostate tumor cells. *Cancer Res* 2005, 65(13):5769–77.
 29. Muller A, Homey B, Soto H, et al. Involvement of chemokine receptors in breast cancer metastasis. *Nature* 2001, 410(6824):50–56.
 30. Wang JM, Deng X, Gong W, Su S. Chemokines and their role in tumor growth and metastasis. *J Immunol Methods* 1998, 220(1–2):1–17.
 31. Crossman LC, Mori M, Hsieh YC, et al. In chronic myeloid leukemia white cells from cytogenetic responders and non-responders to imatinib have very similar gene expression signatures. *Haematologica* 2005, 90(4):459–64.
 32. Stegmaier K, Ross KN, Colavito SA, O'Malley S, Stockwell BR, Golub TR. Gene expression-based high-throughput screening(GE-HTS) and application to leukemia differentiation. *Nat Genet* 2004, 36(3):257–63.
 33. Zou X, Calame K. Signaling pathways activated by oncogenic forms of Abl tyrosine kinase. *J Biol Chem* 1999, 274(26):18141–44.
 34. Raitano AB, Whang YE, Sawyers CL. Signal transduction by wild-type and leukemogenic Abl proteins. *Biochim Biophys Acta* 1997, 1333:201–16.
 35. Lugo TG, Pendergast AM, Muller AJ, Witte ON. Tyrosine kinase activity and transformation potency of bcr-abl oncogene products. *Science* 1990, 247:1079–82.
 36. Druker BJ, Sawyers CL, Kantarjian H. Activity of a specific inhibitor of the BCR-ABL tyrosine kinase in the blast crisis of chronic myeloid leukemia and acute lymphoblastic leukemia with the Philadelphia chromosome. *N Engl J Med* 2001, 344:1038–42.

Chapter 13

Network Dynamics

Herbert M. Sauro

Abstract

Probably one of the most characteristic features of a living system is its continual propensity to change as it juggles the demands of survival with the need to replicate. Internally these adjustments are manifest as changes in metabolite, protein, and gene activities. Such changes have become increasingly obvious to experimentalists, with the advent of high-throughput technologies. In this chapter we highlight some of the quantitative approaches used to rationalize the study of cellular dynamics. The chapter focuses attention on the analysis of quantitative models based on differential equations using biochemical control theory. Basic pathway motifs are discussed, including straight chain, branched, and cyclic systems. In addition, some of the properties conferred by positive and negative feedback loops are discussed, particularly in relation to bistability and oscillatory dynamics.

Key words: Motifs, control analysis, stability, dynamic models.

1. Introduction

Probably, one of the most characteristic features of a living system is its continual propensity to change even though it is also arguably the one characteristic that, as molecular biologists, we often ignore. Part of the reason for this neglect is the difficulty in making time-dependent quantitative measurements of proteins and other molecules although that is rapidly changing with advances in technology. The dynamics of cellular processes, and in particular cellular networks, is one of the defining attributes of the living state and deserves special attention.

Before proceeding to the main discussion, it is worth briefly listing the kinds of questions that can and have been answered by a quantitative approach (*See Table 13.1*). For example, the notion

Table 13.1
Some problems amenable to a quantitative approach

Problem	Representative solution
Rate-limiting steps	Kacser and Burns (49)
Role of feedback and robustness	Savageau (77)
Analysis of cell-to-cell variation	Mettetal et al. (61)
Rationalization of network structure	Voit et al. (87)
Design of synthetic networks	Kaern and Weiss (51)
New principles of regulation	Altan-Bonnet and Germain (1)
New therapeutic approaches	Bakker et al. (5)
Origin of dominance and recessivity	Kacser and Burns (50)
Missing interactions	Ingolia (46)
Multistationary systems	Many Examples Exist (52)

of the rate-limiting step was originally a purely intuitive invention; once analyzed quantitatively, however, it was shown to be inconsistent with both logic and experimental evidence. There are many examples such as this where a quantitative analysis has overturned a long-held view of how cellular networks operate. In the long term, one of the aims of a quantitative approach is to uncover the general principles of cellular control and organization. In turn this will lead to new approaches to engineering organisms and the development of new therapeutics.

Although traditionally the discipline of molecular biology has had little need for the machinery of mathematics, the non-trivial nature of cellular networks and the need to quantify their dynamics have made mathematics a necessary addition to our arsenal. In this chapter we can sketch only some of the quantitative results and approaches that can be used to describe network dynamics. We will not cover topics such as flux balance, bifurcation analysis, or stochastic models, all important areas of study for systems biology. For the interested reader, much more detail can be had by consulting the reading list at the end of the chapter. Moreover, in this chapter we will not deal with the details of modeling specific systems because this topic is covered in other chapters.

1.1. Quantitative Approaches

The most common formal approach to representing cellular networks has been to use a deterministic and continuous formalism, based invariably on ordinary differential equations (ODE). The reason for this is twofold, firstly ODEs have been shown in many cases to represent adequately the dynamics of real networks, and

secondly, there is a huge range of analytical results on the ODE-based models one can draw upon. Such analytical results are crucial to enabling a deeper understanding of the network under study.

An alternative approach to describing cellular networks is to use a discrete, stochastic approach, based usually on the solution of the master equation via the Gillespie method (27,28). This approach takes into account the fact that at the molecular level, species concentrations are whole numbers and change in discrete, integer amounts. In addition, changes in molecular amounts are assumed to be brought about by the inherent random nature of microscopic molecular collisions. In principle, many researchers view the stochastic approach to be a superior representation because it directly attempts to describe the molecular milieu of the cellular space. However, the approach has two severe limitations, the first is that the method does not scale, that is, when simulating large systems, particularly where the number of molecules is large (>200), it is computationally very expensive. Secondly, there are few analytical results available to analyze stochastic models, which means that analysis is largely confined to numerical studies from which it is difficult to generalize. One of the great and exciting challenges for the future is to develop the stochastic approach to a point where it is as powerful a description as the continuous, deterministic approach. Without doubt, there is a growing body of work, such as studies on measuring gene expression in single cells, which depends very much on a stochastic representation. Unfortunately, the theory required to interpret and analyze stochastic models is still immature though rapidly changing (66, 78). The reader may consider the companion chapter by Resat et al. for the latest developments in stochastic dynamics.

In this chapter we will concentrate on some properties of network structures using a deterministic, continuous approach.

2. Stoichiometric Networks

The analysis of any biochemical network starts by considering the network's topology. This information is embodied in the stoichiometry matrix, N (**Note 1**). In the following description we will follow the standard formalism introduced by Reder (70). The columns of the stoichiometry matrix correspond to the distinct chemical reactions in the network, the rows to the molecular species, one row per species. Thus the intersection of a row and column in the matrix indicates whether a certain species takes part in a particular reaction or not, and, according to the sign of the

$$\begin{array}{c}
 \longleftarrow v_j \longrightarrow \\
 \uparrow \left[\begin{array}{ccc} \alpha_{ij} & \dots & \dots \\ \vdots & & \\ \vdots & & \end{array} \right] \\
 \downarrow
 \end{array}$$

Fig. 13.1. Stoichiometry matrix: \mathbf{N} : $m \times n$, where α_{ij} is the stoichiometric coefficient. S_i denotes the i th species, and v_j the j th reaction.

element, whether it is a reactant or product, and by the magnitude, the relative quantity of substance that takes part in that reaction. Stoichiometry thus concerns the relative mole amounts of chemical species that react in a particular reaction; it does not concern itself with the rate of reaction.

If a given network is composed of m molecular species involved in n reactions, then the stoichiometry matrix is an $m \times n$ matrix. Only those molecular species that evolve through the dynamics of the system are included in this count. Any source and sink species needed to sustain a steady state (non-equilibrium in the thermodynamic sense) are set at a constant level and therefore do not have corresponding entries in the stoichiometry matrix (Fig. 13.1).

2.1. The System Equation

To fully characterize a system one also needs to consider the kinetics of the individual reactions as well as the network's topology. Modeling the reactions by differential equations, we arrive at a system equation that involves both the stoichiometry matrix and the rate vector, thus:

$$\frac{d\mathbf{S}}{dt} = \mathbf{N}\mathbf{v}, \quad [1]$$

where \mathbf{N} is the $m \times n$ stoichiometry matrix and \mathbf{v} is the n dimensional rate vector, whose i th component gives the rate of reaction i as a function of the species concentrations.

2.2. Conservation Laws

In many models of real systems, there will be mass constraints on one or more sets of species. Such species are termed *conserved moieties* (71). A recent review of conservation analysis, which also highlights the history of stoichiometric analysis, can be found in (73). In this section only the main results will be given.

A typical example of a conserved moiety in a computational model is the conservation of adenine nucleotide, i.e., when the total amount of ATP, ADP, and AMP is constant during the evolution of the model. Other examples include NAD/NADH, phosphate, phosphorylated proteins forms, and so on. Figure 13.2 illustrates the simplest possible network that displays a conserved moiety; in this case the total mass, $S_1 + S_2$, is constant during the entire evolution of the network.

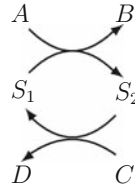


Fig. 13.2. Simple conserved cycle with the constraint, $S_1 + S_2 = T$.

The total amount of a particular moiety in a network is time-invariant and is determined solely by the initial conditions imposed on the system (**Note 2**).

Conserved moieties in the network reveal themselves as linear dependencies in the rows of the stoichiometry matrix (42, 14).

If we examine the system equations for the model depicted in **Fig. 13.2**, it is easy to see that the rate of appearance of S_1 must equal the rate of disappearance of S_2 , in other words $dS_1/dt = -dS_2/dt$. This identity is a direct result of the conservation of mass, namely that the sum $S_1 + S_2$ is constant throughout the evolution of the system.

The stoichiometry matrix for the network depicted in **Fig. 13.2** has two rows $[1, -1]$ and $[-1, 1]$. Since either row can be derived from the other by multiplication by -1 , they are linearly dependent, and the rank of the matrix is 1. Whenever the network exhibits conserved moieties, there will be dependencies among the rows of N , and so the rank of N ($\text{rank}(N)$) will be less than m , the number of rows of N . The rows of N can be rearranged so that the first $\text{rank}(N)$ rows are linearly independent. The species which correspond to these rows can then be defined as the independent species (S_i). The remaining $m - \text{rank}(N)$ are called the dependent species (S_d).

In the simple example shown in **Fig. 13.2**, there is one independent species, S_1 , and one dependent species, S_2 (or, alternatively, S_2 is independent and S_1 dependent).

Once the matrix N has been rearranged as described, we can partition it as

$$N = \begin{bmatrix} N_R \\ N_0 \end{bmatrix},$$

where the submatrix N_R is full rank, and each row of the submatrix N_0 is a linear combination of the rows of N_R . Following Reder (69), we make the following construction. Since the rows of N_0 are linear combinations of the rows of N_R , we can define a link-zero matrix L_0 which satisfies $N_0 = L_0 N_R$. We can combine L_0 with the identity matrix (of dimension $\text{rank}(N)$) to form the link matrix L and hence we can write:

$$N = \begin{bmatrix} N_R \\ N_0 \end{bmatrix} = \begin{bmatrix} I \\ L_0 \end{bmatrix} N_R = L N_R.$$

By partitioning the stoichiometry matrix into dependent and independent sets, we also partition the system equation. The full system equation, which describes the dynamics of the network, is thus:

$$\begin{bmatrix} I \\ L_0 \end{bmatrix} N_R \nu = \frac{dS}{dt} = \begin{bmatrix} dS_i/dt \\ dS_d/dt \end{bmatrix},$$

where the terms dS_i/dt and dS_d/dt refer to the independent and dependent rates of change, respectively. From the above equation, it can be shown that the relationship between the dependent and the independent species is given by: $S_d(t) - S_d(0) = L_0 [S_i(t) - S_i(0)]$ for all time t . Introducing the constant vector $T = S_d(0) - L_0 S_i(0)$, and recalling that $S = (S_i, S_d)$, we can introduce $\Gamma = [-L_0, I]$, and write the vector T concisely as

$$\Gamma S = T.$$

Γ is called the conservation matrix.

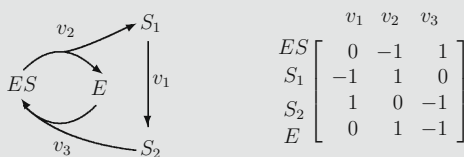
In the example shown in Fig. 13.2, the conservation matrix Γ can be shown to be

$$\Gamma = [1 \ 1].$$

A more complex example is illustrated in Box 1. Algorithms for evaluating the conservation constraints and the Link matrix can be found in (42, 14, 73, 85).

Box 1. Conservation Analysis.

Consider the simple reaction network shown on the left below:



The **stoichiometry matrix** for this network is shown on the right. This network possesses two conserved cycles given by the constraints: $S_1 + S_2 + ES = T_1$ and $E + ES = T_2$. The set of independent species includes $\{ES, S_1\}$, and the set of dependent species $\{E, S_2\}$.

The L_0 matrix can be shown to be:

$$L_0 = \begin{bmatrix} -1 & -1 \\ -1 & 0 \end{bmatrix}.$$

The complete set of equations for this model is therefore:

$$\begin{bmatrix} S_2 \\ E \end{bmatrix} = \begin{bmatrix} -1 & -1 \\ -1 & 0 \end{bmatrix} \begin{bmatrix} ES \\ S_1 \end{bmatrix} + \begin{bmatrix} T_1 \\ T_2 \end{bmatrix}$$

$$\begin{bmatrix} dES/dt \\ dS_1/dt \end{bmatrix} = \begin{bmatrix} 0 & -1 & 1 \\ -1 & 1 & 0 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix}.$$

Note that even though there appears to be four variables in this system, there are in fact only two independent variables, $\{ES, S_1\}$, and hence only two differential equations and two linear constraints.

An excellent source of material related to the analysis of the stoichiometry matrix can be found in the text book by Heinrich and Schuster (37) and more recently (53).

3. Biochemical Control Theory

The system **Eq. [1]** describes the time evolution of the network. This evolution can be characterized in three ways: **thermodynamic equilibrium** where all net flows are zero and no concentrations change in time; **steady state** where net flows of mass traverse the boundaries of the network and no concentrations change in time; and finally the **transient state** where flows and concentrations are both changing in time. Only the steady state and transient states are of real interest in biology. Steady states can be further characterized as stable or unstable, which will be discussed in a later section.

The steady-state solution for a network is obtained by setting the left-hand side of the system **Eq. [1]** to zero, $Nv = 0$, and solving for the concentrations. Consider the simplest possible model:



where we will assume that X_0 and X_1 are boundary species that do not change in time and that each reaction is governed by simple mass-action kinetics. With these assumptions we can write down the system equation for the rate of change of S_1 as:

$$dS_1/dt = k_1 X_0 - k_2 S_1.$$

We can solve for the transient behavior of this system by integrating the system equation and setting an initial condition, $S_1(0) = A_0$, to yield:

$$S_1(t) = A_0 e^{-k_2 t} + \frac{k_1 X_0}{k_2} (1 - e^{-k_2 t}).$$

This equation describes how the concentration of S_I changes in time. The steady state can be determined either by letting t go to infinity or by setting the system equation to zero and solving for S_I ; either way, the steady-state concentration of S_I can be shown to be:

$$S_I = \frac{k_1 X_0}{k_2}.$$

Although simple systems such as this can be solved analytically for both the time course evolution and the steady state, the method rapidly becomes unworkable for larger systems. The problem becomes particularly acute when, instead of simple mass-action kinetics, we begin to use enzyme kinetic rate laws that introduce nonlinearities into the equations. For all intent and purposes, analytical solutions for biologically interesting systems are unattainable. Instead one must turn to numerical solutions; however, numerical solutions are particular solutions, not general, which an analytical approach would yield. As a result, to obtain a thorough understanding of a model, many numerical simulations may need to be carried out. In view of these limitations many researchers apply small perturbation theory (linearization) around some operating point, usually the steady state. By analyzing the behavior of the system using small perturbations, only the linear modes of the model are stimulated and therefore the mathematics becomes tractable. This is a tried and tested approach that has been used extensively in many fields, particularly engineering, to deal with systems where the mathematics makes analysis difficult.

Probably, the first person to consider the linearization of biochemical models was Joseph Higgins at the University of Pennsylvania in the 1950s. Higgins introduced the idea of a “reflection coefficient” (40, 38), which described the relative change of one variable to another for small perturbations. In his Ph.D. thesis, Higgins describes many properties of the reflection coefficients and in later work, three groups, Savageau (75, 77), Heinrich and Rapoport (36, 35), and Kacser and Burns (9, 49) independently and simultaneously developed this work into what is now called Metabolic Control Analysis or Biochemical Systems Theory. These developments extended Higgins’ original ideas significantly and the formalism is now the theoretical foundation for describing deterministic, continuous models of biochemical networks. The theory has, in the past 20 years or so, been further developed with the most recent important advances by Ingalls (45) and Rao (68). In this chapter we will call this approach **Biochemical Control Theory**, or **BCT**.

3.1. Linear Perturbation Analysis

3.1.1. Elementary Processes

The fundamental unit in biological networks is the chemical transformation. Such transformations vary, ranging from simple binding processes, transport processes, to more elaborate aggregated kinetics such as Michaelis-Menten and complex cooperative kinetics.

Traditionally, chemical transformations are described using a rate law. For example, the rate law for a simple irreversible Michaelis-Menten reaction is often given as

$$v = \frac{V_{\max} S}{K_m + S}, \quad [3]$$

where S is the substrate and the V_{\max} and K_m kinetic constants. Such rate laws form the basis of larger pathway models.

A fundamental property of any rate law is the so-called kinetic order, sometimes also called the reaction order. In simple mass-action chemical kinetics, the kinetic order is the power to which a species is raised in the kinetic rate law. Reactions with zero-order, first-order, and second-order are the common types of reactions found in chemistry, and in each case the kinetic order is zero, one, and two, respectively. It is possible to generalize the kinetic order as the scaled derivative of the reaction rate with respect to the species concentration, thus

$$\text{Elasticity Coefficient: } \varepsilon_S^v = \frac{\partial v}{\partial S} \frac{S}{v} = \frac{\partial \ln v}{\partial \ln S} \approx v\%/S\%.$$

When expressed this way, the kinetic order in biochemistry is called the *elasticity coefficient*. Applied to a simple mass-action rate law such as $v = kS$, we can see that $\varepsilon_S^v = 1$. For a generalized mass-action law such as

$$v = k \prod S_i^{n_i},$$

the elasticity for the i th species is simply n_i , that is, it equals the kinetic order. For aggregate rate laws such as the Michaelis-Menten rate law, the elasticity is more complex, for example, the elasticity for the rate law **Eq. [3]** is:

$$\varepsilon_S^v = \frac{K_m}{S + K_m}.$$

This equation illustrates that the kinetic order, though a constant for simple rate laws, is a variable for complex rate laws. In this particular case, the elasticity approaches unity at low substrate concentrations (first-order) and zero at high substrate concentrations (zero-order).

Elasticity coefficients can be defined for any effector molecule that might influence the rate of reaction, this includes substrates, products, inhibitors, activators, and so on. Elasticities are positive for substrates and activators, but negative for products and inhibitors.

At this point, elasticities might seem like curiosities and of no great value; left on their own, this might well be true. The real value of elasticities is that they can be combined into expressions that describe how the whole pathway responds collectively to perturbations. To explain this statement one must consider an additional measure, the control coefficient.

3.1.2. Control Coefficients

Unlike an elasticity coefficient, which describes the response of a single reaction to perturbations in its immediate environment, a control coefficient describes the response of a whole pathway to perturbations in the pathway's environment.

At steady state, a reaction network will sustain a steady rate called the flux, often denoted by the symbol, J . The flux describes the rate of mass transfer through the pathway. In a linear chain of reactions, the steady-state flux has the same value at every reaction. In a branched pathway, the flux divides at the branch points. The flux through a pathway can be influenced by a number of external factors, such as enzyme activities, rate constants, and boundary species. Thus, changing the gene expression that codes for an enzyme in a metabolic pathway will have some influence on the steady-state flux through the pathway. The amount by which the flux changes is expressed by the flux control coefficient.

$$C_{E_i}^J = \frac{dJ}{dE_i} \frac{E_i}{J} = \frac{d \ln J}{d \ln E_i} \approx J\% / E_i\%. \quad [4]$$

In the expression above, J is the flux through the pathway and E_i the enzyme activity of the i th step. The flux control coefficient measures the fractional change in flux brought about by a given fractional change in enzyme activity. Note that the coefficient as well as the elasticity coefficients are defined for small changes.

For a reaction pathway one can plot (**Fig. 13.3**) the steady-state flux, J , as a function of the activity of one of the enzymes. The flux control coefficient can be interpreted on this graph as the scaled slope of the response at a given steady state. Given that the curve is a function of the enzyme activity, it should be clear that the value of the control coefficient is also a function of enzyme activity and consequently the steady state. Control coefficients are not constants but vary according to the current steady state.

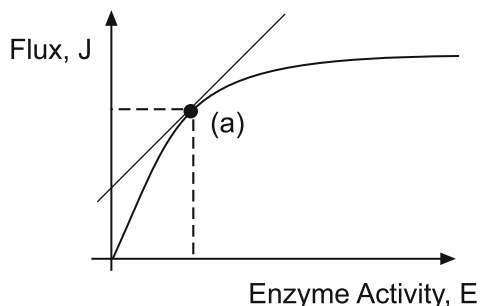


Fig. 13.3. Typical response of the pathway steady-state flux as a function of enzyme activity. The flux control coefficient is defined at a particular operating point, marked (a) on the graph. The value of the coefficient is measured by the scaled slope of the curve at (a).

One can also define a similar coefficient, the concentration control coefficient, with respect to species concentrations, thus:

$$C_{E_i}^S = \frac{dS}{dE_i} \frac{E_i}{S} = \frac{d \ln S}{d \ln E_i} \approx S\%/E_i\%. \quad [5]$$

3.1.3. Relationship Between Elasticities and Control Coefficients

One of the most significant discoveries made early on in the development of BCT (Biochemical Control Theory) was the existence of a relationship between the elasticities and the control coefficients. This enabled one, for the first time, to describe in a general way, how properties of individual enzymes could contribute to pathway behavior. More importantly, this relationship could be studied without the need to solve, analytically, the system Eq. [1]. Particular examples of these relationships will be given in the subsequent sections; here we will concentrate on the general relationship.

There are two related ways to derive the relationship between elasticities and control coefficients, the first is via the differentiation of the system Eq. [1] at steady state and the second by the connectivity theorem.

System Equation Derivation. The system equation can be written more explicitly to show its dependence on the enzyme activities (or any parameter set) of the system: $N\mathbf{v}(s(\mathbf{E}), \mathbf{E}) = \mathbf{0}$. By differentiating this expression with respect to \mathbf{E} , we obtain

$$\frac{d\mathbf{s}}{d\mathbf{E}} = - \left(N_R \frac{\partial \mathbf{v}}{\partial s} L \right)^{-1} N_R \frac{\partial \mathbf{v}}{\partial \mathbf{E}}. \quad [6]$$

The terms $\partial \mathbf{v} / \partial s$ and $\partial \mathbf{v} / \partial \mathbf{E}$ are unscaled elasticities [See (69, 37, 43, 53) for details of the derivation]. By scaling the equation with the species concentration and enzyme activity, the left-hand side becomes the concentration control coefficient expressed in terms of scaled elasticities. The flux control coefficients can also be derived by differentiating the expression: $J = \mathbf{v}[s(\mathbf{p}), \mathbf{p}]$ to yield:

$$\frac{dJ}{d\mathbf{E}} = \left[I - \frac{\partial \mathbf{v}}{\partial s} \left(N_R \frac{\partial \mathbf{v}}{\partial s} L \right)^{-1} N_R \right] \frac{\partial \mathbf{v}}{\partial \mathbf{E}}. \quad [7]$$

Again, the flux expression can be scaled by \mathbf{E} and J to yield the scaled flux control coefficients. These expressions, though unwieldy to some degree, are very useful for deriving symbolic expressions relating the control coefficients to the elasticities. A very thorough treatment together with the derivations of these equations and much more can be found in Hofmeyr 2001.

Theorems. Examination of expressions [6] and [7] yields some additional and unexpected relationships between the control coefficients and the elasticities, called the summation and connectivity theorems. These theorems were originally discovered by modeling small networks using an analog computer (Jim Burns, personal communication), but have since been derived by other means.

The **flux summation theorem** states that the sum of all the flux control coefficients in any pathway is equal to unity.

$$\sum_{i=1}^n C_i^J = 1$$

It is also possible to derive a similar relationship with respect to species concentrations, namely

$$\sum_{i=1}^n C_i^{S_k} = 0$$

In both relationships, n , is the number of reaction steps in the pathway. The flux summation theorem indicates that there is a finite amount of “control” (or sensitivity) in a pathway and implies that control is shared between all steps. In addition, it states that if one step were to gain control, then one or more other steps must lose control.

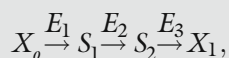
Arguably, the most important relationship is between the control coefficients and the elasticities.

$$\sum C_i^J \varepsilon_S^i = 0$$

This theorem, and its relatives (88, 19, 20), is called the **connectivity theorem** and is probably the most significant relationship in computational systems biology because it relates two different levels of description, the local level, in the form of elasticities, and the system level, in the form of control coefficients. Given the summation and connectivity theorems, it is possible to combine them and solve for the control coefficients in terms of the elasticities. For small networks this approach is a viable way to derive the relationships (19), especially when combined with software such as MetaCon (81), which can compute the relationships algebraically. Box 2 illustrates a simple example of this method.

Box 2. Using Theorems to Derive Control Equations

Consider the simple reaction network, comprising three enzyme-catalyzed reactions, shown below:



where, X_o and X_1 are fixed boundary species. The flux summation theorem can be written down as:

$$C_{E_1}^J + C_{E_2}^J + C_{E_3}^J = 1,$$

while the two connectivity theorems, one centered around each species, are given by:

$$C_{E_1}^J \varepsilon_1^1 + C_{E_2}^J \varepsilon_1^2 = 0$$

$$C_{E_2}^J \varepsilon_2^2 + C_{E_3}^J \varepsilon_2^3 = 0.$$

These three equations can be recast in matrix form as:

$$\begin{bmatrix} 1 & 1 & 1 \\ \varepsilon_1^1 & \varepsilon_1^2 & 0 \\ 0 & \varepsilon_2^2 & \varepsilon_2^3 \end{bmatrix} \begin{bmatrix} C_{E_1}^J \\ C_{E_2}^J \\ C_{E_3}^J \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

The matrix equation can be rearranged to solve for the vector, $[C_{E_1}^J C_{E_2}^J C_{E_3}^J]^T$, by inverting the elasticity matrix, to yield:

$$C_{E_1}^J = \frac{\varepsilon_1^2 \varepsilon_2^3}{\varepsilon_1^1 \varepsilon_2^2 - \varepsilon_1^1 \varepsilon_2^3 + \varepsilon_1^2 \varepsilon_2^3}$$

$$C_{E_2}^J = \frac{-\varepsilon_1^1 \varepsilon_2^3}{\varepsilon_1^1 \varepsilon_2^2 - \varepsilon_1^1 \varepsilon_2^3 + \varepsilon_1^2 \varepsilon_2^3}$$

$$C_{E_3}^J = \frac{-\varepsilon_1^1 \varepsilon_2^2}{\varepsilon_1^1 \varepsilon_2^2 - \varepsilon_1^1 \varepsilon_2^3 + \varepsilon_1^2 \varepsilon_2^3}$$

Further details of the procedure can be found in (19, 20). For larger systems **Eq. [7]** can be used in conjunction with software tools such as Maple, bearing in mind that **Eq. [7]** yields unscaled coefficients.

3.2. Linear Analysis of Pathway Motifs

In the following sections we will describe the application of BCT to some basic and common motifs found in cellular networks. These include, straight chains, branches, cycles, and feedback loops.

3.2.1. Straight Chains

Although linear sequences of reaction steps are actually quite rare in cellular networks (most networks are so heavily branched that uninterrupted sequences are quite uncommon), their study can reveal some basic properties that are instructive to know.

One of the oldest concepts in cellular regulation is the notion of the rate-limiting step. It was Blackman in 1905 (6) who wrote the famous phrase: ‘when a process is conditioned as to its rapidity by a number of separate factors, the rate of the process is limited by the pace of the slowest factor’. It was this statement that started a century long love-affair with the idea of the rate-limiting step in biochemistry, a concept that has lasted to this very day. From the 1930s to the 1950s, there were, however, a number of published

papers which were highly critical of the concept, most notably Burton (11), Morales (62) and Hearon (33) in particular. Unfortunately, much of this work did not find its way into the rapidly expanding fields of biochemistry and molecular biology after the second world war, and instead the intuitive idea first pronounced by Blackman still remains today one of the basic but erroneous concepts in cellular regulation. This is more surprising because a simple quantitative analysis shows that it cannot be true, and there is ample experimental evidence (34, 10) to support the alternative notion, that of shared control.

The confusion over the existence of rate-limiting steps stems from a failure to realize that rates in cellular networks are governed by the law of mass-action, that is, if a concentration changes, then so does its rate of reaction. Many researchers try to draw analogies between cellular pathways and human experiences such as traffic congestion on freeways or customer lines at shopping store check-outs. In each of these analogies, the rate of traffic and the rate of customer checkouts does not depend on how many cars are in the traffic line or how many customers are waiting. Such situations warrant the correct use of the phrase rate-limiting step. Traffic congestion and the customer line are rate-limiting because the only way to increase the flow is to either widen the road or increase the number of cash tills, that is, there is a single factor which determines the rate of flow. In reaction networks, flow is governed by many factors, including the capacity of the reaction (V_{max}) and substrate/ product/effector concentrations. In biological pathways, rate-limiting steps are therefore the exception rather than the rule. Many hundreds of measurements of control coefficients have born out this prediction. A simple quantitative study will also make this clear.

Consider a simple linear sequence of reactions governed by reversible mass-action rate laws:



where X_0 and X_n are fixed boundary species so that the pathway can sustain a steady state. If we assume the reaction rates to have the simple form:

$$v_j = k_j \left(S_j - \frac{S_{j+1}}{q_j} \right),$$

where q_j is the thermodynamic equilibrium constant and k_j the forward rate constant, we can compute the steady state flux, J , to be (37):

$$J = \frac{X_0 \prod_{j=1}^n q_j - X_n}{\sum_{l=1}^n 1/k_l \prod_{j=l}^n q_j}.$$

By modifying the rate laws to include an enzyme factor, such as: $v_j = E_j k_j \left(S_j - \frac{S_{j+1}}{q_j} \right)$, we can also compute the flux control coefficients as (37):

$$C_i^J = \frac{1/k_i \prod_{j=i}^n q_j}{\sum_{l=1}^n 1/k_l \prod_{j=l}^n q_j}.$$

Both equations show that the ability of a particular step to limit the flux is governed not only by the particular step itself but also by *all* other steps. Prior to the 1960s, this was a well-known result (62, 33), but was subsequently forgotten with the rapid expansion of biochemistry and molecular biology. The control coefficient equation also puts limits on the values for the control coefficients in a linear chain, namely $0 \leq C_i^J \leq 1$ and

$$\sum_{i=1}^n C_i^J = 1,$$

which is the flux control coefficient summation theorem. In a linear pathway the control of flux is therefore most likely to be distributed among all steps in the pathway. This simple study shows that the notion of the rate-limiting step is too simplistic and a better way to describe a reaction's ability to limit flux is to state its flux control coefficient.

Although a linear chain puts bounds on the values of the flux control coefficients, branched systems offer no such limits. It is possible that increases in enzyme activity in one limb can decrease the flux through another, hence the flux control coefficient can be negative. In addition, it is possible for the flux control coefficient to be greater than unity (**Note 3**).

3.2.2. Branched Systems

Branching structures in metabolism are probably one of the most common metabolic patterns. Even a pathway such as glycolysis, often depicted as a straight chain in textbooks, is in fact a highly branched pathway.

A linear perturbation analysis of a branched pathway can reveal some interesting potential behavior. Consider the following simple branched pathway (**Fig. 13.4**):

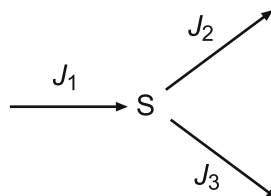


Fig. 13.4. A simple branched pathway. This pathway has three different fluxes, J_1 , J_2 , and J_3 , which at steady state are constrained by $J_1 = J_2 + J_3$.

where J_i are the steady state fluxes. By the law of conservation of mass, at steady state, the fluxes in each limb are governed by the relationship:

$$J_1 - (J_2 + J_3) = 0.$$

In terms of control theory, there will be four sets of control coefficients, one concerned with changes in the intermediate, S, and three sets corresponding to each of the individual fluxes.

Let the fraction of flux through J_2 be given by $\alpha = J_2/J_1$ and the fraction of flux through J_3 be $1 - \alpha = J_3/J_1$. The flux control coefficients for step two and three can be derived and shown to be equal to (19):

$$C_{E_2}^{J_2} = \frac{\varepsilon_1 - \varepsilon_3(1 - \alpha)}{\varepsilon_1 - \varepsilon_2\alpha - \varepsilon_3(1 - \alpha)} > 0 \quad C_{E_3}^{J_2} = \frac{\varepsilon_2(1 - \alpha)}{\varepsilon_1 - \varepsilon_2\alpha - \varepsilon_3(1 - \alpha)} < 0.$$

Note that the flux control coefficient $C_{E_3}^{J_2}$ is negative, indicating that changes in the activity of E_3 decrease the flux in the other limb. To understand the properties of a branched system, it is instructive to look at different flux distributions. For example, consider the case when the bulk of flux moves down J_3 and only a small amount goes through the upper limb J_2 , that is, $\alpha \rightarrow 0$ and $1 - \alpha \rightarrow 1$ (See Fig. 13.5b). Let us examine how the small amount of flux through J_2 is influenced by the two branch limbs, E_2 and E_3 .

$$C_{E_2}^{J_2} \rightarrow \frac{\varepsilon_1 - \varepsilon_3}{\varepsilon_1 - \varepsilon_3} = 1.$$

$$C_{E_3}^{J_2} \rightarrow \frac{\varepsilon_2}{\varepsilon_1 - \varepsilon_3}.$$

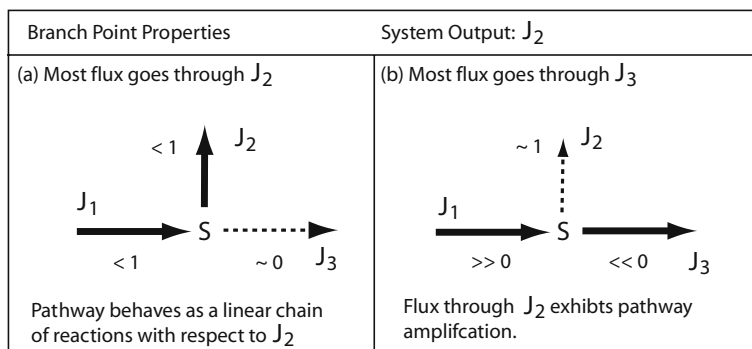


Fig. 13.5. The figure shows two flux extremes relative to the flux through branch J_2 . In case (a) where most of the flux goes through J_2 , the branch reverts functionally to a simple linear sequence of reactions comprising J_1 and J_2 . In case (b), where most of the flux goes through J_3 , the flux through J_2 now becomes very sensitive to changes in activity at J_1 and J_3 . Given the right kinetic settings, the flux control coefficients can become “ultrasensitive” with values greater than one (less than minus one for activity changes at J_3). The values next to each reaction indicates the flux control coefficient for the flux through J_2 with respect to activity at the reaction.

The first thing to note is that E_2 tends to have proportional influence over its own flux. Since J_2 carries only a very small amount of flux, any changes in E_2 will have little effect on S , hence the flux through E_2 is almost entirely governed by the activity of E_2 . Because of the flux summation theorem and the fact that $C_{E_2}^{J_2} = 1$, the remaining two coefficients must be equal and opposite in value. Since $C_{E_3}^{J_2}$ is negative, $C_{E_1}^{J_2}$ must be positive. Unlike a linear chain, the values for $C_{E_2}^{J_2}$ and $C_{E_1}^{J_2}$ are not bounded between zero and one and depending on the values of the elasticities it is possible for the control coefficients to greatly exceed one (48, 55). It is conceivable to arrange the kinetic constants so that every step in the branch has a control coefficient of unity (one of which must be -1). Using the old terminology, we would conclude from this that every step in the pathway is the rate-limiting step.

Let us now consider the other extreme, when most of the flux is through J_2 , that is $\alpha \rightarrow 0$ and $1 - \alpha \rightarrow 0$ (See Fig. 13.5a). Under these conditions the control coefficients yield:

$$C_{E_2}^{J_2} \rightarrow \frac{\varepsilon_1}{\varepsilon_1 - \varepsilon_2}$$

$$C_{E_3}^{J_2} \rightarrow 0$$

In this situation the pathway has effectively become a simple linear chain. The influence of E_3 on J_2 is negligible. Figure 13.5 summarizes the changes in sensitivities at a branch point.

3.2.3. Cyclic Systems

Cyclic systems are extremely common in biochemical networks; they can be found in metabolic, genetic, and particularly signaling pathways. The functional role of cycles is not however fully understood, although in some cases their operational function is beginning to become clear. We can use linear perturbation analysis to uncover some of the main properties of cycles.

Figure 13.6 illustrates two common cyclic structures found in signaling pathways. Such cycles are often formed by a combination of a kinase and a phosphatase. In many cases only one of the molecular species is active. For example, in Fig. 13.6a, let us assume that S_2 is the active (output) species, while in Fig. 13.6b, S_3 is the active (output) species. In a number of cases one observes multiple cycles formed by multi-site phosphorylation. Figure 13.6b shows a common two-stage multi-site cycle. Note that in each case, the cycle steady-state is maintained by the turnover of ATP. One question that can be addressed is how the steady-state output of each cycle, S_2 and S_3 , depends on the input stimulus, S . This stimulus is assumed to be a stimulus of the kinase activity.

One approach to this is to build a detailed kinetic model and solve for the steady-state concentration of S_2 and S_3 as a function of S . This has been done analytically in a few cases (30, 31), but requires the modeler to choose a particular kinetic model for the

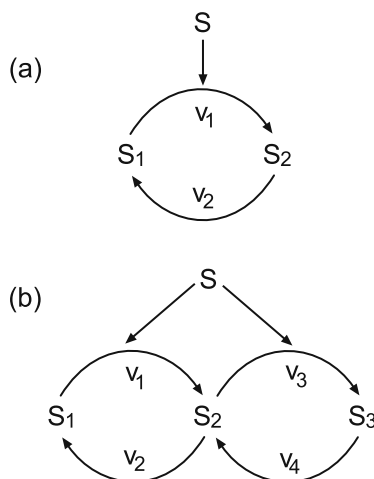


Fig. 13.6. Two common cyclic motifs found in signaling pathways. (a) Single covalent modification cycle, S_2 is the active species, S is the stimulus; (b) Double cycle with S_3 the active species, S is the stimulus.

kinase and phosphatase steps. A perturbation analysis based on BCT need only be concerned with the response characteristics of the kinase and phosphatase steps, not the details of the kinetic mechanism. The response of S_2 to changes in the stimulus S can be shown to be given by the expression (79, 74):

$$C_S^{S_2} = \frac{M_1}{\varepsilon_1^1 M_2 + \varepsilon_2^2 M_1},$$

where $C_S^{S_2}$ is the control coefficient of S_2 with respect to S . M_1 and M_2 are the mole fractions of S_1 and S_2 , respectively, and ε_1^1 the elasticity of v_1 with respect to S_1 and ε_2^2 the elasticity of v_2 with respect to S_2 . If kinase and phosphatase are operating below saturation, then the elasticities will equal one, $\varepsilon_1^1 = 1$ and $\varepsilon_2^2 = 1$; therefore, the response of S_2 to S is simply given by the mole fraction M_1 , which means that the response is bounded between zero and one. This situation is equivalent to the non-ultrasensitive response, sometimes termed the hyperbolic response (31).

In contrast, if the kinase and phosphatase operate closer to saturation, such that the elasticities are much smaller than one, then the denominator in the response equation can be less than the numerator and the control coefficient can exceed one. This situation is representative of zero-order ultrasensitivity and corresponds to the well-known sigmoid response (31). Thus without any reference to detailed kinetic mechanisms, it is possible to uncover the ultrasensitive behavior of the network. We can carry out the same kind of analysis on the dual cycle, **Fig. 13.6b**, to derive the following expression for $C_S^{S_3}$:

$$C_S^{S_3} = \frac{S_1(\varepsilon_2^3 + \varepsilon_2^2) + S_2\varepsilon_1^1}{\varepsilon_2^3\varepsilon_3^4 S_1 + \varepsilon_1^1\varepsilon_3^4 S_2 + \varepsilon_1^1\varepsilon_3^3 S_3}.$$

If we assume linear kinetics on each reaction such that all the elasticities equal one, the equation simplifies to

$$C_S^{S_3} = \frac{2S_1 + S_2}{S_1 + S_2 + S_3}.$$

This shows that given the right ratios for S_1 , S_2 , and S_3 , it is possible for $C_S^{S_3} > 1$. Therefore, unlike the case of a single cycle where near saturation is required to achieve ultrasensitivity, multiple cycles can achieve ultrasensitivity with simple linear kinetics (See Fig. 13.7).

The cyclic models considered here assume negligible sequestration of the cycle species by the catalyzing kinase and phosphatase. In reality, this is not likely to be the case because experimental evidence indicates that the concentrations of the catalyzing enzymes and cycle species are comparable [See (8) for a range of illustrative data]. In such situations additional effects are manifest (21, 72), of particular interest is the emergence of new regulatory feedback loops, which can alter the behavior quite markedly (See 60, 64).

3.2.4. Negative Feedback

A common regulatory motif found in cellular networks is the negative feedback loop (Fig. 13.8). Feedback has the potential to confer many interesting properties on a pathway, with homeostasis probably being the most well known. In this chapter we do not have space to cover all the effects of negative feedback and will focus instead on two properties, homeostasis and instability; however, more details can be found in (74). Using BCT it is easy to show the effect of negative feedback on a pathway.

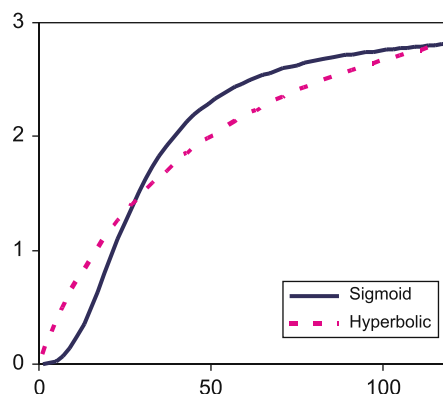


Fig. 13.7. Steady-state responses for the cycles shown in Fig. 13.6. The simplest cycle **6(a)** shows a hyperbolic response when the kinase and phosphatase operate below saturation (*dotted line*). The double cycle **6(b)** shows more complex behavior in the form of a sigmoid response, the kinetics again operating below saturation (*solid line*). This shows that zero-order kinetics is not a necessary condition of ultrasensitivity.

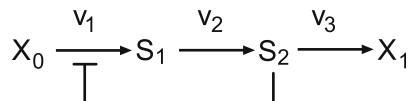


Fig. 13.8. Simple negative feedback loop. v_1 , v_2 , and v_3 are the reaction rates. S_2 acts to inhibit its own production by inhibition of v_1 .

The flux control coefficients for the three steps in **Fig. 13.8** are shown below (77, 48). To aid comparison, the left-hand equations show the equations with feedback while the right-hand equations have been derived assuming no feedback. The feedback term is represented by a single elasticity term, ε_2^1 . This elasticity measures the strength of the feedback and has a negative value, indicating that changes in S_2 result in decreases in the reaction rate of v_1 . For cooperative enzymes, the elasticity may also have values less than -1 .

With Feedback	Without Feedback
$C_{E_1}^J = \frac{\varepsilon_1^2 \varepsilon_2^3}{\varepsilon_1^2 \varepsilon_2^3 - \varepsilon_1^1 \varepsilon_2^3 + \varepsilon_1^1 \varepsilon_2^2 - \varepsilon_1^2 \varepsilon_2^1}$	$C_{E_1}^J = \frac{\varepsilon_1^2 \varepsilon_2^3}{\varepsilon_1^2 \varepsilon_2^3 - \varepsilon_1^1 \varepsilon_2^3 + \varepsilon_1^1 \varepsilon_2^2}$
$C_{E_2}^J = \frac{-\varepsilon_1^1 \varepsilon_2^3}{\varepsilon_1^2 \varepsilon_2^3 - \varepsilon_1^1 \varepsilon_2^3 + \varepsilon_1^1 \varepsilon_2^2 - \varepsilon_1^2 \varepsilon_2^1}$	$C_{E_2}^J = \frac{-\varepsilon_1^1 \varepsilon_2^3}{\varepsilon_1^2 \varepsilon_2^3 - \varepsilon_1^1 \varepsilon_2^3 + \varepsilon_1^1 \varepsilon_2^2}$
$C_{E_3}^J = \frac{\varepsilon_1^1 \varepsilon_2^2 - \varepsilon_1^2 \varepsilon_2^1}{\varepsilon_1^2 \varepsilon_2^3 - \varepsilon_1^1 \varepsilon_2^3 + \varepsilon_1^1 \varepsilon_2^2 - \varepsilon_1^2 \varepsilon_2^1}$	$C_{E_3}^J = \frac{\varepsilon_1^1 \varepsilon_2^2}{\varepsilon_1^2 \varepsilon_2^3 - \varepsilon_1^1 \varepsilon_2^3 + \varepsilon_1^1 \varepsilon_2^2}$

The first difference to notice in the equations is that the denominator, though remaining positive in value, has an additional term compared to the system without feedback, $\varepsilon_1^2 \varepsilon_2^1$. This additional term includes the elasticity of the feedback mechanism.

The numerators for E_1 and E_2 are both unaffected by the feedback. However, because the denominator has an additional positive term, the ratio of numerator to denominator in both cases must be smaller. The flux control coefficients for E_1 and E_2 are therefore reduced in the presence of feedback.

This result might appear at first glance counter-intuitive, surely the “controlled” step must have more “control” (as many undergraduate textbooks will assert)? Closer inspection, however, will reveal a simple explanation. Suppose the concentrations of either E_1 or E_2 are increased. This will cause the concentration of the signal metabolite S_2 to increase. An increase in S_2 will have two effects: the first is to increase the rate of the last reaction step, the second will inhibit the rate through E_1 . The result of this is that the rate increase originally achieved by the increase in E_1 or E_2 will be reduced by the feedback. Therefore, compared to the non-feedback pathway, both enzymes E_1 and E_2 will have less control over the pathway flux. In addition, the greater the feedback elasticity, ε_2^1 the smaller the control coefficients, $C_{E_1}^J$, $C_{E_2}^J$. Thus the stronger the feedback, the less “control” the E_1 and E_2 have over the flux.

What about the flux control coefficient distal to the feedback signal, $C_{E_3}^J$? According to the summation theorem, which states that the sum of the flux control coefficients of a pathway must sum to unity, if some steps experience a reduction in control, then other steps must acquire control. If the flux control of the first two steps decline, then it must be the case that control at the third step must increase. Examination of the third control coefficient equation reveals that as the feedback elasticity (ε_2^1) strengthens, then $C_{E_3}^J$ approaches unity, that is, the last step of the feedback system acquires most of the control.

For drug companies wishing to target pathways, this simple analysis would suggest that the best place to target a drug would be the steps distal to a controlling signal. Traditionally, many have believed it to be the controlled step that should be targeted; however, this analysis indicates that the controlled step is the worst step to target, since it has the least effect on the system. This argument assumes that the targeting does not affect the strength of the feedback itself.

As mentioned previously, one of the most well-known effects of negative feedback is to enhance homeostasis. In this case homeostasis refers to the stabilization of the end product S_2 . We can examine the effect of negative feedback on the homeostasis of S_2 by writing down the concentration control coefficient for $C_{E_3}^{S_2}$

$$C_{E_3}^{S_2} = \frac{\varepsilon_1^1 - \varepsilon_1^2}{\varepsilon_1^2 \varepsilon_2^3 - \varepsilon_1^1 \varepsilon_2^3 + \varepsilon_1^1 \varepsilon_2^2 - \varepsilon_1^2 \varepsilon_2^1}.$$

Note that the numerator is unaffected by the presence of the feedback, whereas the denominator has an additional positive term originating from the feedback mechanism. This means that the feedback decreases the sensitivity of end product, S_2 , with respect to the distal step, E_3 . The effect of the feedback is to stabilize the end product concentration in the face of changing demand from distal steps. This allows a pathway to satisfy the changing demand characteristics of a subsystem distal to the negative feedback loop. We see such an arrangement in many metabolic pathways, clear examples include glycolysis, where demand is measured by ATP consumption, or amino biosynthesis, where demand is protein synthesis. In both cases one could imagine that it is important for the demand system, energy consumption and protein production, to be unimpeded by supply restraints.

Negative feedback therefore has the important task of matching different cellular systems. Hofmeyr and Cornish-Bowden (44) have written extensively on this topic, which they call supply-demand analysis. Interfacing different cellular modules using negative feedback, particularly in signalling pathways, is also discussed in (74).

Only a simple feedback loop has been considered here; for readers who are interested in a more exhaustive analysis, the work by Savageau and co-workers (76, 77, 2) is highly recommended. Moreover, feed-forward negative loops have recently been found to be a common motif and further details can be found in (59).

3.3. Relationship to Engineering Control Theory

In engineering there is much emphasis on questions concerning the stability and performance of technological systems. Over the years, engineers have developed an elaborate and general theory of control, which is applicable to many different technological systems. It is therefore the more surprising that engineering control theory has had little impact on understanding control systems found in biological networks. Part of the problem is related to the rich terminology and abstract nature of some of the mathematics that engineers use, this in turn makes the connection to biological systems difficult to see. This also partly explains why the biological community developed its own theory of control in the form of BCT. Until recently, there was little appreciation of what, if any connection, existed between these two approaches. It turns out the connection is rather more direct any anyone expected. The work by Ingalls (45) in particular [but also (68)], showed that the control coefficients in BCT and the transfer functions used so often in engineering are one and the same thing. This means that much of the machinery of engineering control theory, rather than being perhaps unrelated to biology, can in fact be transferred directly to biological problems.

Following Ingalls (45), let us write down the system equation in the following form:

$$\frac{d\mathbf{s}}{dt} = N\mathbf{v}(s, \mathbf{p}).$$

This equation can be linearized around a suitable operating point such as a steady state to obtain the linearized equation:

$$\frac{d\mathbf{x}}{dt} = \left[N_R \frac{\partial \mathbf{v}}{\partial s} L \right] \mathbf{x}(t) + \left[N_R \frac{\partial \mathbf{v}}{\partial \mathbf{p}} \right] \mathbf{u}(t). \quad [8]$$

This equation describes the rate of change of a perturbation \mathbf{x} around the steady state. For a stable system, the perturbation \mathbf{x} will decay toward the steady state and $\mathbf{x}(t)$ will thus tend to zero. The linearized equation has the standard state space form commonly used in engineering control theory, that is

$$\frac{d\mathbf{x}}{dt} = A\mathbf{x}(t) + B\mathbf{u}(t),$$

with

$$A = N_R \frac{\partial \mathbf{v}}{\partial s} L \quad \text{and} \quad B = N_R \frac{\partial \mathbf{v}}{\partial \mathbf{p}}, \quad [9]$$

$u(t)$ is the input vector to the system, and may represent a set of perturbations in boundary conditions, kinetic constants, or depending on the particular model, gene expression changes.

Because of its equivalence to the state space form, **Eq. [8]** marks the entry point for describing biological control systems using the machinery of engineering control theory. In the following sections two applications, frequency analysis and stability analysis, will be presented, which apply engineering control theory, rephrased using BCT, to biological problems.

3.3.1. Frequency Response

It has been noted previously (3) that chemical networks can act as signal filters, that is, amplify or attenuate specific varying inputs. It may be the case that the ability to filter out specific frequencies has biological significance; for example, a cell may receive many different varying inputs that enter a common signaling pathway; signals that have different frequencies could be identified. In addition, multiple signals could be embedded in a single chemical species (such as Ca^{2+}) and demultiplexed by different target systems. Finally, gene networks tend to be sources of noisy signals that may interfere with normal functioning; one could imagine specific control systems that reduce the noise using high frequency filtering (15).

In steady state, sinusoidal inputs to a linear or linearized system generate sinusoidal responses of the same frequency but of differing amplitude and phase. These differences are a functions of frequency. For a more detailed explanation, Ingalls (45) provides a readable introduction to concept of the frequency response of a system in a biological context.

Whereas the linearized **Eq. [8]** describes the evolution of the system in the time domain, the frequency response must be determined in the frequency domain. Mathematically there is a standard approach, called the Laplace transform, to moving a time domain representation into the frequency representation. By taking the ratio of the Laplace transform of the output to the transform of the input, one can derive the transfer function, which is a complex expression describing the relationship between the input and the output in the frequency domain. The change in the amplitude between the input and output is calculated by taking the absolute magnitude of the transfer function. The phase shift that indicates how much the output signal has been delayed can be computed by computing the phase angle. Note that under a linear treatment, the frequency does not change.

In biological systems the outputs are often the species concentrations or fluxes while the inputs are parameters such as kinetic constants, boundary conditions, or gene expression levels. By

taking the Laplace transform of **Eq. [8]** one can generate its transfer function (45, 68) The transfer function for the species vector s with respect to a set of parameters p is given by:

$$\mathbf{R}_p^s(w) = \left(iw\mathbf{I} - N_R \frac{\partial v}{\partial s} L \right)^{-1} N_R \frac{\partial v}{\partial p}. \quad [10]$$

The response at zero frequency is given by

$$\mathbf{R}_p^s(0) = - \left(N_R \frac{\partial v}{\partial s} L \right)^{-1} N_R \frac{\partial v}{\partial p}.$$

Comparison of the above equation with the concentration control coefficient Equation [6] shows they are equivalent. This is the most important result because it links classical control theory directly with BCT. Moreover, it gives a biological interpretation to the transfer functions so familiar to engineers. The transfer functions can be interpreted as a sensitivity of the amplitude and phase of a signal to perturbations in the input signal. The control coefficients of BCT are the transfer functions computed at zero frequency. Moreover, the denominator term in the transfer functions can be used to ascertain the stability of the system, a topic that will be covered in a later section.

Frequency Analysis of Simple Linear Reaction Chains. The simplest example to consider for a frequency analysis is a two-step pathway, that can be represented as a single gene expressing a protein that undergoes degradation (**Fig. 13.9**). This simple system has been considered previously by Arkin (3) who used a conventional approach to compute the response. Here we will use the BCT approach, which allows us to express the frequency response in terms of elasticities. Using **Eq. [10]** and assuming that the protein concentration has no effect on its synthesis, we can derive the following expression:

$$C_G^P = \frac{1}{\varepsilon_p^D + iw},$$

where ε_p^D is the elasticity for protein degradation with respect to the protein concentration. i is the complex number and w , the frequency input. At zero frequency ($w = 0$) the equation reduces to the traditional control coefficient.

The frequency response of the simple network shown in **Fig. 13.9** is given in **Fig. 13.10**. This response shows a classic low-pass filter response, where at low frequencies the response is high and as the frequency increases the response of the system falls off. The explanation for this is straight forward; at high frequencies, kinetic mechanisms are simply too sluggish to respond fast enough to a rapidly changing signal and the system is unable to pass the input to the output (**Fig. 13.10**).

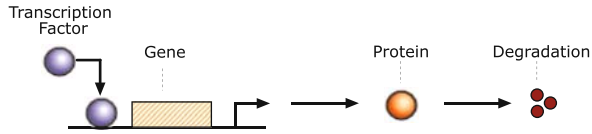


Fig. 13.9. Simple genetic circuit that can act as a low-pass filter.

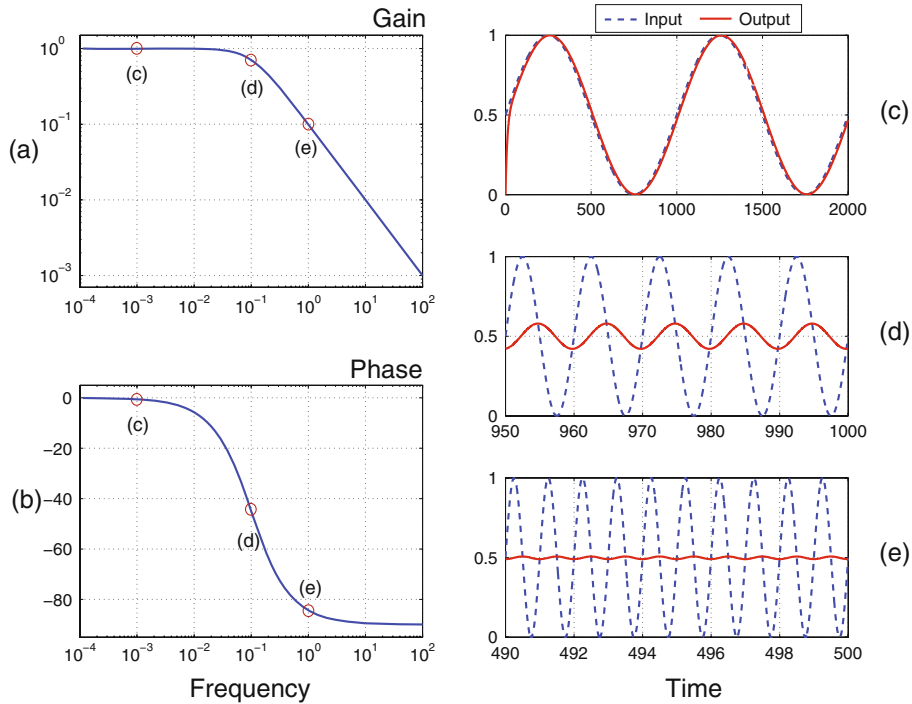


Fig. 13.10. Low-pass frequency response of the simple genetic circuit, **Fig. 13.9**. The two plots on the left indicate the amplitude and phase response, respectively. The three plots on the right show in each case the input signal and corresponding output signal. Each plot on the right was computed at a different frequency; these frequencies are indicated by the marked circles on the plots on the left.

If we cascade a series of genes one after the other (**Fig. 13.11**), the effect is simply to increase the attenuation so that even moderate frequencies are filtered out.

The unscaled response equation for the model shown in **Fig. 13.11** is given by

$$C_{v_1}^{S_3} = \frac{(\tilde{\epsilon})^n \tilde{\epsilon}_p}{(i\omega + \tilde{\epsilon})^n},$$

where the tilde, $\tilde{\sim}$, indicates an unscaled elasticity. n equals the number of genetic stages. We assume that all the elasticities are equal in value.

Many simple systems behave as low-pass filters because physically they are unable to respond fast enough at higher frequencies. Chemical systems are not unusual in this respect. It is possible however

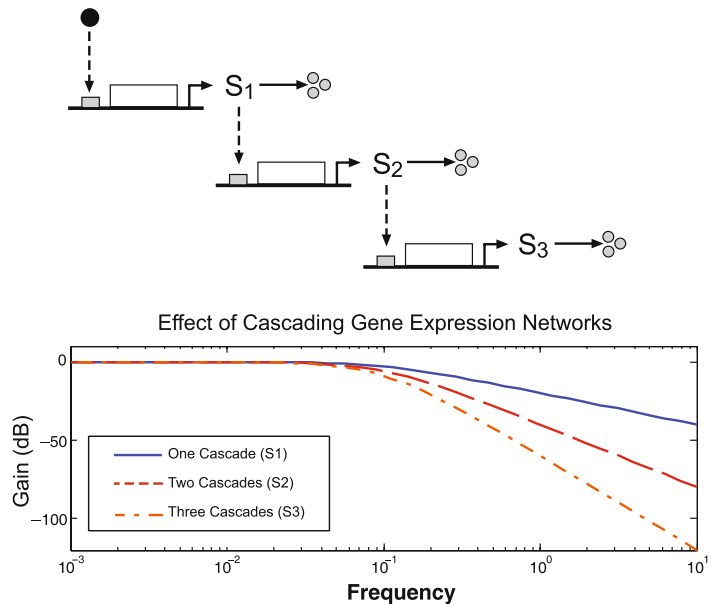


Fig. 13.11. Cascade of simple genetic circuits. See Fig. 13.9 for symbolism. The graph shows the frequency response as the cascade grows in stages. The more stages the greater the attenuation.

through added regulation to change the frequency response. In the paper by Paladugu (65) examples of networks exhibiting a variety of frequency responses are given, including high-pass and band-pass filters. In the next section we will consider how negative feedback can significantly change the frequency response.

Frequency Analysis of a Simple Negative Feedback. In earlier discussions on the effect of negative feedback, the analysis focused on the response to step perturbations on the steady state, effectively the response at zero frequency in the frequency response curve. Here we wish to investigate the frequency response across the entire frequency range.

Figure 13.12 shows the frequency response for the simple network shown in **Fig. 13.8**. The figure includes two graphs, one computed with negative feedback and another without feedback. Without feedback the pathway operates as a simple low-pass filter (Solid line). With feedback (Dotted line), the frequency response is different. As expected, the response at low frequencies is attenuated, which reflects the homeostatic properties of the pathway. What is more interesting is the increase in responsiveness at higher frequencies, that is, the system becomes more sensitive to disturbances over a certain frequency range. This suggests that negative feedback adds a degree of resonance to the system and, given the right conditions, can cause the system to become unstable and

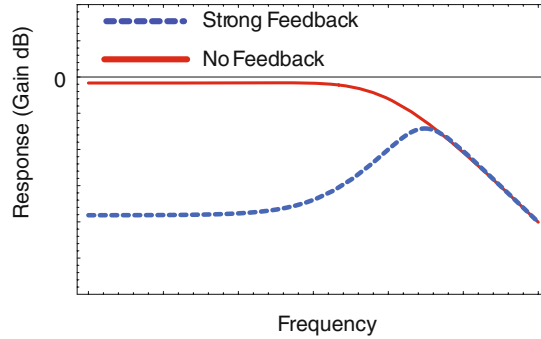


Fig. 13.12. Frequency response of end product S_2 with respect to the input species X_0 for a model of the kind shown in Fig. 13.8.

spontaneously oscillate. The shift in sensitivity to higher frequencies as a result of negative feedback has been observed experimentally in synthetic networks (4).

3.3.2. Stability Analysis

The stability of a system is the response it has to a disturbance to its internal state. Such disturbances can arise as a result of stochastic fluctuations in the concentrations of species or as external disturbances that impose changes on the internal species. If the system recovers to the original state after a disturbance, then it is classed as **stable**; if the system diverges, then it is classed as **unstable**. An excellent review by Jorg Stucki that focuses on stability in biochemical systems can be found in (80).

Consider the simple pathway shown in Eq. [2]. The differential equation for this simple pathway is given by

$$dS_1/dt = k_1 X_0 - k_2 S_1. \quad [11]$$

It can easily be shown that disturbances to S_1 are stable. At steady state, $dS_1/dt = 0$; thus by making a small disturbance, δS_1 in S_1 we can compute the effect this has on the rate of change of δS_1 to be:

$$d(\delta S_1)/dt = -k_2 \delta S_1. \quad [12]$$

This shows that after the initial disturbance, the disturbance itself declines exponentially to zero; in other words, the system returns to the original steady state and the system is therefore stable. By dividing both sides by δS_1 and taking the limit to infinitesimal changes, one can show (53) that the term, $-k_2$, is equal to, $\partial d(S_1/dt)/\partial S_1$. The stability of this simple system can therefore be determined by inspecting the sign of $\partial d(S_1/dt)/\partial S_1$.

Now consider a change to the kinetic law, $k_1 X_o$, governing the first reaction. Instead of simple linear kinetics let us use a cooperative enzyme which is activated by the product S_I . The rate law for the first reaction is now given by:

$$v_1 = \frac{k_1 X_o (X_o + 1) (S_I + 1)^2}{(S_I + 1)^2 (X_o + 1)^2 + 80000}.$$

Setting $X_o = 1$, $k_1 = 100$, $k_2 = 0.14$, a steady-state concentration of S_I can be determined to be 66.9. Evaluating the derivative $\partial d(S_I/dt)/\partial S_I$ at this steady state yields a value of 0.084, which is clearly a positive value. This means that any disturbance to S_I at this particular steady state will cause S_I to increase; in other words, this steady state is unstable.

For single variable systems the question of stability reduces to determining the sign of the $\partial d(S_I/dt)/\partial S_I$ derivative. For larger systems the stability of a system can be determined by looking at all the terms $\partial d(S_i/dt)/\partial S_i$ which are given collectively by the expression:

$$\frac{d(ds/dt)}{ds} = J, \quad [13]$$

where J is called the Jacobian matrix containing elements of the form $\partial d(S_i/dt)/\partial S_i$. Equation [12] can be generalized to:

$$\frac{d(\delta s)}{dt} = J \delta s.$$

Analysis shows that solutions to the disturbance equations [12] and [13] are sums of exponentials where the exponents of the exponentials are given by the eigenvalues of the Jacobian matrix, J (53). If the eigenvalues are negative then the exponents decay (stable), whereas if they are positive then the exponents grow (unstable).

Another way to obtain the eigenvalues is to look at the roots (often called the poles in engineering) of the characteristic equation, which can be found in the denominator of the transfer function, Eq. [10]. For stability, the real parts of all the poles of the transfer function should be negative. If any pole is positive, then the system is unstable. The characteristic equation can be written as a polynomial, where the order of the polynomial reflects the size of the model.

$$a_n s^n + a_{n-1} s^{n-1} + \dots + a_1 s + a_0 = 0$$

A test for stability is that all the coefficients of the polynomial must have the same sign if all the poles are to have negative real parts. Also it is necessary for all the coefficients

Table 13.2
Routh-Hurwitz table

a_n	a_{n-2}	a_{n-4}	\cdots
a_{n-1}	a_{n-3}	a_{n-5}	\cdots
b_1	b_2	b_3	\cdots
c_1	c_2	c_3	\cdots
etc.			

to be nonzero for stability. A technique called the Routh-Hurwitz criterion can be used to determine the stability. This procedure involves the construction of a “Routh Array” shown in **Table 13.2**. The third and fourth rows of the table are computed using the relations:

$$b_1 = \frac{a_{n-1}a_{n-2} - a_n a_{n-3}}{a_{n-1}} \quad b_2 = \frac{a_{n-1}a_{n-4} - a_n a_{n-5}}{a_{n-1}} \text{ etc.}$$

$$c_1 = \frac{b_1 a_{n-3} - b_2 a_{n-1}}{b_1} \quad c_2 = \frac{b_1 a_{n-5} - b_3 a_{n-1}}{b_1} \text{ etc.}$$

Rows to the table are added until a row of zeros is reached. Stability is then determined by the number of sign changes in the **1st** column, which is equal to the number of poles with real parts greater than zero. **Table 13.3** shows the Routh table for the characteristic equation $s^3 + s^2 - 3s - 1 = 0$ where $s = i\omega$. From the **Table 13.3** we see one sign change between the second and the third rows. This tells us that there must be one positive root. Since there is one positive root, the system from which this characteristic equation was derived is unstable.

The advantage of using the Routh-Hurwitz table is that entries in the table will be composed from elasticity coefficients. Thus sign changes (and hence stability) can be traced to particular constraints on the elasticity coefficients. Examples of this will be given in the next section.

Table 13.3
Routh-Hurwitz table

1	-3
1	-1
-2	
-1	

3.4. Dynamic Motifs

3.4.1. Bistable Systems

The question of stability leads on to the study of systems with non-trivial behaviors. In the previous section a model was considered, which was shown to be unstable. This model was described by the following set of rate equations:

$$v_1 = \frac{k_1 X_o (X_o + 1) (S_1 + 1)^2}{(S_1 + 1)^2 (X_o + 1)^2 + 80000},$$

$$v_2 = k_2 S_1.$$

The network is depicted in **Fig. 13.13** and illustrates a positive feedback loop, that is, S_1 stimulates its own production.

The steady state of this simple model is computed at $dS_1/dt = v_1 - v_2 = 0$ or $v_1 = v_2$. If v_1 and v_2 are plotted against S_1 (**Fig. 13.14**), the points where the curves intersect correspond to the steady states of the system. Inspection of **Fig. 13.14** shows three intersection points.

The steady-state solution that was examined earlier ($S_1 = 66.9$) corresponds to the second intersection point and, as shown, this steady state is unstable. Solutions to the system can be found at

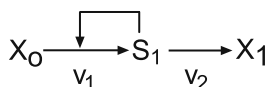


Fig. 13.13. Simple pathway with positive feedback.

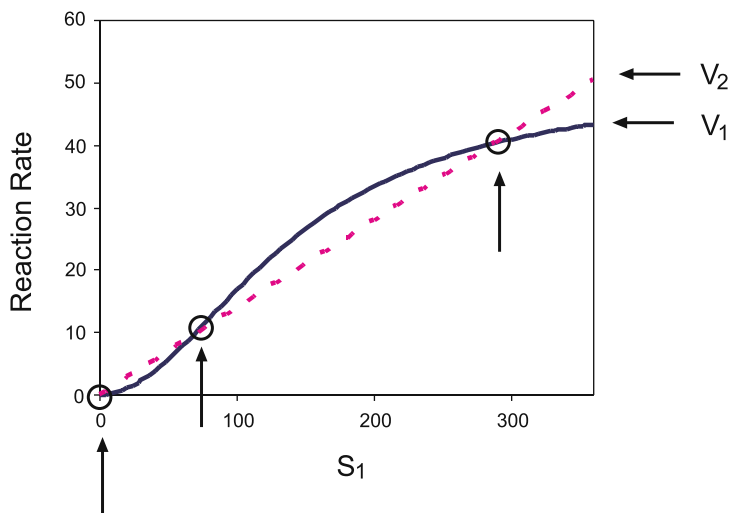


Fig. 13.14. Graph showing v_1 and v_2 plotted against the species concentration, S_1 for the model depicted in **Fig. 13.13**. The intersection points, where $v_1 = v_2$ are marked with small circles and indicate three possible steady states. Rate equations: $v_1 = (k_1 X_o (X_o + 1) (S_1 + 1)^2) / ((S_1 + 1)^2 (X_o + 1)^2 + 80000)$, $v_2 = k_2 S_1$, and parameter values $X_o = 1$, $k_1 = 100$, $k_2 = 0.14$. The steady-state solutions correspond to values of S_1 at 0.019, 66.89, and 288.23.

Table 13.4
Table of steady-state S_I and corresponding value for the Jacobian element. Negative Jacobian values indicate a stable steady state, positive elements indicate an unstable steady state. The table shows one stable and two unstable steady states

Steady State S_I	Jacobian Element: $(dS_I/dt)/dS_I$
0.019	-0.086
66.89	0.084
288.23	-0.135

values of S_I at 0.019, 66.89, and 288.23. By substituting these values into the equation for dS_I/dt we can compute the Jacobian element in each case (Table 13.4).

This system possess three steady states, one unstable and two stable. Such a system is known as a bistable system because it can rest in one of two stable states. One question which arises is what are the conditions for bistability? This can be easily answered using BCT. The unscaled frequency response of S_I with respect to v_I can be computed using Eq. [10] to yield:

$$C_{v_I}^{S_I} = \frac{1}{\varepsilon_1^2 - \varepsilon_1^1 + iw}$$

Constructing the Routh-Hurwitz table indicates one sign change, which is determined by the term, $\varepsilon_1^2 - \varepsilon_1^1$. Note that ε_1^1 has a positive value because S_I activates v_I . Because the pathway is a linear chain the elasticities can be scaled without changes to the stability terms criterion, thus the pathway is stable if

$$\varepsilon_1^1 > \varepsilon_1^2.$$

If we assume first-order kinetics in the decay step, v_2 , then the scaled elasticity, ε_1^2 will be equal to unity, hence

$$\varepsilon_1^1 > 1.$$

This result shows that it is only possible to achieve bistability if the elasticity of the positive feedback is greater than one (assuming the consumption step is first order). The only way to achieve this is through some kind of cooperative or multimeric binding, such as dimerization or tetramer formation. The bistability observed in the lac operon is a possible example of this effect (57, 56).

3.4.2. Feedback and Oscillatory Systems

The study of oscillatory systems in biochemistry has a long history dating back to at least the 1950s. Until recently, however, there was very little interest in the topic from mainstream molecular biology. In fact, one suspects that the concept of oscillatory behavior in cellular networks was considered more a curiosity, and a rare one at that, than anything serious. With the advent of new measurement technologies, particularly high-quality microscopy, and the ability to monitor specific protein levels using GFP and other fluorescence techniques, a whole new world has opened up to many experimentalists. Of particular note is the recent discovery of oscillatory dynamics in the p53/Mdm2 couple (54, 26) and Nf- κ B (41) signaling; thus rather than being a mere curiosity, oscillatory behavior is in fact an important, though largely unexplained, phenomenon in cells.

Basic Oscillatory Designs. There are two basic kinds of oscillatory designs, one based on negative feedback and a second based on a combination of negative and positive feedback. Both kinds of oscillatory design have been found in biological systems. An excellent review of these oscillators and specific biological examples can be found in (84, 18). A more technical discussion can be found in (83, 82).

Negative Feedback Oscillator. Negative feedback oscillators are the simplest kind to understand and probably one of the first to be studied theoretically (32). Savageau (77) in his book provides a detailed analysis and summary of the properties of feedback oscillators. Figure 8 shows a simple example of a system with a negative feedback loop. We can use BCT to analyze this system by deriving the characteristic equations (the denominator of the frequency response) and constructing a Routh-Hurwitz table. Using this technique it can be easily shown that a pathway with only two intermediates in the feedback loop *cannot* oscillate. In general, a two-variable system with a negative feedback is stable under all parameter regimes. Once a third variable has been added, the situation changes and the pathway shown in **Fig. 13.15**, which has three variables, can admit oscillatory behavior.

A critical factor that determines the onset of oscillations, apart from the number of variables, is the strength of the feedback. Savageau (77) showed that if the substrate elasticities were equal (e.g., all first-order kinetics), then the ratio of the feedback elasticity (ϵ_{inh}) to the output elasticity ($\epsilon_{sub}, \epsilon_3^4$) determined the onset of oscillations (**Table 13.5**). **Table 13.5** shows that as the

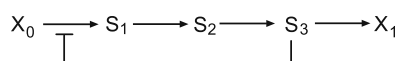


Fig. 13.15. Simple negative feedback model with three variables, S_1 , S_2 , and S_3 . This network can oscillate.

Table 13.5
Relationship between the pathway length and the degree of feedback inhibition on the threshold for stability. ϵ_{inh} is the elasticity of the feedback inhibition and ϵ_{sub} is the elasticity of the distal step with respect to the signal

Length of pathway	Instability threshold $-\epsilon_{inh}/\epsilon_{sub}$
1	Stable
2	Stable
3	8.0
4	4.0
5	2.9
6	2.4
7	2.1
\vdots	\vdots
∞	1.0

pathway becomes longer less feedback inhibition is required to destabilize the pathway. This highlights the other factor that contributes to instability, the delay in routing the signal around the network. All feedback oscillators require some device to provide amplification of a signal combined with a suitable time delay so that the signal response can go out of phase. In metabolic pathways, amplification is often provided by a cooperative enzyme while the delay is provided by the intermediate steps in the pathway. In signaling pathways, amplification can be generated by covalent modification cycles. Amplification can also be provided by another means. The criterion for instability is the ratio of the inhibition elasticity to the substrate elasticity. If the output reaction of the pathway is governed by a saturable enzyme, then it is possible to have ϵ_{sub} less than unity. This means that it is possible to trade cooperativity at the inhibition site with saturation at the output reaction. The modified Goodwin model of Bliss (7) illustrates the model with no cooperativity at the inhibition site, but with some saturation at the output reaction by using a simple Michaelis-Menten rate law.

A second property uncovered by BCT is that stability is enhanced if the kinetic parameters of the participating reactions are widely separated, that is, a mixture of “fast” and “slow” reactions. The presence of “fast” reactions effectively shortens the pathway, and thus it requires higher feedback strength to destabilize the pathway since the delay is now less.

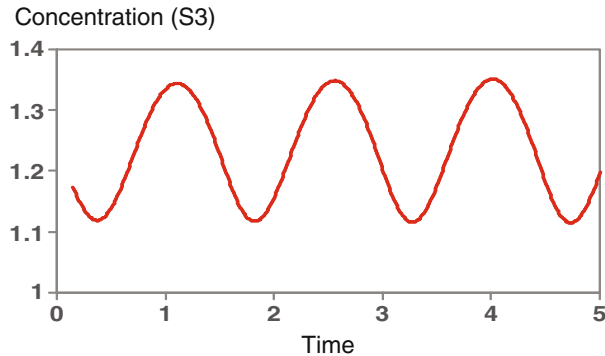


Fig. 13.16. Plot of S_3 versus time for the model shown in Fig. 13.15. Note that the profile of the oscillation is relatively smooth.

One of the characteristics of negative feedback oscillators is that they tend to generate smooth oscillations (Fig. 13.16), and in man-made devices they are often used to generate simple trigonometric functions.

A related oscillator that operates in a similar way to the feedback oscillator is the ring oscillator (See Fig. 13.17). This device is composed of an odd number of signal inverters connected into a closed chain. Instability requires sufficient amplification between each inverter so that the signal strength is maintained. A ring oscillator has been implemented experimentally in *Escherichia coli* (17) where it was termed a repressilator. Ring oscillators with an even number of inverters can be used to form memory units or toggle switches. The even number of units means that the signal latches to either on or off, the final state depending on the initial conditions. Toggle circuits have also been implemented experimentally in *E. coli* (25).

Relaxation Oscillators. A favorite oscillator design amongst theorists (58, 63, 22, 23), as well as biological evolution (86, 24, 29, 67, 12), is the relaxation oscillator. This kind of oscillator operates by charging a species concentration that, upon reaching a threshold, changes the state of a bistable switch. When the switch

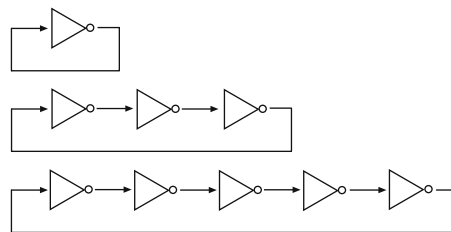


Fig. 13.17. Three ring oscillators, one-stage, three-stage, and five-stage oscillators. All ring oscillators require an odd number of gating elements. Even rings behave as toggle switches.

changes state, it causes the species to discharge. Once the species has discharged, the bistable switch returns to the original state and the sequence begins again. Positive feedback or a two-step ring oscillator forming a toggle switch is used to generate the bistability, and a negative feedback loop provides the signal to switch the bistable switch.

One of the characteristics of a relaxation oscillator is the “spiky” appearance of the oscillations. This is due to the rapid switching of the bistable circuit, which is much faster compared to the operation of the negative feedback. Man-made devices that utilize relaxation oscillators are commonly used to generate saw-tooth signals. **Figure 13.18** illustrates a plot from a hypothetical relaxation oscillator published by Tyson’s group (84).

Oscillator Classification. As previously discussed, oscillators fall into two broad categories, feedback oscillators and relaxation oscillators. Within the relaxation oscillation group, some authors (84) have proposed to divide this group into two and possibly three additional subgroups; these include substrate-depletion, activator-inhibitor, and toggle-based relaxation oscillators. The grouping is based on two-variable oscillators and a comparison of the sign patterns in the Jacobian matrix. Although toggle-based relaxation oscillations have the same Jacobian sign pattern as substrate-depletion based oscillations, the bistability is implemented differently.

Figure 13.19 shows examples of six different oscillators, together with their classification and stylized forms.

Even though each mechanistic form (first column) in **Fig. 13.19** looks different, the stylized forms (second column) fall into one of three types. The stylized forms reflect the structure

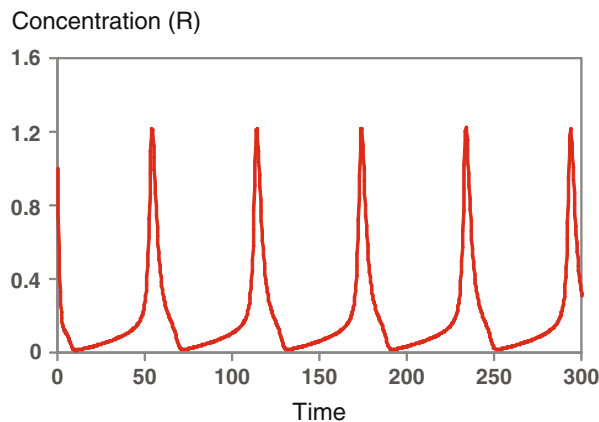


Fig. 13.18. Typical spiky appearance of oscillatory behavior from a relaxation oscillator, from Tyson (84), model 2(c).

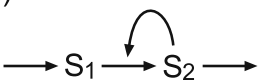
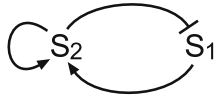
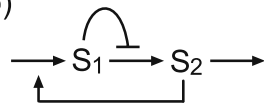
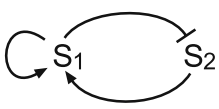
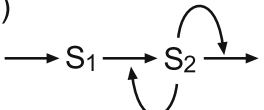
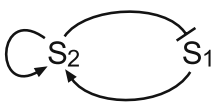
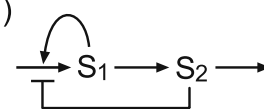
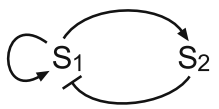
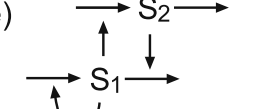
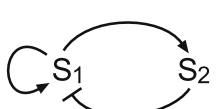
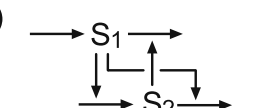
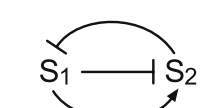
Mechanism	Stylized Form	Type
a) 		SD
b) 		SD
c) 		SD
d) 		AI
e) 		AI
f) 		SD/T

Fig. 13.19. Classification of Relaxation Oscillators into substrate-depletion, activator-inhibitor, and toggle-based. Note that although the mechanistic networks are quite variable, the underlying operation is the same as shown in the stylized column. Type codes: SD = Substrate-Depletion; AI = Activator- Inhibitor; SD/T = Substrate-Depletion/ Toggle. The stylized form is generated by computing the Jacobian matrix for each network. Elements in the Jacobian indicate how each species influences changes in another. Model (a) corresponds to model (c) in **Fig. 13.2** of (84) and model (e) to model (b) in **Fig. 13.2** of (84).

of the Jacobian for each model. Only a limited number of sign patterns in the Jacobian can yield oscillators (39). Using evolutionary algorithms (16, 65), many hundreds of related mechanisms can be generated, see the model repository at www.sys-bio.org for a large range of examples. Although many of these evolved oscillators look quite different, each one can be classified in a only a few basic configurations.

4. Summary

This chapter has focused on describing some of the theory that is available to analyze the dynamics of deterministic/continuous models of biochemical networks. Some areas have been omitted, in particular bifurcation analysis has not been discussed but is probably one of the more important tools at our disposal because it can be used to uncover the different qualitative behavioral regimes a network might possess. Bifurcation analysis would require an entire chapter to describe; however, good starting points include the chapter by Conrad and Tyson (13) and the book by Izhikevich (47).

The most significant area missing from this chapter is undoubtedly a discussion on stochastic modeling (89). As more experimental data becomes available on times series changes in species concentrations, it is becoming abundantly clear that many processes, particularly genetic networks, are noisy. In prokaryotic systems we are often dealing with small numbers of molecules and the stochastic nature of reaction dynamics becomes an important consideration. Unfortunately, there is at present little accessible theory on the analysis of stochastic models, which greatly impedes their utility. In almost all cases, the analysis of stochastic systems relies exclusively on numeric simulation, which means generalizations are difficult to make. Some researches have started to consider the theoretical analysis of stochastic systems (66, 78) and the field is probably one of the more exiting areas to consider in the near future.

5. Notes



1. A recent and potentially confusing trend has been to use the symbol S to signify the stoichiometry matrix. The use of the symbol N has, however, a long tradition in the field, the letter N being used to represent “number,” indicating stoichiometry. The symbol, S , is usually reserved for species.
2. There are rare cases when a “conservation” relationship arises out of a non-moiety cycle. This does not affect the mathematics, but only the physical interpretation of the relationship. For example, $A \rightarrow B + C$; $B + C \rightarrow D$ has the conservation, $B - C = T$.
3. Possibly inviting the use of the term, ultra-rate-limiting?

6. Reading List

The following lists books and articles that cover the material in this chapter in much more depth.

Introductory and Advanced Texts on Systems Analysis

Fell, D (1996) *Understanding the Control of Metabolism*, Ashgate Publishing, ISBN: 185578047X

Heinrich R, Schuster S (1996) *The Regulation of Cellular Systems*. Chapman and Hall, ISBN: 0412032619

Klipp E, et al. (2005) *Systems Biology in Practice, Concepts, Implementation and Application*. Wiley-VCH Verlag, ISBN: 3527310789

Izhikevich, E. M. (2007) *Dynamical Systems in Neuroscience: The Geometry of Excitability and Bursting*, MIT Press, ISBN: 0262090430

Control Theory

Ingalls, B. P. (2004) A frequency domain approach to sensitivity analysis of biochemical systems, *Journal of Physical Chemistry B*, 108, 1143–1152

Bistability and Oscillations

Tyson J, et al. (2003) Sniffers, buzzers, toggles and blinkers: dynamics of regulatory and signaling pathways in the cell. *Current Opinion in Cell Biology*, 15, 221–231

Stochastic Modeling

Wilkinson D. J. (2006) *Stochastic Modeling for Systems Biology*. Chapman and Hall, ISBN: 1584885408

Acknowledgments

I wish to acknowledge Ravishankar R. Vallabhajosyula for assistance in preparing the simulation data and figures for the gene cascade circuits. This work was supported by a generous grant from the NSF (award number CCF- 0432190).

References

1. Altan-Bonnet G, Germain RN (2005). Modeling T cell antigen discrimination based on feedback control of digital ERK responses. *PLoS Biol* 3(11):1925–1938.
2. Alves R, Savageau MA (2000). Effect of overall feedback inhibition in unbranched biosynthetic pathways. *Biophys J* 79: 2290–2304.
3. Arkin AP (2000). Signal Processing by Biochemical Reaction Networks. In J. Walleczek (Ed.), *Self-Organized Biological Dynamics and Nonlinear Control*, pp. 112–114. Cambridge University Press.
4. Austin DW, Allen MS, McCollum JM, Dar RD, Wilgus JR, Saylor GS, Samatova NF, et al. (2006). Gene network shaping of

- inherent noise spectra. *Nature* 439(7076): 608–611.
5. Bakker BM, Westerhoff HV, Opperdoes FR, Michels PAM (2000). Metabolic control analysis of glycolysis in trypanosomes as an approach to improve selectivity and effectiveness of drugs. *Mol Biochem Parasitology* 106:1–10.
 6. Blackman FF (1905). Optima and limiting factors. *Ann Botany* 19:281–295.
 7. Bliss RD, Painter PR, Marr AG (1982). Role of feedback inhibition in stabilizing the classical operon. *J Theor Biol* 97(2):177–193.
 8. Blüthgen N, Bruggeman FJ, Legewie S, Herzog H, Westerhoff HV, Kholodenko BN (2006). Effects of sequestration on signal transduction cascades. *FEBS J* 273(5): 895–906.
 9. Burns JA (1971). Studies on Complex Enzyme Systems. Ph. D. thesis, University of Edinburgh. <http://www.cds.caltech.edu/hsauro/Burns/jimBurns.pdf>
 10. Burrell MM, Mooney PJ, Blundy M, Carter D, Wilson F, Green J, Blundy KS, et al. (1994). Genetic manipulation of 6-phosphofructokinase in potato tubers. *Planta* 194:95–101.
 11. Burton AC (1936). The basis of the principle of the master reaction in biology. *J Cell Comp Physiol* 9(1):1–14.
 12. Chen KC, Calzone L, Csikasz-Nagy A, Cross FR, Novak B, Tyson JJ (2004). Integrative analysis of cell cycle control in budding yeast. *Mol Biol Cell* 15(8): 3841–3862.
 13. Conrad ED, Tyson JJ (2006). Modeling Molecular Interaction Networks with Non-linear Ordinary Differential Equations. In J. S. Zoltan Szallasi and V. Periwal (Eds.), *System Modeling in Cellular Biology From Concepts to Nuts and Bolts*, Chapter 6, pp. 97–123. MIT Press.
 14. Cornish-Bowden A, Hofmeyr JHS (2002). The Role of Stoichiometric Analysis in Studies of Metabolism: An Example *J Theor Biol* 216:179–191.
 15. Cox C, McCollum J, Austin D, Allen M, Dar R, Simpson M (2006). Frequency domain analysis of noise in simple gene circuits. *Chaos* 16(2):26102–26102.
 16. Deckard A, Sauro HM (2004). Preliminary studies on the in silico evolution of biochemical networks. *Chembiochem* 5:1423–31.
 17. Elowitz MB, Leibler S (2000). A synthetic oscillatory network of transcriptional regulators. *Nature* 403:335–338.
 18. Fall CP, Marland ES, Wagner JM, Tyson JJ (2002). *Computational Cell Biology*. Springer-Verlag.
 19. Fell DA, Sauro HM (1985a). Metabolic control analysis: additional relationships between elasticities and control coefficients. *Eur J Biochem* 148:555–561.
 20. Fell DA, Sauro HM (1985b) Substrate cycles: do they really cause amplification? *Biochem Soc Trans* 13:762–763.
 21. Fell DA, Sauro HM (1990). Metabolic control analysis: the effects of high enzyme concentrations. *Eur J Biochem* 192: 183–187.
 22. Field RJ, Koros E, Noyes RM (1972). Oscillations in chemical systems, Part 2. Thorough analysis of temporal oscillations in the bromate-cerium-malonic acid system. *J Am Chem Soc* 94:8649–8664.
 23. Field RJ, Noyes RM (1974). Oscillations in chemical systems. IV. Limit cycle behavior in a model of a real chemical reaction. *J Chem Phys* 60(5):1877–1884.
 24. FitzHugh R (1955). Mathematical models of threshold phenomena in the nerve membrane. *Bull Math Biophys* 17:257–278.
 25. Gardner TS, Cantor CR, Collins JJ (2000). Construction of a genetic toggle switch in *Escherichia coli*. *Nature* 403:339–342.
 26. Geva-Zatorsky N, Rosenfeld N, Itzkovitz S, Milo R, Sigal A, Dekel E, Yarnitzky T, et al. (2006). Oscillations and variability in the p53 system. *Mol Syst Biol* 2:2006–2006.
 27. Gillespie DT (1976). A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. *J Comp Phys* 22:403–434.
 28. Gillespie DT Exact stochastic simulation of coupled chemical reactions. *J Phys Chem* 81:2340–2361.
 29. Goldbeter A (1997). *Biochemical Oscillations and Cellular Rhythms: The Molecular Bases of Periodic and Chaotic Behaviour*. Cambridge University Press;
 30. Goldbeter A, Koshland DE (1981). An amplified sensitivity arising from covalent modification in biological systems. *Proc Natl Acad Sci* 78:6840–6844.
 31. Goldbeter A, Koshland DE (1984). Ultra-sensitivity in biochemical systems controlled by covalent modification. Interplay between zero-order and multistep effects. *J Biol Chem* 259:14441–14447.
 32. Goodwin B (1965). Oscillatory behaviour in enzymatic control processes. *Adv Enzyme Regul* 3:425–438.

33. Hearon JZ (1952). Rate behavior of metabolic reactions. *Physiol Rev* 32:499–523.
34. Heinisch J (1986). Isolation and characterisation of the two structural genes coding for phosphofruktokinase in yeast. *Mol Gen Genet* 202:75–82.
35. Heinrich R, Rapoport TA (1974a). A linear steady state treatment of enzymatic chains. Critique of the crossover theorem and a general procedure to identify interaction sites with an effector. *Eur J Biochem* 42:97–105.
36. Heinrich R, Rapoport TA (1974b). A linear steady-state treatment of enzymatic chains; general properties, control and effector strength. *Eur J Biochem* 42:89–95.
37. Heinrich R, Schuster S (1996). *The Regulation of Cellular Systems*. Chapman and Hall.
38. Higgins J (1965). Dynamics and control in cellular systems. In B. Chance, R. W. Estabrook, and J. R. Williamson (Eds.), *Control of Energy Metabolism*, pp. 13–46. Academic Press.
39. Higgins J (1967). The Theory of Oscillating Reactions. *Ind Eng Chem* 59(5):18–62.
40. Higgins JJ (1959). Ph. D. thesis: A theoretical study of the kinetic properties of sequential enzyme reactions, Univ. Pennsylvania.
41. Hoffmann A, Levchenko A, Scott ML, Baltimore D (2002). The Ikappa B-NF-kappa B signaling module: temporal control and selective gene activation. *Science* 298:1241–1245.
42. Hofmeyr JHS (1986). Steady state modelling of metabolic pathways: a guide for the perspective simulator. *Comp Appl Biosci* 2:5–11.
43. Hofmeyr JHS (2001). Metabolic Control Analysis in a Nutshell. In *Proceedings of the Second International Conference on Systems Biology*. Caltech.
44. Hofmeyr JS, Cornish-Bowden A (2000). Regulating the cellular economy of supply and demand. *FEBS Lett* 476(1–2):47–51.
45. Ingalls BP (2004). A frequency domain approach to sensitivity analysis of biochemical systems. *J Phys Chem B* 108:1143–1152.
46. Ingolia NT (2004). Topology and robustness in the Drosophila segment polarity network. *PLoS Biol* 2(6):805–815.
47. Izhikevich EM (2007). *Dynamical systems in neuroscience: the geometry of excitability and bursting*. MIT Press.
48. Kacser H (1983). The control of enzyme systems in vivo: elasticity of the steady state. *Biochem Soc Trans* 11:35–40.
49. Kacser H, Burns JA (1973). The Control of Flux. In D. D. Davies (Ed.), *Rate Control of Biological Processes*, Volume 27 of *Symp Soc Exp Biol*, pp. 65–104. Cambridge University Press.
50. Kacser H, Burns JA (1981). The molecular basis of dominance. *Genetics* 97(3–4): 639–666.
51. Kaern M, Weiss R (2006). Synthetic Gene Regulatory Systems. In J. S. Zoltan Szallasi and V. Periwal (Eds.), *System Modeling in Cellular Biology From Concepts to Nuts and Bolts*, Chapter 13, pp. 269–298. MIT Press.
52. Kholodenko BN (2006). Cell-signalling dynamics in time and space. *Nat Rev Mol Cell Biol* 7(3):165–176.
53. Klipp E, Herwig R, Kowald A, Wierling C, Lehrach H (2005). *Systems Biology in Practice*. Wiley-VCH Verlag.
54. Lahav G, Rosenfeld N, Sigal A, Geva-Zatorsky N, Levine AJ, Elowitz MB, Alon U (2004). Dynamics of the p53-Mdm2 feedback loop in individual cells. *Nat Gen* 36(2):147–150.
55. LaPorte DC, Walsh K, Koshland DE (1984). The branch point effect. Ultrasensitivity and subsensitivity to metabolic control. *J Biol Chem* 259(22):14068–14075.
56. Laurent M, Kellershohn N (1999). Multistability: a major means of differentiation and evolution in biological systems. *TIBS* 24:418–422.
57. Levandoski MM, Tsodikov OV, Frank DE, Melcher SE, Saecker RM, Record MT, Jr (1996). Cooperative and anticooperative effects in binding of the first and second plasmid Osym operators to a LacI tetramer: evidence for contributions of non-operator DNA binding by wrapping and looping. *J Mol Biol* 260(5):698–717.
58. Lotka AJ (1920). Undamped oscillations derived from the law of mass action. *J Am Chem Soc* 42:1595–1599.
59. Mangan S, Alon U Structure and function of the feed-forward loop network motif. *Proc Natl Acad Sci USA* 100(21):11980–11985.
60. Markevich NI, Hoek JB, Kholodenko BN (2004). Signaling switches and bistability arising from multisite phosphorylation in protein kinase cascades. *J Cell Biol* 164:353–359.
61. Mettetal JT, Muzzey D, Pedraza JM, Ozbudak EM, van Oudenaarden A (2006). Predicting stochastic gene expression dynamics in single cells. *Proc Natl Acad Sci USA* 103(19):7304–7309.

62. Morales MF (1921). A note on limiting reactions and temperature coefficients. *J Cell Comp Physiol* 30:303–313.
63. Nicolis G (1971). Stability and dissipative structures in open systems far from equilibrium. *Adv Chem Phys* 19:209–324.
64. Ortega F, Garcés JL, Mas F, Kholodenko BN, Cascante M (2006). Bistability from double phosphorylation in signal transduction. *FEBS J* 273(17):3915–3926.
65. Paladugu S, Chickarmane V, Deckard A, Frumkin J, McCormack M, Sauro H (2006). In silico evolution of functional modules in biochemical networks. *IEE Proc-Systems Biol* 153:223–235.
66. Paulsson J, Elf J (2006). Stochastic Modeling of Intracellular Kinetics. In J. S. Zoltan Szallasi and V. Periwal (Eds.), *System Modeling in Cellular Biology From Concepts to Nuts and Bolts*, Chapter 8, pp. 149–175. MIT Press.
67. Pomerening JR, Sontag ED, Ferrell JE (2003). Building a cell cycle oscillator: hysteresis and bistability in the activation of Cdc2. *Nat Cell Biol* 5(4):346–351.
68. Rao CV, Sauro HM, Arkin AP (2004). Putting the ‘Control’ in Metabolic Control Analysis. 7th International Symposium on Dynamics and Control of Process Systems, DYCOPS, 7.
69. Reder C (1988). Metabolic control theory: a structural approach. *J Theor Biol* 135:175–201.
70. Reder C, Mazat JP (1988). Aspects géométriques de la théorie du contrôle du métabolisme in: Le contrôle du métabolisme. Master’s thesis, Bordeaux.
71. Reich JG, Selkov EE (1981). Energy metabolism of the cell. Academic Press.
72. Sauro HM (1994). Moiety-conserved cycles and metabolic control analysis: problems in sequestration and metabolic channelling. *BioSystems* 33:15–28.
73. Sauro HM, Ingalls B (2004). Conservation analysis in biochemical networks: computational issues for software writers. *Biophys Chem* 109(1):1–15.
74. Sauro HM, Kholodenko BN (2004). Quantitative analysis of signaling networks. *Prog Biophys Mol Biol* 86:5–43.
75. Savageau MA (1972). The behaviour of intact biochemical control systems. *Curr Topics Cell Reg* 6:63–130.
76. Savageau MA (1974). Optimal design of feedback control by inhibition: steady-state considerations. *J Mol Evol* 4:139–156.
77. Savageau MA (1976). Biochemical systems analysis: a study of function and design in molecular biology. Addison-Wesley.
78. Scott M, Ingalls B, Kærn M (2006). Estimations of intrinsic and extrinsic noise in models of nonlinear genetic networks. *Chaos* 16(2):26107–26107.
79. Small JR, Fell DA (1990). Covalent modification and metabolic control analysis: modification to the theorems and their application to metabolic systems containing covalently-modified enzymes. *Eur J Biochem* 191:405–411.
80. Stucki JW (1978). Stability analysis of biochemical systems – a practical guide. *Prog Biophys Mol Biol* 33:99–187.
81. Thomas S, Fell DA (1994). MetaCon – a program for the algebraic evaluation of control coefficients of arbitrary metabolic networks. In H. V. Westerhoff (Ed.), *Biothermokinetics*, pp. 225–229. Intercept Ltd.
82. Tyson J, Othmer HG (1978). The dynamics of feedback control circuits in biochemical pathways. In R. Rosen and F. M. Snell (Eds.), *Progress in Theoretical Biology*, 5:1–62.
83. Tyson JJ (1975). Classification of instabilities in chemical reaction systems. *J Chem Phys* 62:1010–1015.
84. Tyson JJ, Chen KC, Novak B (2003). Sniffers, buzzers, toggles and blinkers: dynamics of regulatory and signaling pathways in the cell. *Curr Opin Cell Biol* 15:221–231.
85. Vallabhajosyula RR, Chickarmane V, Sauro HM (2006). Conservation analysis of large biochemical networks. *Bioinformatics* 22(3):346–353.
86. van der Pol B, van der Mark J (1928). The heartbeat considered as a relaxation oscillation, and an electrical model of the heart. *Philos Mag Suppl* 6:763–775.
87. Voit E, Neves AR, Santos H (2006). The intricate side of systems biology. *Proc Natl Acad Sci USA* 103(25):9452–9457.
88. Westerhoff HV, Chen YD (1984). How do enzyme activities control metabolite concentrations? *Eur J Biochem* 142:425–430.
89. Wilkinson DJ (2006). Stochastic Modelling for Systems Biology. Chapman and Hall.

Chapter 14

Kinetic Modeling of Biological Systems

Haluk Resat, Linda Petzold, and Michel F. Pettigrew

Abstract

The dynamics of how the constituent components of a natural system interact defines the spatio-temporal response of the system to stimuli. Modeling the kinetics of the processes that represent a biophysical system has long been pursued with the aim of improving our understanding of the studied system. Due to the unique properties of biological systems, in addition to the usual difficulties faced in modeling the dynamics of physical or chemical systems, biological simulations encounter difficulties that result from intrinsic multi-scale and stochastic nature of the biological processes.

This chapter discusses the implications for simulation of models involving interacting species with very low copy numbers, which often occur in biological systems and give rise to significant relative fluctuations. The conditions necessitating the use of stochastic kinetic simulation methods and the mathematical foundations of the stochastic simulation algorithms are presented. How the well-organized structural hierarchies often seen in biological systems can lead to multi-scale problems and the possible ways to address the encountered computational difficulties are discussed. We present the details of the existing kinetic simulation methods and discuss their strengths and shortcomings. A list of the publicly available kinetic simulation tools and our reflections for future prospects are also provided.

Key words: Biological network, kinetic simulation, stochastic simulation algorithm, tau-leaping, hybrid kinetic model, simulation software.

1. Background and Introduction

Reactions occurring among a set of reactants define a kinetic reaction network. Traditionally, the set of reactions is described by nonlinear ordinary or partial differential equations, solutions of which provide insights into the dynamics of the studied processes. The size of such networks can vary considerably. In some instances, hundreds of thousands of reactions may be needed to

describe the complex, coupled phenomena. Although kinetic modeling and the underlying fundamental mathematics are used in vastly different fields of science and engineering, here we will concentrate on biological kinetic networks that model cellular activities as a set of reaction processes. Biological networks are intrinsically very rich in character; feedback loops are common, and bifurcations and oscillations can lead to interesting dynamical behavior. Although there are many commonalities, two aspects, in particular, distinguish biological networks from networks found in other fields. Firstly, in biological systems, copy numbers of many species are very low, which can give rise to significant relative fluctuations. Secondly, biological systems tend to have well-organized structural hierarchies.

The traditional way of modeling the time evolution of the molecular populations in a reacting system is to use a set of coupled, first-order, ordinary differential equations (ODEs) called the *reaction rate equations* (RREs). RREs are usually inferred from phenomenological arguments under the assumption that the system is *well stirred* or *spatially homogeneous*, that is, no persistent correlations develop among the positions of the molecules. RRE is nonlinear in the presence of bimolecular reactions, which is almost always the case. Owing to the possibility of widely differing reaction rates, RRE can often exhibit vastly different time scales, a condition known as *dynamical stiffness*. Sometimes, under certain simplifying assumptions such as quasi steady-state or partial equilibrium, RREs reduce to a system of differential-algebraic equations (DAEs).

However, a deterministic approach alone may not be sufficient for cellular systems when the underlying problem is inherently stochastic. Fluctuations in the concentrations of biomolecules can be quite large and this may require special treatment in modeling the biological networks (1–4). This is particularly true for the molecular species involved in the transcriptional regulation of gene expression. In certain such cases, as the small copy number limit is reached, differential equation-based models break down since the assumption of continuous molecular concentrations is no longer valid. In such cases a more general approach, based on stochastic methods in which natural fluctuations are treated in a direct and explicit manner, has been advocated for the quantitative analysis of biological networks (5–7).

A simple argument demonstrates why fluctuations can be considerable in cellular systems. The typical diameter of a microbial cell is a few micrometers (μm), which corresponds to a cellular volume on the order of ~ 10 femtoliter (fl). In this volume, the concentration of 1 molecule is roughly equal to 160 picomolar (pM). This discrete nature is, of course, more severe when the molecules are confined to even smaller substructures, which is typically the case in eukaryotic cells. Thus, given that typical

binding affinities of the biomolecules are often in the picomolar to micromolar range, and ignoring whether the thermodynamics are still valid at this limit, even very small fluctuations in the molecule copy numbers can cause a regime change in biomolecular interactions. Therefore, the fluctuations resulting from a few discrete reactions can significantly affect the dynamical patterns in cellular systems. In a series of papers Gillespie showed the equivalency of deterministic and stochastic formalisms in the limit as the populations of all constituent chemical species tend to infinity, and discussed the importance of stochastic fluctuations when the system sizes are small (1, 2). We note that stochastic differential equations are often used in physics, but they are usually obtained by starting with an assumed ODE and then adding a “noise” term that has been carefully crafted to give a preconceived outcome. Although such approaches are computationally less expensive, they do not capture the true nature of the intrinsic fluctuations. Therefore, the use of *discrete stochastic approaches* is often more suitable in studying biological systems.

Although they provide a more realistic modeling frame, discrete stochastic simulations are often too slow and computationally expensive. Building upon earlier fundamental work (1, 2, 8), the development of new mathematical methods and numerical algorithms have been pursued by many groups to overcome some of the computational bottlenecks (5, 6, 9–23). Development of approximate algorithms was also pursued. For example, the probability weighted dynamic Monte Carlo method (24) has been shown to enable the simulation of network models with tens of thousands of reactions and hundreds of compartments using desktop computers (25), and various tau-leaping algorithms, which allow the use of larger time steps with tolerable inexactness (26–32), can also lead to significant speed ups in the stochastic simulations.

The spatial organization and structural hierarchies that often occur in biological systems provide an additional degree of complexity. Information transfer and biochemical activity can depend on location in cellular systems. Indeed, spatial organization becomes particularly important when the kinetics of the interaction networks depend on the local environmental factors, i.e., when they depend on local conditions and the environment or when material transport is required to convey the information. A good example to this is syntrophic (mutually interdependent) microbial systems, in which organisms in the community depend on each other for metabolite use and production. Here the metabolic activity necessary for cellular growth and survival can depend on the substrate availability and favorable environmental factors that are jointly established by multiple organisms. The relative concentration of the organisms and their geometrical arrangement can lead to heterogeneous distributions reflecting

local variations in the microbial composition. Thus, the dynamical behavior of microbes in syntrophic systems relies on the combination of highly localized molecular events (i.e., metabolism) and macroscopic transport of products over a much larger spatial scale (i.e., between the cells of syntroph organisms). An analogous situation arises in mammalian systems, where eukaryotic cells have compartments with specific functionalities. These subcellular compartments exchange material in a concerted fashion as part of fundamental cellular processes such as synthesis, degradation, and receptor signaling. At the same time, the milieu of the cell within its tissue group is a consequence of the tissue itself. Thus, compartmentalized reactions, though many spatial scales smaller than the tissue in which they are found, simultaneously drive and respond to changes in the tissue. This hierarchy of compartmentalized activity represents a key challenge in multi-scale modeling.

Particle transport dynamics can also play a central role in transferring information in many biological problems (33). This is particularly true when messenger substrate molecules or interacting proteins are transported over distances many orders of magnitude larger than their size. For example, energy metabolism of many bacteria requires cytochrome shuffling as part of the electron transfer mechanism. Oxygen availability and penetration can be very important in bacterial growth, both in batch reactors and in biofilms. Plant ecology too involves location-specific activities. Most plants allow their root nodule cells to be infected by microbes or fungi for arbuscule development, which is used to obtain nutrients (particularly phosphorus) from the soil. At the same time, plants use their leaves for the necessary carbon production, some of which is also transferred to the microbial/fungal mycorrhizal symbiotic (intimate interdependent) partner. Material transport also plays a central role in determining cellular response in higher organisms, for which stress-induced Ca^{++} /cAMP release in mammalian systems would be a good example (33).

Recent progress in experimental techniques, particularly fluorescent labeling and high-resolution microscopy, is beginning to make it possible to collect biological information about local dynamical activities (see, for e.g., Refs. (34–38)). Models that represent dynamics using spatially averaged quantities cannot capture the effects due to local inhomogeneities. For the types of problems described above, computational biology studies aiming to complement the experimental efforts require the development and analysis of spatially resolved multi-scale models. The standard chemical master equation (CME) and stochastic simulation algorithm (SSA) assume that the system is well stirred or spatially homogeneous (1, 4), which for some problems might be an enormous simplification. Currently, the strategy

most likely to work appears to be the *locally homogeneous* approach, which dates back to at least the mid-1970s (39, 40). In this approach, the system volume is subdivided into K subvolumes that are small enough for each to be considered spatially homogeneous. Each subvolume forms a *reactor compartment*, and the compartments are linked at the higher whole-system scale. Using this framework, whenever a chemical reaction occurs in the system we now know where it occurred, at least to within the resolution of the subdivisions. However, this spatial resolution comes at a considerable cost in computational complexity. The dimensionality of the state space and the number of chemical reaction channels have been increased by a factor of K , the number of subvolumes. Moreover, since a large set of reactions describing *the diffusive transfer between subvolumes* is needed to complete the model, the increase in computational complexity is in fact even greater.

From a purely mathematical point of view, the locally homogeneous approach to a spatially inhomogeneous system is essentially the same as for a spatially homogeneous system, in that one winds up with a CME and an SSA of the same “jump Markov” form (4, 41, 42). However, the resulting network model has a much higher dimensional state space and consists of a much larger set of reaction channels. The challenge of determining how to divide the system into subvolumes efficiently and without sacrificing the details of the problem must be overcome before this model can be routinely implemented. Another challenge is the development of algorithms that scale efficiently with the model dimension, as it would be necessary to handle the complexity resulting from the steep increase in network sizes. For large systems, the conspicuous absence of algorithms and software capable of taking advantage of available computing resources indicates a need for the development of hybrid algorithms in which the problem is separated into deterministic and stochastic parts to increase computational efficiency, without sacrificing scientific veracity.

While both deterministic and stochastic formulations are currently used in the kinetic modeling of biological systems, as has been mentioned above, discrete stochastic methods are more suitable for biological systems. With this in mind, and because the numerical simulation of ODE models is a well-established area with excellent and available software, in the remainder of this chapter we concentrate on stochastic simulation methods. We first summarize the fundamental steps and the mathematical foundations for kinetic simulation of reaction networks. We follow that with a discussion of shortcomings and size scaling properties of existing stochastic simulation algorithms. We then conclude by briefly highlighting the future research trends from our perspective.

2. Simulation Methods

2.1. General Description

Kinetic simulation of biochemical systems involves the following basic steps.

1. Identification of the kinetic problem: This step involves the determination of the input and output variables as well as the intermediates. Key species and physical properties to be simulated are determined and tabulated.
2. Model formulation: Although that does not have to be the case, most studies employ RREs to model biochemical systems. In RRE, one simply defines the changes in the concentrations (or equivalently the number of molecules) as a function of time and location. We note that constraints can be embedded in RRE using a Lagrange undetermined multipliers type formalism.
3. Choosing a method: At this step a decision needs to be made as to whether to adopt a deterministic or stochastic formulation. Although it is still relatively uncommon, a hybrid approach that moves between deterministic and stochastic regimes is another possibility. In the deterministic approach, a reasonable time step is chosen adaptively, based on the estimated local error according to a differential equation model. In contrast, discrete stochastic simulations model each individual reaction event. Using the probabilities of the reactions (called the *reaction propensities*), one determines the occurrence statistics of the involved reactions. This is generally done in one of two ways: In next reaction type methods, the sequence of the reactions are chosen according to the reaction propensities and the reaction times are computed to determine the elapsed time. In lumped reaction approaches, one can choose the size of the elapsed time step and compute how many times the involved reactions occur according to their probabilities.
4. Simulation: Integrating RRE deterministically involves choosing an appropriate algorithm from a wide selection of sophisticated numerical methods for systems of ODEs and DAEs. Highly efficient and reliable software is readily available, although one must usually determine whether or not the problem is stiff and then choose the software accordingly. Roughly speaking, an ODE or DAE system is stiff if it involves a wide range of time scales and the fastest of those scales correspond to stable processes. In chemically reacting systems, the wide range of time scales can come from some reaction rates being several orders of magnitude or more greater than others. A solver that is designed for non-stiff

systems will run very slowly (because it needs to choose time steps on the scale of the fastest process in the system to maintain stability for its explicit formulas), if it is applied to a stiff problem. The cure for stiffness is to approximate the differential equation using implicit methods. The software that is available for the accurate solution of stiff ODE and DAE systems always makes use of implicit methods. For more information on ODE and DAE methods and software, see Ascher and Petzold (43). Stochastic simulation algorithms are not as advanced as the deterministic integrators and they do not scale well with the model size and complexity. Even though recent algorithmic improvements have led to impressive gains, stochastic simulations are still too slow for most realistic systems. However, as discussed in the previous section, fluctuations can be large in biological systems and, more importantly, stochastic variations may have implications for biological functions and responses. Therefore, when feasible, stochastic simulation methods should be preferred. This is particularly true for processes that involve small number of regulatory molecules with large variations in their expression levels.

5. Trajectory analysis: Time courses obtained during the simulations are catalogued and combined to derive the statistical distributions of the desired output/prediction quantities, such as the concentrations of experimentally monitored species and the resulting material/mass flow in the system or the probability distribution of the reactions. Similarly, from simulation results for different model parameter sets, one can study the sensitivity to parameter variations.

The following simple example illustrates Steps (1–3). Assume that we are studying a receptor (R) system that binds a ligand L. The ligand-bound receptors (RL) binds an effector/adaptor/scaffold molecule E that helps with signal transduction. Receptors that are in complex with the effector molecules are labeled as RLE. For this case, the kinetic system consists of the following two reversible reactions $L+R \leftrightarrow RL$ and $RL+E \leftrightarrow RLE$. Forward and reverse reactions for these two reactions are second- and first-order reactions, respectively. For simplicity, let us use a mass-action formalism where the rates for the first- ($A \rightarrow C$) and second- ($A+B \rightarrow C$) order reactions are represented as $k[A]$ and $k[A][B]$, respectively, where $[X]$ stands for the concentration (or the number of molecules with the proper volume conversion) of species X.

In this example, the species list would be L, R, RL, E, and RLE. The reaction list will contain four reactions (i) $L+R \rightarrow RL$, (ii) $RL \rightarrow L+R$, (iii) $RL+E \rightarrow RLE$, and (iv) $RLE \rightarrow RL+E$. If RLE is the species of interest (signal transducing receptors), the output of the simulations would be $[RLE]$, the concentration of the species RLE. This identification completes Step (1).

Development of RRE follows a simple rule. Reactions that decrease or increase the concentration of a species appear as a source term in the rate equation for that species. For example, the reaction $RL+E \rightarrow RLE$ decreases $[RL]$ and $[E]$ and increases $[RLE]$. So the term $k_3[RL][E]$ is included as a negative contribution in the rate equations for $d[RL]/dt$ and $d[E]/dt$, and as a positive contribution in the $d[RLE]/dt$ rate equation. Using this logic RRE equations for the above example would be:

$$\begin{aligned}d[L]/dt &= -k_1[L][R] + k_2[RL] \\d[R]/dt &= -k_1[L][R] + k_2[RL] \\d[RL]/dt &= +k_1[L][R] - k_2[RL] - k_3[RL][E] + k_4[RLE] \\d[E]/dt &= -k_3[RL][E] + k_4[RLE] \\d[RLE]/dt &= +k_3[RL][E] - k_4[RLE]\end{aligned}$$

Once the model is defined as above, preparation for the simulation is slightly different depending on the method to be used. If a deterministic approach is chosen, then RRE is investigated in terms of stiffness of the equations, and an appropriate integrator is chosen. We note that for more general formalisms singularity of the Jacobian is another point of concern, but that subject is outside the scope of this chapter. If a stochastic simulation approach is chosen, then RRE is converted into a reaction table (Table 14.1) form where propensities are associated with each reaction.

In the stochastic approach, the stoichiometry matrix defines the change in the species concentrations when the corresponding reaction occurs. Entries for the reactants and products appear as negative and positive, respectively. With this information at hand, at every step of the stochastic simulation one can update the system configuration $\mathbf{x}(t)$ according to the occurrence of reactions as $\mathbf{x} \rightarrow \mathbf{x} + \mathbf{v} \rightarrow \omega$ using the information about how many times the reactions occur ω .

Table 14.1
Reactions and stoichiometry matrix defining the example model

	Reaction	Propensity	Stoichiometry (ν)				
			[L]	[R]	[RL]	[E]	[RLE]
(1)	$L+R \rightarrow RL$	$k_1[L][R]$	-1	-1	+1	0	0
(2)	$RL \rightarrow R+L$	$k_2[RL]$	+1	+1	-1	0	0
(3)	$RL+E \rightarrow RLE$	$k_3[RL][E]$	0	0	-1	-1	+1
(4)	$RLE \rightarrow RL+E$	$k_4[RLE]$	0	0	+1	+1	-1

Step (4), execution of the simulations, is the key step in the kinetic modeling studies. The numerical solution of ODEs or DAEs is a well-established area with widely available software (43) (also see **Section 3**). Thus, we will focus on stochastic simulation methods in the remainder of this chapter.

2.2. Stochastic Simulation Algorithms – Fundamentals

The Stochastic Simulation Algorithm (SSA) developed by Gillespie (1–4) and its variants are the basis for stochastic kinetic simulation of biological systems. In SSA it is assumed that successive *reactive* molecular collisions in the system are separated by many *non-reactive* collisions, which effectively wipe out all positional correlations among the molecules. Therefore, SSA is suitable for the simulation of well-stirred systems.

Defining the *state* of the system by the vector $\mathbf{x}(t)$ whose components $x_i(t)$ ($i=1$ to N for N species) denote the number of molecules of species S_i in the system at time t , one can establish from kinetic theory the following result. For each elemental chemical reaction channel R_j ($j=1, \dots, M$), there exists a function a_j such that if the system is in state $\mathbf{x}=(x_1, \dots, x_N)$ at time t , then the *probability* that reaction R_j will occur somewhere inside the system in the next infinitesimal time dt is $a_j(\mathbf{x})dt$. This can be regarded as the fundamental premise of stochastic chemical kinetics. The function a_j is called the *propensity function* of reaction channel R_j . Each reaction channel R_j is completely characterized by its propensity function together with its *state change vector* \mathbf{v}_j , which represents the change in the population of the molecular species caused by one R_j event, i.e., the stoichiometric coefficients of the reactions. Thus, if the system is in state \mathbf{x} and one R_j reaction occurs, the system jumps to state $\mathbf{x}+\mathbf{v}_j$. This is mathematically known as a *jump Markov process* and a great deal is known about such processes (8, 41).

The stochastic evolution of the state vector $\mathbf{x}(t)$ is specified by the function $P(\mathbf{x}, t | \mathbf{x}_0, t_0)$, defined as the *probability* that $\mathbf{x}(t)$ will equal \mathbf{x} given that $\mathbf{x}(t_0)=\mathbf{x}_0$ for $t_0 \leq t$. It can be proven that this function obeys the following time-evolution equation (4, 44):

$$\frac{\partial P(\mathbf{x}, t | \mathbf{x}_0, t_0)}{\partial t} = \sum_{j=1}^M [a_j(\mathbf{x} - \mathbf{v}_j)P(\mathbf{x} - \mathbf{v}_j, t | \mathbf{x}_0, t_0) - a_j(\mathbf{x})P(\mathbf{x}, t | \mathbf{x}_0, t_0)]. \quad [1]$$

This elegant equation is known as the *chemical master equation* (CME). Unfortunately, it is of such a large dimension (it includes a differential equation corresponding to every possible state of the system) that its solution is almost always intractable, both analytically and numerically.

An alternate way of analyzing the behavior of $\mathbf{x}(t)$ is to construct its unbiased realizations using SSA (1, 2). The steps of SSA basically are:

- i) Initialize the system's state $\mathbf{x}=\mathbf{x}_0$ at time $t=t_0$;
- ii) Evaluate the reaction propensities $a_j(\mathbf{x})$ and their sum $a_0(\mathbf{x})$;
- iii) Draw two random numbers r_1 and r_2 from a uniform distribution in the unit-interval, and compute the time interval between the reactions $\tau=-\ln(r_1)/a_0(\mathbf{x})$ and determine which reaction type R_j occurs next by finding the smallest integer j that satisfies $\sum_{j'=1}^j a_{j'}(\mathbf{x}) > r_2 a_0(\mathbf{x})$;
- iv) Record and update the changes in the system $t \rightarrow t+\tau$ and $\mathbf{x} \rightarrow \mathbf{x}+\mathbf{v}_j$; and
- v) Return to step (ii) or end the simulation.

The SSA advances the system in time from one reaction event to the next, and it does so in a way that is rigorously equivalent to the CME. The SSA is therefore an *exact* method for simulating the time evolution of a spatially homogeneous chemical system. Because SSA processes reaction events serially, i.e., one after another, it is often too slow to be practical when applied to real systems of interest. An extremely large molecular population of even one reactant species, and/or a very fast reaction, will usually cause the time increments τ to be extremely small.

2.3. Acceleration of Discrete Stochastic Simulation

Several lines of research have been pursued in attempts to improve upon the exact SSA. These include more efficient implementations of SSA and the development of approximate SSAs with improved numerical efficiencies.

The efficiency of the algorithms can strongly depend on the formulation and numerical implementation and these are the key aspects to consider when developing simulation software. Gibson and Bruck used an efficient implementation strategy for the exact SSA (10) to remove a key limitation, which restricted the practical use of the first reaction method of Gillespie (1), by not requiring the generation of N_r new random uniform deviates following a reaction event where N_r is the number of active reactions. Rather than a single waiting time between events, the method stores the absolute firing time for each reaction in an indexed priority binary heap queue, with times increasing with increasing depth in the heap. Assuming a static reaction network topology, a reaction dependency graph is created initially. Following an event, which is identified by the element at the top of the heap, a single random uniform deviate is generated, reaction propensities are efficiently updated using the dependency graph, and any absolute firing time requiring updating is adjusted through a linear transformation. Lastly, the heap queue is reordered. Gibson and Bruck have shown that the algorithm has a time complexity of order $O(N_r + N_E \log N_r)$, where N_E is the number of events and generally requires only

a single random uniform deviate per event (45). While the next reaction method can be efficient for large, very sparse networks, significant overhead may incur in maintaining the indexed priority queue. This has led Cao et al. to conclude that, for most kinetic systems, the most efficient formulation of SSA is an optimized direct Gillespie method (46).

One way to speed up the next reaction method type algorithms is to allow multiple reactions to take place in longer time steps, which is the underlying idea behind the probability-weighted dynamic Monte Carlo (PW-DMC) method (24) and the tau-leaping based algorithms (28–32, 47). The basic premise behind the PW-DMC method is that reactions with large propensities dominate the stochastic simulation. By attaching weights that are updated as the simulation progresses, the PW-DMC approach attempts to equalize the propensities of the reactions constituting the network model. In this process, the large propensities are scaled down with an associated factor, which corresponds to how many times the involved reaction occurs when selected. It was shown that the PW-DMC algorithm solves the multiple time scale problem to a large extent (24). Efficient statistical sampling obtained with the PW-DMC algorithm made the simulation of an integrated stochastic model of EGFR signaling network, which contains tens of thousands of reactions, possible using desktop computers in reasonable wall clock periods (25). The success of PW-DMC results from its effectiveness in bundling the occurring reactions and, in this aspect, it is conceptually similar to the approach pursued in tau-leaping methods. The main differences between the PW-DMC and the tau-leaping methods are that in PW-DMC the bundling of the reactions is done for each reaction type (i.e., some reactions can be selected to be left out if desired) and the integration time does not have to be set in advance, but the tolerance in the fluctuation levels for the species need to be predefined. These features make PW-DMC suitable to deal with the reaction networks that can be partitioned into subclasses according to their spatio-temporal properties where the allowed fluctuation levels can be dictated on a per reaction basis. It was shown (24) that, while the fluctuation variations may be affected, PW-DMC conserves the mean quantities. In this respect, PW-DMC is a weak order (i.e., not a strong order) stochastic method. The fluctuations in the computed quantities depend on the achieved speed-up, which can be several orders of magnitude without any significant effect on the computed mean quantities (24).

In the tau-leaping (τ L) methods (47), advances in the system configuration are determined based on a *pre-selected* time step τ during which more than one reaction event may occur. Tau-leaping is based on the same premises that underlie CME and

SSA. If τ is chosen small enough to satisfy the *leap condition*, which says that none of the propensity functions may change noticeably during τ , then given $\mathbf{x}(t)=\mathbf{x}$, the system state at time $t+\tau$ can be *approximately* computed as

$$\mathbf{x}(t + \tau) = \mathbf{x} + \sum_{j=1}^M P_j(a_j(\mathbf{x}), \tau) \mathbf{v}_j. \quad [2]$$

The P_j are statistically independent Poisson random variables; they physically represent the number of times each reaction channel fires in time τ . The Poisson random variable $P(a, \tau)$ is in fact defined as the number of events that will occur in time τ , given that $a dt$ is the probability of an event occurring in the next infinitesimal time dt . We emphasize that τ must be chosen small enough that none of the propensity functions changes significantly during the leap. Algorithms have been developed for estimating the largest value of τ that will satisfy the condition for a prescribed level of simulation accuracy (26, 28).

While the τ L methods show significant gains in efficiency, the unbounded Poisson distribution can lead to nonphysical system configurations in which negative species populations arise when the expected number of events is large enough to consume the reactant populations. Cao et al. have shown how to avoid negative populations in explicit Poisson τ L (P τ L) methods (48). The non-negative P τ L algorithm is based on the fact that negative populations typically arise from multiple firings of reactions that are only a few firings away from consuming all the molecules of one of their reactants. To focus on those reaction channels, the modified P τ L algorithm introduces a control parameter n_c , a positive integer that is usually set somewhere between 10 and 20. Any reaction channel that is currently within n_c firings of exhausting one of its reactants is then classified as a critical reaction. The algorithm chooses τ in such a way that no more than one firing of all the critical reactions can occur during the leap. Essentially, the algorithm simulates the critical reactions using SSA, and the remaining non-critical reactions using τ -leaping (48).

An alternative approach is based on choosing the reaction numbers using binomially distributed random variables that have the appropriate finite sampling ranges. If a short time interval τ_ϵ is considered, for each reaction R_j , the Poisson distribution is replaced with a binomial distribution $B(p_j, \omega_j^*)$ with the same mean $a_j\tau$. Here ω_j^* is an upper bound (or limit) on the reaction number and p_j is the probability that R_j will fire in the interval τ_ϵ/ω_j^* . In the resulting binomial tau-leaping (B τ L) method, the leap size is then chosen as the smaller of the times t_j computed from the constraints $p_j = a_j t_j / \omega_j^* \leq 1$, $j=1:M$ and the upper bound τ_ϵ . Once the tau step has been determined, each network reaction number ω_j is generated by sampling the corresponding binomial distribution.

Although the binomial distribution has the desirable property of having a finite sampling range, this in itself does not completely resolve the non-negativity problem. In networks with multiple channel reactant dependencies, the interdependence arising from species participating in multiple reactions may still generate negative species populations. Tian and Burrage have shown how the B τ L completely resolves the non-negativity problem when there is at most a two-reactant dependence in a reaction network (31). In the B τ L method of Chatterjee et al. (29, 30), the issue of multiple-channel reactant dependencies is dealt with by assigning to each reaction a binomial random variable for the number of reaction events during a step and by modifying the reaction number limits through tracking currently available population sizes as reactions fire during that step. Yet, as critically noted by Chatterjee et al., in the latter approach selected reaction numbers may depend on the order in which reactions are processed (29, 30).

The multinomial tau-leaping (M τ L) method of Pettigrew and Resat efficiently extends the B τ L method to networks with arbitrary multiple-channel reactant dependencies (32). Improvements were achieved by a combination of three factors: First, tau-leaping steps are determined simply and efficiently using a priori information and Poisson distribution-based estimates of expectation values for reaction numbers. Second, networks are partitioned into closed groups of reactions and corresponding reactants in which no group reactant set is found in any other group. Third, product formation is factored into upper bound estimation of the number of times a particular reaction occurs. Together, these features allow for larger time leaping steps, while the numbers of reactions occurring simultaneously in a multi-channel manner are estimated accurately using a multinomial distribution (32). Partitioning of the reaction network into groups with closure can be done with a graph theoretic algorithm based on the Dulmage-Mendelsohn decomposition of the stoichiometric matrix representing the network. Two methods for selecting an upper limit Ω^* on the system reaction number was advocated, the rate-limiting reactant and the rejection methods (32). The former is a conservative approach, which guarantees that the chosen reactions do not lead to negative populations after the system's configuration is updated according to the occurring reactions. In contrast, the selection process in the rejection method only guarantees that the system configuration after the tau-leaping step is non-negative. It however does not ensure that all possible multi reaction paths that take the system to that final configuration involve only physically feasible intermediate stages. It was argued that, since one can only detect the relationship between the starting and end points of a move during the simulation, ensuring the state of the system to always correspond to physically feasible configurations at the

observation points (i.e., starting and end points of the moves in a simulation run) is the most crucial aspect in obtaining accurate results (32). Therefore, allowing a simulation trajectory to temporarily cross the solution space boundary would introduce insignificant inaccuracies because statistics for intermediate states have only an indirect impact. In the M τ L method, tau-leaping step is then chosen as the smaller of the times t computed from the constraint $p = t\Sigma/\Omega^* \leq 1$ and the upper bound τ_ϵ where Σ is the system propensity. The actual system reaction number Ω is then sampled from a binomial distribution as discussed above, and the reaction numbers ω_j , $j=1:M-1$, are drawn from a multinomial distribution with corresponding probabilities $\alpha_j = a_j/\Sigma$ and $\omega_M = \Omega - \omega_1 - \omega_2 - \dots - \omega_{M-1}$. Implemented advances over the B τ L method were shown to lead to significant speed-ups in the simulations of biologically relevant models (32). It was also shown that M τ L is less sensitive to the error parameter of the τ L methods, which is a desirable aspect of the kinetic simulation algorithms.

2.4. Thermodynamic Limit

It is well known that the *Poisson* random variable $P(a, \tau)$, which has mean and variance $a\tau$, can be approximated by a *normal* random variable with the same mean and variance when $a\tau \gg 1$. Using this fact, it can be shown that if τ satisfies not only the leap condition but also the conditions $a_j(x)\tau \gg 1$ for *all* j , which physically means that during time τ every reaction channel can be expected to fire many more times than once, then the basic tau-leaping Eq. [2] further approximates to (49)

$$X(t + \tau) = x + \sum_{j=1}^M v_j a_j(x) \tau + \sum_{j=1}^M v_j \sqrt{a_j(x) \tau} N_j(0, 1). \quad [3]$$

Here the $N_j(0,1)$ are statistically independent normal random variables with mean 0 and variance 1. This modified update formula is computationally faster than the tau-leaping formula Eq. [2], which it approximates, because samples of the normal random variable are much easier to generate than samples of the Poisson random variable. Also, since by hypothesis every reaction channel fires many more times than once during each time step τ , simulations using the updating Eq. [3] will be *very* much faster than SSA.

If the arguments leading to Eq. [3] are repeated with τ replaced by a *macroscopic infinitesimal* dt , which by definition is *small* enough that no propensity function changes significantly during dt yet *large* enough that every reaction channel fires many more times than once during dt , then one can deduce the equation

$$\frac{dx(t)}{dt} = \sum_{j=1}^M v_j a_j[x(t)] + \sum_{j=1}^M v_j \sqrt{a_j[x(t)]} \Gamma_j(t), \quad [4]$$

where the gammas are statistically independent *Gaussian white noise* processes (49–51). This equation is called the *chemical Langevin equation* (CLE), and it is an example of what is known to mathematicians as a *stochastic differential equation*. It is entirely equivalent to the *Langevin leaping formula* given by Eq. [3].

We note that stochastic differential equations are often used in physics, but they are usually obtained by starting with an assumed ODE and then adding a “noise” term that has been carefully crafted to give some preconceived outcome. So it is noteworthy that the noise term in Eq. [4], namely the second summation there, has actually been *derived* from the same premises that underlie the CME and the SSA using physically transparent assumptions and approximations. Use of the Langevin formulas (3) and (4) is typically justified whenever all reactant species are present in sufficiently large numbers.

At the *thermodynamic limit*, molecular populations of all the reactants *and* the system volume become infinitely large, while the concentrations of all species stay constant. It can be proven that, in this limit, all propensity functions scale *linearly* with the system size. Thus, the deterministic terms on the right sides of Eqs. [3] and [4] grow in proportion to the system size, while the fluctuating terms grow in proportion to the *square root* of the system size. This is another way of showing that for large systems “relative fluctuations in the species populations scale as the inverse square root of the system size,” a central rule of thermodynamics. In the *full thermodynamic limit*, the foregoing scaling behavior implies that the fluctuating terms in Eqs. [3] and [4] become negligibly small compared to the deterministic terms, and the CLE reduces to

$$\frac{dx(t)}{dt} = \sum_{j=1}^M v_j a_j[x(t)]. \quad [5]$$

This is none other than the traditional RRE, which is more commonly used in terms of the concentration vector. Note that the RRE has not been assumed here; instead, it has been *derived* from the same premises that underlie the CME and the SSA, using a series of physically understandable approximations (49, 50). Thus, the discrete, stochastic CME/SSA and the continuous, deterministic RRE are actually connected across the ends of representation spectrum. The challenge, of course, is to develop both the software infrastructure and the theory necessary to make reliable adaptive decisions so that we can accurately and efficiently simulate complex reaction systems based on detecting a discrete or continuous regime and using an optimal blend of stochastic and deterministic methods.

2.5. Scaling Properties with Respect to the Network Size

In addition to the multi-scale nature of the involved kinetic processes, computational efficiency of stochastic simulation algorithms also depend on several measures of the size of the investigated network model, such as the overall complex population, the total number of reactions, and the average number of nodal interactions or connectivity in a network. Understanding how algorithms scale with size and nodal connectivity is an important consideration in the design of numerical experiments for dynamic biological systems. As the complexity of the systems studied in biological systems increases, such scaling issues become a serious point of concern.

To study how stochastic kinetic simulation algorithms scale with network size, the construction of scalable reaction models that can be used in benchmark studies has been advocated (42). Pettigrew and Resat used two scalable reaction models to compare the efficiency of the exact Gillespie algorithm as implemented using the popular Gibson-Bruck method (GGB) (10) with the inexact random substrate method of Firth and Bray (FB) (52). Foundations of these two algorithms are quite distinct from each other and the latter algorithm has been stated to be an efficient method when the total complex population is small and the number of multi-state interactions large.

The first investigated scalable model, the Size Scalable Model (SSM), is a four major-compartment model that is typical for a signal transduction network in eukaryotic cells (42). The SSM involves two unimolecular and five bimolecular reversible reactions with mass transfer between compartments for eight complex types including ligands, adapter complexes, and receptors (42). Upon phosphorylation the receptor can become active in signal transduction. A sub-compartmentalization process, which is equivalent to forming smaller locally homogeneous regions, allows for the creation of refined models where the size of the model increases in proportion to the number of created subcompartments while maintaining the connectivity (i.e., the topology) between the species. For SSM, it was found that the GGB algorithm, whose numerical efficiency scales with the logarithm of the total number of active reactions, performs significantly better in all cases including those characterized by a very small number (~ 100) of complex types, each with a very small population (42).

The second scalable model, the Variable Connectivity Model (VCM), is a single compartment signal transduction network involving two unimolecular and two bimolecular reversible reactions for three complex types: ligand, receptor, and ligand-receptor complexes (42). The receptors in this model have multiple phosphorylation sites. The size and the node connectivity between the species increase in proportion to the number of phosphorylation sites N . For the VCM, it was shown that the FB random substrate method, whose

efficiency scales with the square of the total complex population, can outperform the GGB algorithm when the node connectivity is sufficiently high and the total complex population sufficiently low. Performance of the FB and GGB algorithms were determined as a function of the number of phosphorylation sites for two distinct population sets. In the first set the total complex population was initially set at 192 complexes (128 ligands and 64 receptors), while for the second set the total complex population was 10 times higher with the 2:1 ratio of ligands to receptors unchanged. It was found that, for the smaller population set, the FB algorithm becomes more efficient than the GGB when the number of phosphorylation sites N reaches 6, whereas in the larger population set the GGB is clearly superior for small N up to a cross-over point that occur around $N \sim 11$. Thus, it was concluded that with the exception of special cases, GGB is a more efficient method than the FB (42).

This simple study using two sample realistic networks clearly illustrated that the most efficient algorithm can depend on the problem type and the used simulation method should be chosen after a careful study of the topology and multi-scale properties of the investigated network.

3. Simulation Tools

Table 14.2 lists and briefly describes some of the existing software tools that may be used in the kinetic simulations of biological systems. We would like to note that the list is in no way complete and it only represents a small percentage of the available tools.

4. Future Prospects of Stochastic Simulation

In recent years there has been rapid progress in the analysis and development of algorithms for the accelerated discrete stochastic simulation of chemically reacting systems. Although these developments provide a good foundation, existing stochastic simulation methods are far from being computationally efficient to simulate detailed biological models. This is particularly true for the models that involve multiple time and spatial scales while still requiring high resolution for accurate representation. Thus, the development of efficient and adaptive multi-scale algorithms and the

Table 14.2
Available simulation tools (in alphabetical order)

Name	Brief Description
BIONETGEN	Automatic generation of physicochemical models of biological systems with combinatorial complexity such as cellular signaling from user-specified rules for biomolecular interactions at the protein domain level, integrated with tools for reaction network simulation and analysis (53). http://cellsignaling.lanl.gov/bionetgen/
BIO-SPICE	Open source software for intra- and inter-cellular modeling and simulation of spatio-temporal processes, pathways, and interaction- network tools for data analysis (MIAMESpice), model composition and visualization (BioSenS, Charon, JDesigner, Sal), mathematical tools including ordinary and partial differential and stochastic equation-based methods and algorithms for the simulation of reaction and diffusion-reaction networks (BioNetS Simpathica) (54, 55). https://biospice.org/index.php
CELLDESIGNER	Gene regulatory network modeling, visual representation of biochemical semantics with SBML and database support, integration with Matlab or Mathematica ODE Solvers, and direct access to SBW SBML compliant simulators such as Jarnac (56, 57). http://celldesigner.org/index.html
CELLERATOR	Open- source Mathematica- based package for the simulation and analysis of signal transduction networks in cells and multicellular tissues, automated equation generation with arrow-based reaction notation (58). http://www-aig.jpl.nasa.gov/public/mls/cellerator/
COPASI	Stochastic and deterministic simulation of network pathways, steady- state and metabolic control analysis including stability analysis, elementary and mass conservation analysis, optimization and parameter estimation and SBML support (59). http://www.copasi.org/tiki-index.php
DIZZY	Stochastic and deterministic kinetic modeling of integrated large-scale genetic, metabolic, and signaling networks with domain compartmentalization, features include modular simulation framework, reusable modeling elements, complex kinetic rate laws, multi-step reaction processes, steady-state noise estimation (60). http://magnet.systemsbiology.net/software/Dizzy
E-CELL	Object-oriented software for modeling, simulation, and analysis of large scale cellular networks, multi-algorithm multi-time- scale simulation method with access to Gillespie-Gibson and Langevin stochastic algorithms, Euler and higher- order adaptive methods for ordinary and algebraic differential equations, parallel and distributed computing capabilities and SBML support (61, 62). http://www.e-cell.org/software/e-cell-system
GRID CELLWARE	Grid-based modeling and simulation tool for biochemical pathways, integrated environment for diverse mathematical representations, parameter estimation using swarm algorithm and optimization, user-friendly graphical display and

(continued)

Table 14.2 (continued)

Name	Brief Description
	<p>capability for large, complex models, stochastic algorithms such as Gillespie, Gibson, and tau-leaping and deterministic algorithms based on ordinary differential equation solvers (63).</p> <p>http://www.bii.a-star.edu.sg/research/sbg/cellware</p>
MCELL	<p>Monte Carlo simulator for investigating cellular physiology, model description language or MDL used to specify ligand creation, destruction, and release as well as chemical reactions involving effectors such as receptors or enzymes, ligand diffusion modeled with 3D random walks, wide variety of numerical and imaging options (64).</p> <p>http://www.mcell.cnl.salk.edu and www.mcell.psc.edu</p>
MESORD	<p>Open source C++ software for stochastic simulation of kinetic reactions and diffusion based on next subvolume method (NSM) for structured geometries in three dimensions, SBML-compatible, three-dimensional OpenGL simulation viewer (65).</p> <p>http://www.mcell.cnl.salk.edu and http://mesord.sourceforge.net/</p>
NWKSIM	<p>Fortran 90/95 platform for large-scale general kinetic reaction network simulation, Java swing graphical user interface, unique modeling features for the computation of evolving multiple compartments with trafficking and signal transduction processes, stochastic algorithms include direct Gillespie, probability weighted dynamic Monte Carlo and weighted Gibson-Gillespie Bruck, binomial and multinomial tau-leaping algorithms (32, 42).</p>
SIGTRAN	<p>Fortran 90/95 modeling and simulation platform for large-scale reaction networks, Java swing graphical user interface, dual stochastic and deterministic (ODE/DAE) simulation modes, multi-state macromolecule specification and simulation using the Firth-Bray algorithm, reaction-diffusion modeling with Fricke-Wendt algorithm, molecule tagging and tracking in signal transduction networks using a fully automated graph-theoretic algorithms for the determination of unambiguous set of base species, SBML file support (42).</p>
SIMBIOLOGY	<p>Matlab toolkit for modeling, simulating, and analyzing biochemical pathways, graphical user interface with visual pathway expression, manual or SBML file input, stochastic or deterministic solvers, parameter estimation and sensitivity analysis, ensemble runs and post-run analysis tools with plotting.</p> <p>http://www.mathworks.com/products/simbiology/description4.html</p>
SMARTCELL	<p>Object-oriented C++ platform for the modeling and simulation of diffusion-reaction networks in whole-cell context with support for any cell geometry with different cell compartments and species localization, includes DNA transcription and translation, membrane diffusion, and multistep reactions, as well as cell growth, localization, and diffusion modeling based on mesoscopic stochastic reaction algorithm (66).</p> <p>http://smartcell.embl.de/introduction.html</p>
STOCHKIT	<p>Efficient C++ stochastic simulation framework for intracellular biochemical processes, stochastic algorithms includes Gillespie SSA and explicit, implicit and trapezoidal tau-leaping methods, Kolmogorov distance and histogram</p>

(continued)

Table 14.2 (continued)

Name	Brief Description
	distance for quantifying difference quantification in statistical distribution shapes via Matlab, extensible to new stochastic and multi-scale algorithms (67). http://www.engineering.ucsb.edu/~cse/StochKit/StochKit.html
SBTOOLBOX	Matlab toolbox offering open and user extensible environment for prototyping new algorithms, and building applications for the analysis and simulation of biological systems, deterministic and stochastic simulation, network identification, parameter estimation and sensitivity analysis, bifurcation analysis, SBML model import (68). http://www.sbtoolbox.org/
VIRTUAL CELL	Model creation and simulation of cell biological processes, associates biochemical and electrophysiological data for individual reactions with experimental microscopic image data describing subcellular locations, cell physiological events simulated within empirically derived geometries, reusable, updatable, and accessible models, simulation data stored on the Virtual Cell database server, and is easily exportable in a variety of formats (69, 70). http://www.vcell.org/

software for stochastic-deterministic simulations are urgently needed. This is particularly true for the simulation of spatially resolved models because immense recent progress in imaging is beginning to make the data about the inhomogeneous distribution of molecular species available. As the expense associated with spatially resolved models makes fully stochastic simulations nearly infeasible, efforts to develop hybrid approaches will become very important. Hybrid methods (22, 71) have recently been proposed to simulate multi-scale chemically reacting systems. These methods combine the traditional deterministic reaction rate equation, or alternatively the chemical Langevin equation, with the SSA. The idea is to split the system into two regimes: the continuous regime and the discrete regime. The RRE is used to simulate the fast reactions between species with large populations and the SSA is used for slow reactions or species with small populations. The conditions for the continuous regime are (1) the number of instances of each molecular species in a reaction in the continuous regime must be large relative to one, and (2) the number of reaction events of each reaction occurring within one time step of the numerical solver must be large relative to one (72). If either condition is not satisfied for a reaction channel, that reaction channel must be handled in the discrete regime.

The hybrid methods efficiently use the multi-scale properties of the problem, but there are still some fundamental unsolved problems. The first is the automatic partitioning of systems. This

needs to be done very carefully, because both accuracy and efficiency depend on a proper partitioning. It will require a precise awareness of the assumptions under which algorithms and approximation available in the multi-scale framework are valid, and the corresponding tests on the system to determine the partitioning. This is one reason why a proper theoretical foundation for each algorithm is so critical – to ensure that we are using it only to make approximations for which the corresponding assumptions are valid. Concerns about automatic step size selection continue to exist. Another difficult problem arises in the case when a reaction is fast, but one of the corresponding reactants has a small population, thus failing to satisfy the second requirement above for the deterministic regime. Such a reaction channel will still be handled by the SSA, resulting in a very slow simulation. New approaches and approximations, such as the slow-scale SSA (73), have the potential to overcome this limitation.

Another promising area of research is the application of tau-leaping-based stochastic algorithms to problems involving reaction-diffusion processes. When the SSA algorithm is applied to reaction-diffusion processes, the standard approach is to subdivide the heterogeneous system volume into a sufficiently large number of small-volume elements $V_0 \sim O(h^3)$ so that the assumption of a homogeneous, well-stirred system within each element is recovered. With diffusion considered as a first-order reaction in which molecules of a species are exchanged between neighboring elements at a rate $r_D \propto V_0 D/h^2$, kinetic reactions within an element occur at a rate $r_C \propto V_0 \Sigma$. Here D and Σ are, respectively, the diffusion constant and the total system propensity. By choosing elements sufficiently small so that $r_C \ll r_D$ is true for all elements, the SSA may be applied, although under these circumstances it is not usual to be confronted with a reaction network where the total number of processes and species is now several orders of magnitude larger than that found in a single element. Moreover, it has been observed in stochastic simulations with the SSA that reaction-diffusion processes are strongly dominated by diffusion events. The increase in network size and diffusion-dominated events both contribute to a severe reduction in efficiency and it is highly likely that a number of more recent stochastic algorithms for reaction-diffusion suffer from the same problem. It is possible that τ -leaping methods may be well suited to overcoming this issue with the stochastic simulation of reaction-diffusion problems.

As stochastic simulation is used more and more often in studying the dynamics of biological systems, and as biological models become larger and more complicated, there is a need for the development and efficient implementation of simulation algorithms that can handle the increasing complexity. Thus, the issue of how an algorithm scales with the system size is crucial in assessing the limitations of the algorithm. Scaling concerns expose the

serious need for establishing realistic scalable models that can be used in benchmark studies for algorithm efficiency comparison. Pettigrew and Resat (42) have introduced two such models. It was observed (42) that the scaling qualities of the algorithms investigated break down significantly when the network reaches a certain size. We suspect that this is a common occurrence for all of the existing algorithms, which unfortunately can impose severe limitations on the utility of stochastic kinetic approaches. Roughly estimating, existing stochastic simulation algorithms can be used to investigate reaction networks containing $\sim 10^5$ reactions using a desktop machine. Utilization of high-performance resources may push this limit to 10^8 . Considering that even the smallest microbial community or physiologically relevant eukaryotic network can easily contain $> 10^{12}$ reactions, it is clear that the scaling properties of the stochastic simulation algorithms need significant improvements before they can be widely used for larger problems. As discussed above, hybrid methods that reduce the computational complexity by treating the non-critical parts of the network using deterministic models may offer the most suitable solution to the scaling bottleneck in the near future.

We conclude by observing that while recent improvements in algorithm development, computational implementation, and hybrid approaches are indications of a promising future for the stochastic kinetic simulation of biological systems, there is still plenty of room for achieving significant advances.

References

1. Gillespie DT (1976) A general method for numerically simulating stochastic time evolution of coupled chemical-reactions. *Journal of Computational Physics* **22**, 403–434.
2. Gillespie DT (1977) Exact stochastic simulation of coupled chemical-reactions. *Journal of Physical Chemistry* **81**, 2340–2361.
3. Gillespie DT (1977) Concerning validity of stochastic approach to chemical-kinetics. *Journal of Statistical Physics* **16**, 311–318.
4. Gillespie DT (1992) A rigorous derivation of the chemical master equation. *Physica A* **188**, 404–425.
5. Arkin A, Ross J, and McAdams HH (1998) Stochastic kinetic analysis of developmental pathway bifurcation in phage lambda-infected *Escherichia coli* cells. *Genetics* **149**, 1633–1648.
6. McAdams HH and Arkin A (1997) Stochastic mechanisms in gene expression. *Proceedings of the National Academy of Sciences, USA* **94**, 814–819.
7. McAdams HH and Arkin A (1999) It's a noisy business! Genetic regulation at the nanomolar scale. *Trends in Genetics* **15**, 65–69.
8. Bortz AB, Kalos MH, and Lebowitz JL (1975) New algorithm for Monte-Carlo simulation of Ising spin systems. *Journal of Computational Physics* **17**, 10–18.
9. Endy D and Brent R (2001) Modelling cellular behaviour. *Nature* **409**, 391–395.
10. Gibson MA and Bruck J (2000) Efficient exact stochastic simulation of chemical systems with many species and many channels. *Journal of Physical Chemistry A* **104**, 1876–1889.
11. Goss PJE and Peccoud J (1998) Quantitative modeling of stochastic systems in molecular biology by using stochastic Petri nets. *Proceedings of the National Academy of Sciences, USA* **95**, 6750–6755.

12. Kastner J, Solomon J, and Fraser S (2002) Modeling a Hox gene network in silico using a stochastic simulation algorithm. *Developmental Biology* **246**, 122–131.
13. Kepler TB and Elston TC (2001) Stochasticity in transcriptional regulation: Origins, consequences, and mathematical representations. *Biophysical Journal* **81**, 3116–3136.
14. Rao CV and Arkin AP (2003) Stochastic chemical kinetics and the quasi-steady-state assumption: Application to the Gillespie algorithm. *Journal of Chemical Physics* **118**, 4999–5010.
15. Simpson ML, Cox CD, and Sayler GS (2003) Frequency domain analysis of noise in autoregulated gene circuits. *Proceedings of the National Academy of Sciences, USA* **100**, 4551–4556.
16. Smolen P, Baxter DA, and Byrne JH (1999) Effects of macromolecular transport and stochastic fluctuations on dynamics of genetic regulatory systems. *American Journal of Physiology-Cell Physiology* **277**, C777–C790.
17. Levchenko A (2003) Dynamical and integrative cell signaling: challenges for the new biology. *Biotechnol Bioeng* **84**, 773–782.
18. Haseltine EL and Rawlings JB (2002) Approximate simulation of coupled fast and slow reactions for stochastic chemical kinetics. *Journal of Chemical Physics* **117**, 6959–6969.
19. Haseltine EL and Rawlings JB (2005) On the origins of approximations for stochastic chemical kinetics. *Journal of Chemical Physics* **123**, 164115.
20. Puchalka J and Kierzek AM (2004) Bridging the gap between stochastic and deterministic regimes in the kinetic simulations of the biochemical reaction networks. *Biophysical Journal* **86**, 1357–1372.
21. Fricke T and Wendt D (1995) The Markov Automaton – A new algorithm for simulating the time-evolution of large stochastic dynamic-systems. *International Journal of Modern Physics C-Physics and Computers* **6**, 277–306.
22. Elf J, Doncic A, and Ehrenberg M (2003) Mesoscopic reaction-diffusion in intracellular signaling. In: *SPIE's First International Symposium on Fluctuations and Noise*, pp. 114–124.
23. Elf J and Ehrenberg M (2003) Fast evaluation of fluctuations in biochemical networks with the linear noise approximation. *Genome Research* **13**, 2475–2484.
24. Resat H, Wiley HS, and Dixon DA (2001) Probability-weighted dynamic Monte Carlo method for reaction kinetics simulations. *Journal of Physical Chemistry B* **105**, 11026–11034.
25. Resat H, Ewald JA, Dixon DA, and Wiley HS (2003) An integrated model of epidermal growth factor receptor trafficking and signal transduction. *Biophysical Journal* **85**, 730–743.
26. Gillespie DT and Petzold LR (2003) Improved leap-size selection for accelerated stochastic simulation. *Journal of Chemical Physics* **119**, 8229–8234.
27. Rathinam M, Petzold LR, Cao Y, and Gillespie DT (2003) Stiffness in stochastic chemically reacting systems: The implicit tau-leaping method. *Journal of Chemical Physics* **119**, 12784–12794.
28. Cao Y, Gillespie DT, and Petzold LR (2006) Efficient step size selection for the tau-leaping simulation method. *Journal of Chemical Physics* **124**, 044109.
29. Chatterjee A, Mayawala K, Edwards JS, and Vlachos DG (2005) Time accelerated Monte Carlo simulations of biological networks using the binomial tau-leap method. *Bioinformatics* **21**, 2136–2137.
30. Chatterjee A, Vlachos DG, and Katsoulakis MA (2005) Binomial distribution based tau-leap accelerated stochastic simulation. *Journal of Chemical Physics* **122**, 024112.
31. Tian TH and Burrage K (2004) Binomial leap methods for simulating stochastic chemical kinetics. *Journal of Chemical Physics* **121**, 10356–10364.
32. Pettigrew MF and Resat H (2007) A multinomial tau-leaping method for stochastic kinetic simulations. *Journal of Chemical Physics* **126**, 084101.
33. Stundzia AB and Lumsden CJ (1996) Stochastic simulation of coupled reaction-diffusion processes. *Journal of Computational Physics* **127**, 196–207.
34. Burke P, Schooler K, and Wiley HS (2001) Regulation of epidermal growth factor receptor signaling by endocytosis and intracellular trafficking. *Molecular Biology of the Cell* **12**, 1897–1910.
35. Ozcelik S, Orr G, Hu D, Chii-Shiang C, Resat H, Harms GS, Opresko LK, Wiley HS, and Colson SD (2004) FRET measurements between small numbers of molecules identifies subtle changes in receptor interactions. *Proceedings of the International Society of Optical Engineering* **5323**, 119–127.

36. Viollier PH, Thanbichler M, McGrath PT, West L, Meewan M, McAdams HH, and Shapiro L (2004) Rapid and sequential movement of individual chromosomal loci to specific subcellular locations during bacterial DNA replication. *Proceedings of the National Academy of Sciences, USA* **101**, 9257–9262.
37. McAdams HH and Shapiro L (2003) A bacterial cell-cycle regulatory network operating in time and space. *Science* **301**, 1874–1877.
38. Judd EM, Ryan KR, Moerner WE, Shapiro L, and McAdams HH (2003) Fluorescence bleaching reveals asymmetric compartment formation prior to cell division in *Caulobacter*. *Proceedings of the National Academy of Sciences, USA* **100**, 8235–8240.
39. Gardiner CW, McNeil KJ, Walls DF, and Matheson IS (1976) Correlations in stochastic theories of chemical reactions. *Journal of Statistical Physics* **14**, 307–331.
40. Chaturvedi S, Gardiner CW, Matheson IS, and Walls DF (1977) Stochastic analysis of a chemical reaction with spatial and temporal structures. *Journal of Statistical Physics* **17**, 469–489.
41. Gillespie DT (1992) *Markov Processes: An Introduction for Physical Scientists*. Academic Press, San Diego, California.
42. Pettigrew MF and Resat H (2005) Modeling signal transduction networks: a comparison of two stochastic kinetic simulation algorithms. *Journal of Chemical Physics* **123**, 114707.
43. Ascher UM and Petzold LR (1998) *Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations*. SIAM, Philadelphia.
44. McQuarrie DA (1967) Stochastic approach to chemical kinetics. *Journal of Applied Probability* **4**, 413–478.
45. Gibson MA and Bruck J (1998) An efficient algorithm for generating trajectories of stochastic gene regulation reactions. *California Institute of Technology Report ETR026*.
46. Cao Y, Li H, and Petzold LR (2004) Efficient formulation of the stochastic simulation algorithm for chemically reacting systems. *Journal of Chemical Physics* **121**, 4059–4067.
47. Gillespie DT (2001) Approximate accelerated stochastic simulation of chemically reacting systems. *Journal of Chemical Physics* **115**, 1716–1733.
48. Cao Y, Gillespie DT, and Petzold LR (2005) Avoiding negative populations in explicit Poisson tau-leaping. *Journal of Chemical Physics* **123**, 054104.
49. Gillespie DT (2000) The chemical Langevin equation. *Journal of Chemical Physics* **113**, 297–306.
50. Gillespie DT (2002) The chemical Langevin and Fokker-Planck equations for the reversible isomerization reaction. *Journal of Physical Chemistry A* **106**, 5063–5071.
51. Gillespie DT (1996) The multivariate Langevin and Fokker-Planck equations. *American Journal of Physics* **64**, 1246–1257.
52. Morton-Firth CJ (1998) Ph.D. thesis. Stochastic simulation of cell signalling pathways. University of Cambridge, Cambridge, UK.
53. Blinov ML, Faeder JR, Goldstein B, and Hlavacek WS (2004) BioNetGen: software for rule-based modeling of signal transduction based on the interactions of molecular domains. *Bioinformatics* **20**, 3289–3291.
54. Sauro HM, Hucka M, Finney A, Wellock C, Bolouri H, Doyle J, and Kitano H (2003) Next generation simulation tools: the Systems Biology Workbench and BioSPICE integration. *Omic* **7**, 355–372.
55. Adalsteinsson D, McMillen D, and Elston TC (2004) Biochemical Network Stochastic Simulator (BioNetS): Software for stochastic modeling of biochemical networks. *BMC Bioinformatics* **5**, 24.
56. Funahashi A, Tanimura N, Morohashi M, and Kitano H (2003) CellDesigner: A process diagram editor for gene-regulatory and biochemical networks. *Bio Silico* **1**, 159–162.
57. Kitano H (2003) A graphical notation for biochemical networks. *Bio Silico* **1**, 169–176.
58. Shapiro BE, Levchenko A, Meyerowitz EM, Wold BJ, and Mjolsness ED (2003) Cellerator: Extending a computer algebra system to include biochemical arrows for signal transduction simulations. *Bioinformatics* **19**, 677–678.
59. Hoops S, Sahle S, Gauges R, Lee C, Pahle J, Simus N, Singhal M, Xu L, Mendes P, and Kummer U (2006) COPASI – A COMplex PATHway SIMulator. *Bioinformatics* **22**, 3067–3074.
60. Ramsey S, Orrell D, and Bolouri H (2005) Dizzy: Stochastic simulation of large-scale genetic regulatory networks. *Journal of Bioinformatics and Computational Biology* **3**, 415–436.

61. Takahashi K, Ishikawa N, Sadamoto Y, Sasamoto H, Ohta S, Shiozawa A, Miyoshi F, Naito Y, Nakayama Y, and Tomita M (2003) E-Cell 2: Multi-platform E-Cell simulation system. *Bioinformatics* **19**, 1727–1729.
62. Takahashi K, Kaizu K, Hu B, and Tomita M (2004) A multi-algorithm, multi-timescale method for cell simulation. *Bioinformatics* **20**, 538–546.
63. Dhar PK, Meng TC, Somani S, Ye L, Sakharkar K, Krishnan A, Ridwan AB, Wah SH, Chitre M, and Hao Z (2005) Grid cellware: The first grid-enabled tool for modelling and simulating cellular processes. *Bioinformatics* **21**, 1284–1287.
64. Stiles JR and Bartol TM (2001) Monte Carlo methods for simulating realistic synaptic microphysiology using MCell. In: *Computational Neuroscience: Realistic Modeling for Experimentalists*. De Schutter E, ed., CRC Press, Boca Raton, pp. 87–127.
65. Hattne J, Fange D, and Elf J (2005) Stochastic reaction-diffusion simulation with MesoRD. *Bioinformatics* **21**, 2923–2924.
66. Ander M, Beltrao P, Di Ventura B, Ferkinghoff-Borg J, Foglierini M, Kaplan A, Lemerle C, Tomas-Oliveira I, and Serrano L (2004) SmartCell, a framework to simulate cellular processes that combines stochastic approximation with diffusion and localisation: Analysis of simple networks. *System Biology (Stevenage)* **1**, 129–138.
67. Cao Y and Petzold LR (2005) *Trapezoidal tau-leaping formula for the stochastic simulation of bio-chemical systems*. In: *Proceedings of Foundations of Systems Biology in Engineering (FOSBE 2005)*, pp. 149–152.
68. Schmidt H and Jirstrand M (2006) Systems biology toolbox for MATLAB: A computational platform for research in systems biology. *Bioinformatics* **22**, 514–515.
69. Slepchenko BM, Schaff JC, Macara I, and Loew LM (2003) Quantitative cell biology with the virtual cell. *Trends Cell Biology* **13**, 570–576.
70. Moraru, II, Schaff JC, Slepchenko BM, and Loew LM (2002) The virtual cell: An integrated modeling environment for experimental and computational cell biology. *Annals of the New York Academy of Sciences* **971**, 595–596.
71. Isaacson SA and Peskin CS (2004) Incorporating diffusion in complex geometries into stochastic chemical kinetics simulations. Courant Institute of Mathematical Sciences Report.
72. Mattheyses T and Simmons M (2004) Hybrid simulation of cellular behavior. *Bioinformatics* **20**, 316–322.
73. Cao Y, Gillespie DT, and Petzold L (2005) The slow-scale stochastic simulation algorithm. *Journal of Chemical Physics* **122**, 014116.

Chapter 15

Guidance for Data Collection and Computational Modelling of Regulatory Networks

Adam Christopher Palmer, and Keith Edward Shearwin

Abstract

Many model regulatory networks are approaching the depth of characterisation of bacteriophage λ , wherein the vast majority of individual components and interactions are identified, and research can focus on understanding whole network function and the role of interactions within that broader context. In recent years, the study of the system-wide behaviour of phage λ 's genetic regulatory network has been greatly assisted by the combination of quantitative measurements with theoretical and computational analyses. Such research has demonstrated the value of a number of general principles and guidelines for making use of the interplay between experiments and modelling. In this chapter we discuss these guidelines and provide illustration through reference to case studies from phage λ biology.

In our experience, computational modelling is best facilitated with a large and diverse set of quantitative, *in vivo* data, preferably obtained from standardised measurements and expressed as absolute units rather than relative units. Isolation of subsets of regulatory networks may render a system amenable to 'bottom-up' modelling, providing a valuable tool to the experimental molecular biologist. Decoupling key components and rendering their concentration or activity an independent experimental variable provide excellent information for model building, though conclusions drawn from isolated and/or decoupled systems should be checked against studies in the full physiological context; discrepancies are informative. The construction of a model makes possible *in silico* experiments, which are valuable tools for both the data analysis and the design of wet experiments.

Key words: Computational modelling, systems biology, gene regulatory network, experiment design, promoter regulation, *in silico* experiment, bacteriophage λ , DNA looping.

1. Introduction

In our studies of small gene regulatory networks, our model organisms are two temperate bacteriophages, lambda (λ) and the unrelated P2-like phage 186. Both phage make a decision between lysis

and lysogeny upon infection of their host *Escherichia coli*, and contain within their genetic circuitry a module that operates as a bistable genetic switch when isolated and inserted into *E. coli*. Phage λ 's bistable switch is a paradigm for the molecular basis of epigenetics, and the lysis-lysogeny decision is the most thoroughly characterised model system of developmental decision-making. Both phage λ and 186 are sufficiently well characterised that most key components of the lysis-lysogeny decisions have been identified, allowing research to extend to both the smaller and larger scales, respectively, to detailed characterisation of the molecular mechanisms and to the role of an interaction within the context of a larger network.

Phage allows us to study the organism on all scales, from atomic resolution of proteins to behaviour of the whole organism. We observe the basis of information processing and decision-making at the level of protein-protein and protein-nucleic acid interactions, which often converge at transcription regulation. However, the behaviour of a subset of a genetic regulatory network, even as small as two promoters and two regulatory genes, can display the complexity of behaviour beyond the ability of intuition or qualitative description to fully appreciate. Therefore, our studies of phage frequently require us to apply quantitative experimental methods and mathematical and/or computational modelling in the interpretation of data.

A minority of biochemical and genetic experiments provide data with the qualities necessary for mathematical modelling; a change in the character of experimental data is being advocated by those who see molecular biology progressing to a stage where quantitative and systems-level understanding will be a central component of future progress (1, 2). This lab's 35 years of experience in phage λ and 186 can testify that as the identification of key components in a system draws to completion, quantitative studies and theoretical analyses are necessary to assemble the data provided by reductionist experiments into a coherent picture of the whole network and to guide future experiments into network function.

2. Data Collection for Modelling

Production of experimental data suitable for mathematical modelling is challenging; for a majority of applications, a large set of highly quantitative data is a must. Furthermore, although data are usually obtained as relative units, expression in absolute units is often valuable, potentially requiring an entirely different experiment for calibration, which again requires quantitative techniques.

We find the following guidelines useful in the production of experimental data suitable for computational modelling.

2.1. Use Quantitative Techniques

Models based upon purely qualitative descriptions of interacting systems will possibly have different solutions capable of describing the same data, and are likely to suffer from a lack of detail and accordingly a lack of predictive power. With quantitative data, the process of fitting a model to the data provides estimates of relevant parameters, sometimes providing information that may not be at all accessible to direct experimental measurement. Production of reliably quantitative data requires replicate experiments, especially when working *in vivo*.

2.2. Acquire a Large and Diverse Data Set

For a mathematical model to be held in confidence, it must have fewer parameters than the number of independent data points; a large and diverse data set has the greatest chance of delivering an accurate model. Data particularly amenable to modelling include measurements of concentrations or reaction rates, as a function of either time or concentration of a regulatory molecule. Such concentration or time series appear frequently in biological literature, but usually with sparse data, adequate to prove a qualitative point. The inclusion of more frequent or more densely spaced measurements in these studies can easily make data more amenable to computational modelling, from which much more may potentially be learnt about the system.

The ‘diversity’ of a data set is a trait distinct from size, and is also valuable. While it is useful to measure, for example, a time series with many data points, there is a different advantage in measuring that time series under different circumstances, such as with certain components of the system altered or removed. Such changes, particularly mutations, are routine in most biochemical experiments, and they are no less useful when acquiring data for the purposes of computational modelling, as the acquisition of a new data set with one or two parameters altered can be extremely useful for fitting a model to the data.

2.3. Keep Conditions as Close to Physiological as Possible

In attempting to model a process inside a cell, it is of course important that any quantitative measurements needed for the model be performed under conditions as close to physiological as possible. *In vivo* experiments are ideal, but for many measurements only *in vitro* techniques are available; in these cases, *in vitro* experiments performed under conditions most similar to *in vivo* are most useful. This can be accomplished by the use of physiologically realistic salt concentrations and pH, with the inclusion of macromolecular crowding agents, and by working at the same temperature as any *in vivo* experiments.

2.4. Measure Absolute Units or Use Standardised Measurements

Even using quantitative techniques, measurements often only provide relative units, e.g. a *fold* change is provided in protein concentration or promoter activity. Quantitative techniques may provide only enough data to draw qualitative conclusions. For the construction of a computational model, it is much more desirable to obtain absolute units, e.g. number of proteins per cell, or promoter initiations per minute. Calibration against a known standard will provide this information, which can be immensely valuable in modelling. Beyond assisting your model, providing absolute units provides a reliable way to make quantitative comparisons of data obtained from different laboratories. Such calibrations themselves require quantitative techniques and may require substantial effort; if calibrations are not possible or simply low priority, the use of standardised measurements may at least assist the comparison of data from different laboratories, the difficulty of which has been lamented by the researchers interested in modelling (1).

3. Case Study: Measurement of Prokaryotic Promoters

Current techniques for the measurement of prokaryotic promoters provide a good example of the efforts that can be taken to acquire the highest quality of data for computational modelling. Promoter regulation is amenable to quantitative measurement by the placement of a ‘reporter’ gene downstream of a promoter, whose product is easily measurable, and demonstrates minimal interference with cellular behaviour. In *E. coli* the leading example is the *lacZ* gene, whose product β -galactosidase is measured in the Miller LacZ assay by the rate of enzymatic cleavage of a chromogenic substrate (3). This assay has been refined to automated kinetic *lacZ* assays, which provide a highly quantitative technique for the measurement of promoter activity, with sensitivity spanning 4 orders of magnitude (4). Chromosomal single-copy reporters are used to avoid noise due to plasmid copy number variability, by the use of reporters integrated into the *E. coli* chromosome at a specific phage attachment site, using either phage themselves or a system of plasmids that exploits the integration machinery of temperate phage (5, 6). Transcriptional terminators isolate the region of interest from read-through effects from the surrounding *E. coli* chromosome (Fig. 15.1). When combining a LacZ reporter with a lactose/IPTG-inducible expression system, *lacY* (permease) should be deleted to avoid feedback from transcription (output) to transport of the inducer (input) (7). It is worth noting that chromosomal single-copy reporters, long used in *E. coli*, have recently been implemented in mammalian cell lines (8).

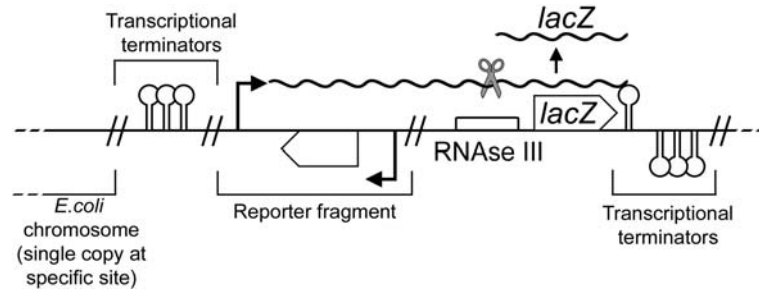


Fig. 15.1. Schematic diagram of chromosomal single-copy *lacZ* reporter. Double slashes indicate junctions between DNA of different origin.

Transcriptional fusions to *lacZ* are used to enable the placement of an RNaseIII site preceding the *lacZ* gene, such that all *lacZ* transcripts are cleaved between their start sites and the LacZ coding sequence (**Fig. 15.1**). This yields a standardised measurement, as *lacZ* transcripts of identical length and sequence are produced from any promoter, providing mRNA translation efficiencies and half-lives, which are promoter and context independent (9). By using a standardised assay, promoter activities measured by this assay are directly comparable to any prokaryotic promoter assayed, provided the host strain and its growth conditions are kept constant. Translation regulation can be studied by LacZ assays through the use of translational fusions to *lacZ* (5). Translational fusions to *lacZ* also can provide information on the efficiency of translation from a given message and ribosome-binding site, subject to the limitation that fusion of a protein to LacZ may alter its activity and fusion of a transcript to the *lacZ* coding sequence may alter mRNA stability.

For the purposes of computational modelling, LacZ units can be converted into the absolute units of RNA polymerase initiations/minute, by the work of (10), whose exhaustive study of constitutive promoters in *E. coli* characterised the RNA polymerase initiation rate from a range of promoters, as a function of growth rate. Any given quantitative assay of promoter activity can be calibrated with the measurement of one or more of the promoters studied by (10).

In cases where an inducible promoter provides a variable supply of protein, quantitative western blotting allows for the conversion of inducer concentration to protein concentration (an absolute unit), given knowledge of the cells' internal volume and a protein standard of known concentration. A protein standard can be obtained from purified protein or from some fixed internal supply of protein, such as is common in measurements of the λ lysogenic repressor CI, where the steady concentration present in a lysogen (Wild-type Lysogenic Unit = W.L.U.) is taken as a point of reference.

4. Experiment Design for 'Bottom- Up' Modelling

Molecular biology has made enormous strides with the reductionist approach: 'The idea is that you could understand the world, all of nature, by examining smaller and smaller pieces of it. When assembled, the small pieces would explain the whole' (11). 'Bottom-Up' modelling is the technique of assembling these small pieces of information into the whole, and accordingly for the typical molecular biology laboratory, this is the most useful way to incorporate theoretical methods into an experimental program. In contrast, 'top-down' modelling involves models based on global data sets such as whole genome or proteome expression profiles, for which theoretical analyses are an absolute necessity. A general investigative procedure that may be of assistance to experimentalists considering the incorporation of bottom-up modelling follows (Fig. 15.2).

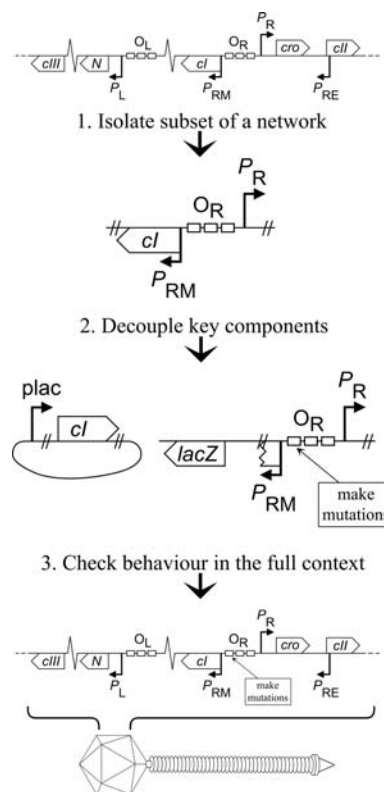


Fig. 15.2. Generalised experimental procedure facilitating bottom-up modelling. Illustrations are derived from the case study of Section 5.1.

4.1. Isolate Subsets of a Network

While not always possible, if a small subset of a network can be identified and studied in isolation from its full natural context, detailed characterisation of components and their interactions is greatly assisted. Though this does not provide all physiologically relevant information about the isolated components, such as their responses to now absent inputs, the more thorough characterisation that can be achieved with a smaller and less complex system provides an excellent foundation for subsequent expansion to a larger system. A subset of a network need not necessarily be a ‘module’ in the sense of an independently functioning unit: there may be value even in dissecting the system beyond the point of functionality.

Studies of temperate bacteriophage have the rare privilege of being able to isolate components, e.g. the bistable switch, and express them within their usual cytoplasmic environment, e.g., *E. coli*, in the absence of any other transcribed phage genes. For many areas of research, the techniques for isolation of a system while remaining *in vivo* do not yet exist, making *in vitro* reconstitution the best procedure available. It is reasonable to expect though that in time the technologies will be developed which will allow for subsets of biochemical networks to be isolated and studied under *in vivo* conditions; the history of phage λ research demonstrates the value of these technologies.

4.2. Decouple Key Components

Where a complex system is able to settle into one or more stable states, probing the conditions of these states provides much less information than can be obtained with a method of probing the full continuum of states, which exist when the system displays dynamic behaviour or switches between stable points. An experimental approach to achieve this is to decouple key regulatory components, i.e., remove them from the context of their usual control mechanisms, and make their concentration or activity an independent experimental variable.

With regulatory proteins or stimuli controlled experimentally, they can be varied independently over the physiologically relevant scope, potentially spanning the entire range from ‘knock-out’ to overexpression. Detailed measurements over this range will be much more informative than the measurement of stable states or extremes only. In particular, this approach is an ideal way to succeed in the advice mentioned in **Section 2.2**.

Finally, feedback is a common theme in dynamical systems of sufficient complexity to justify a modelling approach; gene regulatory networks and signal transduction pathways have shown themselves to be prime examples. Even small systems may exhibit complex behaviour specifically due to feedback, which can complicate the interpretation of experimental results. Removal of feedback mechanisms by decoupling of a feedback-regulated

component may both simplify data analysis and expose the purpose of feedback when compared to the non-decoupled behaviour.

4.3. Check Behaviour in the Full Context

It is *in vivo* behaviour that we ultimately aim to understand, and all interactions observed in a smaller subsystem exist to serve a greater purpose within an entire cell or organism. Therefore, it is of critical importance that any conclusions drawn from studies of isolated and/or decoupled systems be checked against the behaviour observed in the full physiological context. The return to wild-type can be taken in steps, beginning with the restoration of regulatory links that may have been removed in the process of decoupling (as described above in investigating feedback), prior to returning to the study of the subsystem within its full physiological context.

We find it highly productive to characterise the effects of mutations upon both the isolated subset and the whole organism. If the effect of a mutation on the isolated subset does not explain the mutation's phenotype in the physiological context, a clue is provided to the discovery of new components or interactions that were excluded from the previously chosen subset. The physiological role of long-range DNA looping in phage λ was found by just such an observation; this example is detailed in **Section 5.1**.

A noteworthy alternative is the 'module-replacement' approach developed by Little and Atsumi, wherein a regulatory factor and the *cis*-acting sites to which it binds (collectively a module) are replaced by an equivalent exogenous factor and its cognate binding sites (12, 13). The module-replaced system can in principle be characterised both as an isolated subset and in the whole organism. Firstly, this technique will check assumptions about the properties and functions of the regulatory module, by assessing whether the replacement module indeed provides functionality; failure may indicate that more has been removed than is appreciated. Secondly, the inserted module can be engineered to exclude one or more features, such as a specific binding site, much more easily than its wild-type counterpart; all that is necessary is to *not add* a feature, rather than removing a pre-existing feature.

4.4. Model Building

When a molecular system is characterised to the extent that most components have been identified and hypotheses have been developed about interactions between components, bottom-up models can be built by assembling the known or suspected interactions into a deterministic or stochastic model. Competing models can be built containing different sets of interactions or molecular mechanisms, and their ability to explain the quantitative experimental data will assist in the discrimination between models.

4.4.1. Modelling Techniques

The choice of modelling technique is of significance to both the accuracy and the educational value of the model. For relatively simple systems, an analytic model can be constructed, that is, a simple set of descriptive equations, while increasingly complex systems will require such approaches as the use of the partition function from statistical thermodynamics, or modelling with ordinary differential equations (ODEs). In our experience, time-series data is best described by ODEs, and the partition function is most useful for explaining data expressed as a function of the concentration of some component. All these techniques are deterministic: they do not account for statistical fluctuations in concentrations and rates, or ‘molecular noise’, which is increasingly appreciated as an important factor in the behaviour of biochemical systems (14). Therefore, even were modellers provided with perfect and complete information about a system, a stochastic simulation would be the only way to produce accurate data. However, deterministic methods remain very worthwhile due to their greater educational value: inspection of analytic equations can make immediately clear the relevance of parameters and can highlight important relationships between components, and deterministic models can easily produce graphical comparisons of parameters, e.g. oscillation frequency versus half-life of an mRNA, through the use of mathematical computing packages. Stochastic simulations are in general computationally demanding, and parametric plots will require a simulation at every desired point on the graph, which may be simply too time-consuming. We find a combination of deterministic and stochastic modelling to be most useful, initially modelling with deterministic techniques to gain insight into the system’s behaviour and the underlying principles of the model, and finally turning to stochastic simulation for accuracy in the final stages of parameter and data fitting. To give a specific example, our laboratory and colleagues developed a mathematical model of transcriptional interference in *E. coli*, which contains three implementations: analytic, ‘mean-field’ (probabilistic), and stochastic, the three techniques producing similar but not identical output (15). The development of this model benefited greatly from the combination of approaches, each providing unique contributions to our understanding of the system.

4.4.2. Level of Detail

The level of detail included heavily influences the educational value of a model. A model with an excessive number of parameters might be able to be made to fit any data set, destroying the ability to discriminate between alternative hypotheses. Conversely, it is important for all biologically relevant behaviour to be included; the dismissal of one seemingly unimportant feature may render a model irredeemably inaccurate. In an interdisciplinary team, proficient communication between modellers and experimental biologists is necessary for the important decisions of which features

are of relevance to the model, and which can be discarded. The choice of detail is especially significant to modellers of biology, as here we face a problem: biochemical systems feature an extraordinarily large number of parameters relative to most physical or chemical systems, thus the inclusion of exhaustive detail in a model may require impossibly extensive experimental investigation to provide the necessary parameters. Much may be omitted without a loss of validity, e.g. a promoter could be characterised in terms of protein production rate, without including mRNA production, translation, and degradation; but if mRNA regulation is a significant feature of the system, this detail may be vital.

4.4.3. Parameter Fitting

Given the number of parameters that are likely to exist in any biological model, it is unlikely that each one has been experimentally measured, requiring parameters to be selected on the basis of those values that allow the model to reproduce experimental data: the process of parameter fitting. This is a procedure performed more easily with deterministic than stochastic models, though it may be found that parameters selected to fit a deterministic model may require adjustment when shifting to stochastic simulation (16). If data have been obtained from replicate experiments, the knowledge of confidence limits at each point is of assistance to the fitting process, as individual data points can be weighted according to the precision of their measurement, by minimising

$$\chi^2 = \sum_i \left[\frac{(\text{experimental value})_i - (\text{model value})_i}{(\text{experimental confidence limit})_i} \right]^2.$$

A model may be held in greater confidence when only a small number of parameters need be fitted to the data. Indeed, with many fewer parameters than data points and a conceptually sound model, the process of fitting the model to the data can itself constitute an accurate method of measuring those parameters. The process of parameter fitting can be assisted by the examination of literature for other measurements or estimates of your parameters, and by checking that fitted parameters produce biologically reasonable values. The process of parameter fitting can itself be an investigative tool: fits that produce unreasonable values may need adjustment, or may in fact be highlighting failures in the model or mistaken assumptions. Biological systems have demonstrated clever techniques to circumvent physical laws, such as the use of dual operators in the *lac* operon, providing the Lac repressor with an effective association rate to a single operon that is apparently faster than diffusion (17); a fit that provides a physically or biologically unreasonable value just might be a clue to an exciting discovery.

*4.4.4. Discrimination
Between Alternative
Hypotheses*

After the construction of a basic framework for a model, a large number of alternative hypotheses can be incorporated into the model and assessed by their ability to fit experimental data while selecting realistic parameters. This may highlight some hypotheses as worthy of direct experimental investigation, while hypotheses incapable of satisfactory fits can be ostensibly discarded, until such time as new data are acquired or the model is revised.

*4.4.5. Long-Term
Development of a Model*

Over the course of a prolonged investigation into a system, the data available for modelling gradually builds, which places an increasing demand on the accuracy of the model: it should be fully consistent with all relevant data. Successfully obtaining a precise match between theory and experiment will become increasingly challenging, but correspondingly the confidence that a successful fit reflects on an accurate model will also increase. With more data to place constraints on a model and fewer parameters in need of fitting, the ability to discriminate between alternative hypotheses improves, increasing the educational value of the model.

**4.5. Make Predictions
with *In Silico*
Experiments**

Theorists may be content to have produced a model that adequately explains all data available for a given system, but for the experimentalist the principal value of a model lies in the ability to guide the choice and design of future experiments. A model can be altered to include whatever changes to the system may be planned for future experiments, such as the introduction of mutations or additional components, or if appropriate a model may be applied to a new system. The predictions of the model constitute an *in silico* experiment; and if the investigators have adequate confidence in the accuracy of the model, based on fits to past data, then the most interesting of these predictions can make excellent targets for wet experimental investigation.

Provided a respectable model is available, *in silico* experiments are well worth the effort, especially given how little effort they take: typically orders of magnitude less time and money than the identical wet experiment. *In silico* experiments can suggest which future experiments are likely to reveal the most useful information; when working quantitatively, *in silico* experiments may demonstrate that a particular perturbation to the system is unlikely to produce a sufficiently large change in the measured variable(s) to answer a question. This may guide the experimentalists to an improved choice of perturbation, potentially saving much time and effort. When a model contains ambiguity, i.e., two or more theories of significant difference that equally explain previous data, it may be valuable to design experiments that distinguish between these alternatives: here *in silico* experiments, performed with competing models, are invaluable. Finally, certain changes to the system under study may produce emergent behaviour that would elude

the intuition: *in silico* experiments can reveal such features and direct the investigator to experiments that may otherwise have been discarded as uninteresting.

A large pool of possible wet experiments can therefore be assessed by *in silico* experiments to select the most promising and make predictions about results. These results are then almost certain to be very interesting: should the model's prediction be correct, there is value in both the result itself and in the predictive power of the model. Should a model fail to anticipate the experimental findings, the previous theoretical understanding of the system has been challenged, and new hypotheses need be incorporated into models in the effort to explain these results. *In silico* experimentation is a good way to identify the wet experiments most likely to refute an accepted hypothesis and 'prove yourself wrong', to drive the development of new theories.

4.6. Introduce New Components

A reasonable benchmark for understanding of the chosen subsystem is the ability of a model to quantitatively explain all available data, preferably including 'diverse' data such as the behaviour of mutants in addition to the wild-type system. Having elucidated the roles of components and their interactions in a chosen subsystem, the system of study can be enlarged to include more components or more inputs/outputs. A quantitatively characterised and understood subsystem provides a theoretical framework of great value to subsequent expansion of the system under study, which is likely to make the identification of new components and new interactions easier than was the initial characterisation of a subsystem.

5. Case Study: Promoter Regulation in Phage λ

In this section we use phage λ research to illustrate the principles of experimental design that facilitate the incorporation of 'bottom-up' modelling into an investigative program, and to provide examples where modelling has provided information inaccessible by other means. In particular, we focus on research into promoter regulation, which we find to be a nexus of decision-making, the product of numerous inputs, and thus a valuable position to probe the behaviour of a system.

5.1. Design of Experiments

In our studies of transcription regulation in temperate bacteriophage, thorough quantitative characterisation of interactions is instrumental in understanding network function. Therefore, we typically aim to measure the response of each promoter to the full physiological range of all components relevant to its regulation. However, there are too many interacting components to deconstruct

behaviour within the whole organism into knowledge of individual interactions, and it is therefore necessary to isolate subsystems of interest.

An example of such a subsystem that has been studied by many labs including our own is the autoregulation of the P_{RM} promoter by its own product CI. P_{RM} is part of the Right Operator (O_R) of phage λ , which contains three adjacent CI dimer-binding sites, O_{R1} , O_{R2} , and O_{R3} , with decreasing affinity for CI dimers. At the strongest binding site O_{R1} , CI represses P_R ; at O_{R2} CI also represses P_R , and activates P_{RM} by cooperative binding to RNA polymerase; at the weakest binding site O_{R3} , CI represses P_{RM} . Within the whole phage, the positive and negative feedback of this small system is complicated by the many other features, such as transcription of cI from P_{RE} , and Cro, which also binds to O_{R1} , 2, and 3. Therefore, to quantitatively investigate this autoregulation, the cI gene and O_R with its promoters P_R and P_{RM} need to be examined in isolation. This can be accomplished by placing the relevant region of phage λ into the *E. coli* chromosome as a single-copy reporter (isolation of a subsystem; **Fig. 15.2**, Step 1). Furthermore, to study the response of P_R and P_{RM} to a range of CI concentrations, cI is not placed downstream of P_{RM} , subject to autoregulatory control, but placed on a plasmid under the control of an inducible promoter (decoupling key components; **Fig. 15.2**, Step 2), with *lacZ* replacing cI downstream of P_{RM} . By varying the concentration of inducer, the response of P_{RM} to a wide range of CI concentrations can be examined in detail. To further understand the properties of CI autoregulation, measurements of P_{RM} activity were also performed in the presence of mutations to O_{R3} , which strengthen (*cI2*) or abolish (*rI*) CI binding (acquiring a diverse data set). These experiments showed that P_{RM} was not repressed even at high CI concentrations, and that the *rI* and *cI2* mutations had very little impact on P_{RM} activity (**Fig. 15.3**), indicating that CI association to O_{R3} and repression of P_{RM} was not influential to this system (4).

These findings in the isolated system were compared with results obtained in the whole phage, by introducing the *rI* and *cI2* mutations to λ phage (checking behaviour in the full context; **Fig. 2**, Step 3). These mutations revealed phenotypic changes in the process of UV induction of lysogens, wherein CI is degraded in response to UV-induced DNA damage resulting in lysis of the host and release of phage. The *rI* mutation, which makes little change to Cro association to O_{R3} but substantially weakens CI binding, produced defective UV induction, while the *cI2* mutation, which strengthens CI association and weakens Cro binding to O_{R3} , induced more readily. This suggested higher [CI] in the *rI* lysogen and lower [CI] in the *cI2* lysogen, subsequently confirmed by experiment, indicating that these mutations did indeed alter P_{RM} activity, despite the apparent lack of effect on isolated P_{RM} (4).

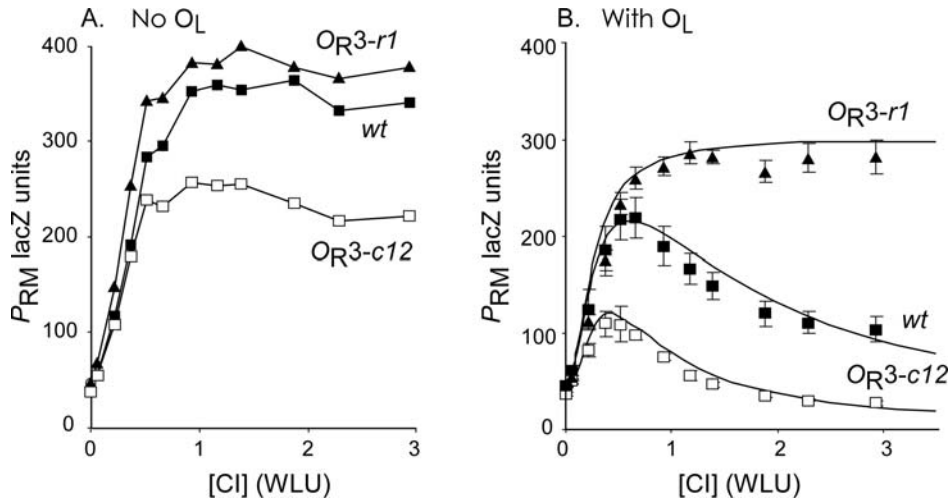


Fig. 15.3. (A) Activity of wild-type and mutant P_{RM} promoters in the absence of O_L . Reproduced from (4). (B) As (A) but in the presence of O_L . Points with error bars show 95% confidence limits of experimental measurements. Solid lines represent the result of physicochemical modelling of CI regulation, incorporating O_R - O_L long-range interactions. Reproduced from (19).

Drawing on previous observations that CI can mediate DNA loops between O_R and the Left Operator (O_L) (18), it was reasoned that the discrepancy between studies of whole phage (as lysogens) and of isolated P_{RM} was the presence of O_L . The introduction of O_L to the isolated P_{RM} *lacZ* reporters revealed substantial repression of P_{RM} at physiological CI concentrations, confirmed by enhancement of repression in a *c12* mutant and lack of repression in an *r1* mutant (4) (Fig. 15.3). This experimental procedure was able to extract from a complex network that O_R to O_L DNA looping plays a critical role in the regulation of P_{RM} , and produced data of a quality enabling a thorough statistical mechanical model of this process. Subsequent work not detailed here confirmed a model in which CI bound to O_{R1} and O_{R2} forms an octamer with CI at O_{L1} and O_{L2} , forming a long-range DNA loop, allowing a CI tetramer to form from dimers at O_{R3} and O_{L3} , stabilising occupation of O_{R3} and P_{RM} repression (19) (Fig. 15.4).

5.2. Computational Modelling of Data

The wealth of thermodynamic data on CI association to operators in O_R and O_L permitted the description of the system by the partition function, which relates the probability of a system to exist in state ' i ', P_i , to the standard free energy of that state, ΔG_i , through

$$P_i \propto \exp\left(\frac{-\Delta G_i}{RT}\right),$$

where R is the gas constant and T is the temperature in Kelvin. Degeneracy must be explicitly accounted for, e.g. when modelling the binding of repressors to operators, a state containing ' n '-bound

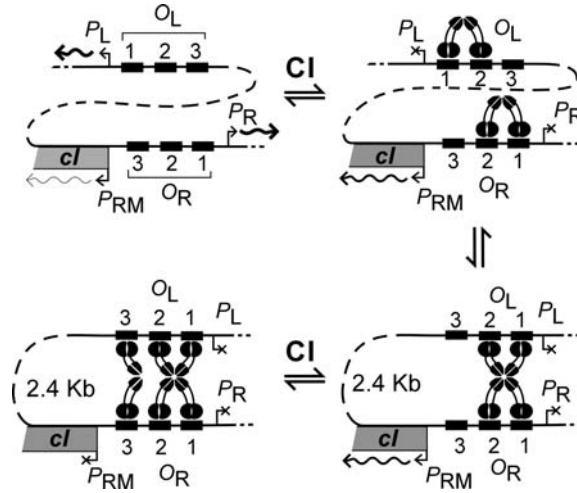


Fig. 15.4. Model of CI regulation with long-range DNA looping. Cartoon depicting the major predicted CI: DNA complexes at O_R and O_L on the λ chromosome and their effects on transcription as CI concentration increases. Reproduced from (19).

repressors requires an additional factor of (number of repressors)ⁿ, as the same state can be composed in this number of different ways, each known as a microstate. Hence

$$P_i = \frac{[\text{repressor}]^{n_i} \exp\left(\frac{-\Delta G_i}{RT}\right)}{Z},$$

where Z is a normalisation factor known as the partition function:

$$Z = \sum_i [\text{repressor}]^{n_i} \exp\left(\frac{-\Delta G_i}{RT}\right)$$

Previous measurements of CI affinity for all six operators at O_R and O_L , as well as cooperative CI interactions between adjacent dimers, permitted the creation of a partition function describing all possible combinations of CI binding, as well as long-range DNA looping between O_R and O_L . By detailing the prevalence of each species present at any $[CI]$, it is possible to fully describe P_R and P_{RM} activities as a function of $[CI]$. The experimental data to be fitted by this model were measurements of P_R and P_{RM} activities as a function of $[CI]$, with and without O_L , and also with rI or $cI2$ mutations at O_{R3} . Out of 29 parameters in this model (free energies of CI association/cooperativity, and basal/activated/repressed promoter activities), 26 had been explicitly measured, leaving only 3 degrees of freedom with which to fit the model to data. These three parameters were the strength of non-specific DNA binding by CI, the free energy of formation of a long-range DNA loop maintained by a CI octamer bound to O_{R1} , O_{R2} , O_{L1} , and O_{L2} (ΔG_{oct}), and the free energy of formation of a CI tetramer across

O_{R3} and O_{L3} (ΔG_{tet}), which is reasoned to only form once the DNA loop has been stabilised by octamer formation. This model was able to accurately reproduce all data, but for an overestimate of P_R repression at low [CI], lending strong support to the model of O_R - O_L DNA looping and its role in pRM autoregulation (19).

The full implications of this research are discussed in more detail elsewhere (4, 19–21); here is described two key conclusions that were provided solely by the thermodynamic model. Describing the prevalence of each species as a function of [CI] demonstrated that the lysogenic CI concentration lies precisely at the point of the sharpest transition between states of active P_{RM} and of repressed P_{RM} ; therefore, the lysogenic state appears poised to produce the most sensitive response in P_{RM} activity to dampen fluctuations in [CI], providing optimum stability and resistance to molecular noise. Computational modelling thus provided the means to make this striking observation of network architecture, which could not have been directly acquired from the experimental methods of this study: it was necessary though to acquire data of a quality enabling modelling. By fitting ΔG_{oct} and ΔG_{tet} to P_{RM} activities (Fig. 15.3), it was possible to estimate the in vivo free energy of DNA looping between operators 3.8 kb apart. This combination of quantitative experimental data and a partition function to produce a thermodynamic model is the only technique that has yet produced in vivo measurements of the energetics of DNA looping. This technique has been used to probe the in vivo mechanical properties of DNA in work, which challenges prevailing models of DNA (22).

The same approaches to experimental design and data collection described in this case study do not necessarily require a system as extensively characterised as λ in order to facilitate computational modelling. Similar approaches were applied to the study of transcriptional interference between two promoters in phage 186, for the purpose of characterising the molecular mechanisms of interference (23, 24). On the basis of the diverse set of standardised, quantitative, in vivo data collected, a general mathematical model of transcriptional interference by RNA polymerase traffic in *E. coli* was developed (15). This model was able to accurately explain all data in the studies of (23), and further enhanced our understanding of the mechanisms of interference observed in this and other studies. Our laboratory is currently using a combination of in vivo and in silico experiments to discriminate between different hypotheses for the transcriptional interference observed in other systems, where the model is demonstrating significant predictive power. As demonstrated, a variety of experimental programs in our laboratory have benefited from bottom-up computational modelling, whose application has been facilitated by the approach to experimental design and data collection described here. We therefore expect that these principles can find widespread utility in

the study of regulatory networks, in enabling the construction of bottom-up computational models and their use as experimental tools.

Acknowledgements

We thank J. Barry Egan and Ian B. Dodd for discussions. Research in our laboratory is supported by the U.S. NIH (GM062976) and the Australian Research Council.

References

1. Kolch W, Calder M, Gilbert D. When kinases meet mathematics: the systems biology of MAPK signalling. *FEBS Lett* 2005;579(8):1891–5.
2. Kitano H. Computational systems biology *Nature* 2002;420(6912):206–10.
3. Miller JH, ed. Experiments in Molecular Genetics. Cold Spring Harbor, NY: Cold Spring Harbor Laboratory Press; 1972.
4. Dodd IB, Perkins AJ, Tsemitsidis D, Egan JB. Octamerization of lambda CI repressor is needed for effective repression of P(RM) and efficient switching from lysogeny. *Genes Dev* 2001;15(22):3013–22.
5. Simons RW, Houman F, Kleckner N. Improved single and multicopy lac-based cloning vectors for protein and operon fusions. *Gene* 1987;53(1):85–96.
6. Haldimann A, Wanner BL. Conditional-replication, integration, excision, and retrieval plasmid-host systems for gene structure-function studies of bacteria. *J Bacteriol* 2001;183(21):6384–93.
7. Jensen PR, Hammer K. Artificial promoters for metabolic optimization. *Biotechnol Bioeng* 1998;58(2–3):191–5.
8. Su LT, Agapito MA, Li M, et al. TRPM7 regulates cell adhesion by controlling the calcium-dependent protease calpain. *J Biol Chem* 2006;281(16):11260–70.
9. Linn T, St Pierre R. Improved vector system for constructing transcriptional fusions that ensures independent translation of lacZ. *J Bacteriol* 1990;172(2):1077–84.
10. Liang S, Bipatnath M, Xu Y, et al. Activities of constitutive promoters in *Escherichia coli*. *J Mol Biol* 1999;292(1):19–37.
11. Blakeslee S. Scientist at Work: John Henry Holland; searching for simple rules of complexity. *The New York Times* 1995 December 26, 1995.
12. Atsumi S, Little JW. Regulatory circuit design and evolution using phage lambda. *Genes Dev* 2004;18(17):2086–94.
13. Atsumi S, Little JW. Role of the lytic repressor in prophage induction of phage lambda as analyzed by a module-replacement approach. *Proc Natl Acad Sci U S A* 2006;103(12):4558–63.
14. Raser JM, O’Shea EK. Noise in gene expression: origins, consequences, and control. *Science* 2005;309(5743):2010–3.
15. Sneppen K, Dodd IB, Shearwin KE, et al. A mathematical model for transcriptional interference by RNA polymerase traffic in *Escherichia coli*. *J Mol Biol* 2005;346(2):399–409.
16. Forger DB, Peskin CS. Stochastic simulation of the mammalian circadian clock. *Proc Natl Acad Sci U S A* 2005;102(2):321–4.
17. Vilar JM, Leibler S. DNA looping and physical constraints on transcription regulation. *J Mol Biol* 2003;331(5):981–9.
18. Revet B, von Wilcken-Bergmann B, Bessert H, Barker A, Muller-Hill B. Four dimers of lambda repressor bound to two suitably spaced pairs of lambda operators form octamers and DNA loops over large distances. *Curr Biol* 1999;9(3):151–4.
19. Dodd IB, Shearwin KE, Perkins AJ, Burr T, Hochschild A, Egan JB. Cooperativity in long-range gene regulation by the lambda CI repressor. *Genes Dev* 2004;18(3):344–54.
20. Hochschild A. The lambda switch: cI closes the gap in autoregulation. *Curr Biol* 2002;12(3):R87–9.

21. Ptashne M. A Genetic switch. Phage Lambda Revisited. 3rd ed. Cold Spring Harbor, N.Y.: Cold Spring Harbor Laboratory Press; 2004.
22. Saiz L, Rubi JM, Vilar JM. Inferring the in vivo looping properties of DNA. *Proc Natl Acad Sci U S A* 2005;102(49):17642–5.
23. Callen BP, Shearwin KE, Egan JB. Transcriptional interference between convergent promoters caused by elongation over the promoter. *Mol Cell* 2004;14(5):647–56.
24. Shearwin KE, Callen BP, Egan JB. Transcriptional interference – a crash course. *Trends Genet* 2005;21(6):339–45.

Chapter 16

A Maximum Likelihood Method for Reconstruction of the Evolution of Eukaryotic Gene Structure

Liran Carmel, Igor B. Rogozin, Yuri I. Wolf, and Eugene V. Koonin

Abstract

Spliceosomal introns are one of the principal distinctive features of eukaryotes. Nevertheless, different large-scale studies disagree about even the most basic features of their evolution. In order to come up with a more reliable reconstruction of intron evolution, we developed a model that is far more comprehensive than previous ones. This model is rich in parameters, and estimating them accurately is infeasible by straightforward likelihood maximization. Thus, we have developed an expectation-maximization algorithm that allows for efficient maximization. Here, we outline the model and describe the expectation-maximization algorithm in detail. Since the method works with intron presence-absence maps, it is expected to be instrumental for the analysis of the evolution of other binary characters as well.

Key words: Maximum likelihood, expectation-maximization, intron evolution, ancestral reconstruction, eukaryotic gene structure.

1. Introduction

In eukaryotes, many protein-coding genes have their coding sequence broken into pieces – the exons – separated by the non-coding spliceosomal introns. These introns are removed from the nascent pre-mRNA and the exons are spliced together to form the intronless mRNA by the spliceosome, a large and elaborate macromolecular complex comprising several small RNA molecules and numerous proteins. No spliceosomal introns have ever been found in prokaryotes, and there are no eukaryotes with a completely sequenced genomes, not even the very basal ones, which would not possess introns (1–3) and the accompanying splicing machinery (4).

Despite the introns being such a remarkable idiosyncrasy of eukaryotic genomes, their origin and evolution are not thoroughly understood (5, 6). It is generally accepted that introns can be regarded as units of evolution and that their presence/absence pattern is a result of stochastic processes of loss and gain. However, the nature of these processes is vigorously debated. Recent large-scale attempts to study these processes using extant eukaryotic genomes led to incongruent conclusions.

In a study on reconstruction of intron evolution, Rogozin et al. (7) analyzed ~700 sets of intron-bearing orthologous genes from eight eukaryotic species. The multiple alignment of the orthologs within each set was computed, and the intron positions were projected on the alignments to form presence/absence maps. Using Dollo parsimony to infer ancestral states, these authors observed a diverse repertoire of behaviors. Some lineages endured extensive losses, while others experienced mostly gain events. Early forerunners, such as the last common ancestor of multicellular life, were shown to be relatively intron-rich. This work suggested that both gain and loss of introns played significant roles in shaping the modern eukaryotic gene structure. However, as these inferences rely upon the Dollo parsimony reconstruction, the number of gains in terminal branches (leaves of the phylogenetic tree) is overestimated, resulting in underestimation (potentially, significant) of the number of introns in ancient lineages.

The same data set was analyzed by Roy and Gilbert (8, 9) using a different methodology. They adopted a simple evolutionary model, according to which different lineages are associated with different loss and gain probabilities. Using a variation on maximum likelihood estimation, they obtained considerably higher estimates for the number of introns in early eukaryotes and a correspondingly lower level of gains in all lineages, i.e., a clear dominance of loss events in the evolution of eukaryotic genes. Roy and Gilbert have substantially simplified the mathematics involved in the estimation procedure, at the expense of introducing into the computation considerations of parsimony, which yielded an inference technique that is a hybrid between parsimony and maximum likelihood. This hybrid, however, excludes from consideration different evolutionary scenarios, resulting in inflated estimates of the number of introns in early eukaryotes (10).

The model of Roy and Gilbert is *branch-specific*, i.e., it assumes that the gain and loss rates depend only on the branch, thus tacitly presuming that all genes behave identically with respect to intron gain and loss. Exactly the inverse approach was adopted by Qiu et al. (11). These authors developed a *gene-specific* model, whereby different gene families are characterized by different rates of intron gain and loss, but for a particular gene these rates are constant across the entire phylogenetic tree. They used a different data set combined with a Bayesian estimation technique and concluded

that almost all extant introns were gained during the eukaryotic evolution. This suggests evolution is dominated by intron gain events with few losses. However, the validity of a gene-specific model is disputable as it is hard to reconcile with the accumulating evidence on large differences between lineages (12–15).

Recently, two maximum likelihood estimation techniques have been developed for essentially the same branch-specific evolutionary model as the one of Roy and Gilbert. Csuros (10) used a direct approach, while Nguyen et al. (16) developed an expectation-maximization algorithm. Both methods encountered the same problem of estimating the number of unobserved intronless sites. Each employed a technically different but conceptually similar method to evaluate this number. Both techniques were applied to the eight-species data of Rogozin et al. (7), yielding very close estimates. As expected, these methods predict intron occupancy level of ancient lineages higher than those predicted by Dollo parsimony and lower than those predicted by the hybrid technique of Roy and Gilbert. Notably, these estimates are generally closer to those obtained using Dollo parsimony, and they imply an evolutionary landscape comprising both losses and gains, with some excess of gains.

While the Dollo parsimony (7) and the hybrid technique of Roy and Gilbert (8, 9) showed some methodological biases, the other analyses of intron evolution (10, 11, 16) used well-established estimation techniques. Nevertheless, these studies kept yielding widely diverging inferences. The reason seems to be the differences in the underlying evolutionary models, neither being sufficient to describe the complex reality of intron evolution. The branch-specific model fails to account for important differences between genes, whereas the gene-specific model ignores the sharp differences between lineages. Additionally, rate variability between sites, known to be an important factor in other fields of molecular evolution (17, 18), should be taken into account also in the evolution of gene structure. This is particularly important for intron gain in light of the accumulating evidence in favor of the proto-splice model, according to which new introns are preferentially inserted inside certain sequence motifs (19–21). This means that sites could dramatically differ in their gain rate depending on their position relative to a proto-splice site.

Here we describe a model of evolution that takes into consideration all of the above factors. In order to efficiently estimate the model parameters by maximum likelihood, we have developed an expectation-maximization algorithm. We also compiled a data set that is considerably larger than previously used ones, consisting of 400 sets of orthologous genes from 19 eukaryotic species. Applying our algorithm to this data set, we obtained high-precision estimates, revealing a fascinating evolutionary history of gene structure, where both losses and gains played significant roles

albeit the contribution of losses was somewhat greater. Moreover, we identified novel properties of intron evolution: (i) all eukaryotic lineages share a common, universal, mode of intron evolution, whereby the loss and gain processes are positively correlated. This suggests that the mechanisms of intron gain and loss share common mechanistic components. In some lineages, additional forces come into play, resulting either in elevated loss rate or in elevated gain rate. Lineages exhibiting an increased loss rate are dispersed throughout the entire phylogenetic tree. In contrast, lineages with excessive gains are much rarer, and all of them are ancient. (ii) Intron loss rates of individual genes show no correlation with any other genomic property. By contrast, intron gain rate of individual genes show several remarkable relationships, not always easily explained. In brief, intron gain rate is positively correlated with expression level, negatively correlated with sequence evolution rate, and negatively correlated with the gene length. Moreover, genes of apparent bacterial origin have significantly lower rates of intron gain than genes of archaeal origin. (iii) We showed that the remarkable conservation of intron positions is, mainly ($\sim 90\%$), due to shared ancestry, and only in a minority of the cases ($\sim 10\%$), due to parallel gain at the same location. (iv) We determined that the density of potential intron insertion sites is about 1 site per 7 nucleotides.

2. Materials

The algorithm learns the parameters of the model by comparing the structure of orthologous genes in extant species. To carry out this comparison, it requires two sets of input data, to be described in this section. The first is a phylogenetic tree, defining topological relationships between a set of eukaryotic species. The second is a collection of genes, for which one can identify orthologs in at least a subset of the species above.

2.1. Multiple Alignments

Suppose that we have G sets of aligned orthologous genes from S species. To represent the gene structure, we transform these alignments into intron presence–absence maps by substituting for each nucleotide (or amino acid) 0 or 1, depending on whether an intron is present or absent in the respective position. We allow for missing data by using a third symbol (*), and consequently a gene might be included in the input data even if it is missing in part of the species. Every site in an alignment, called *pattern*, is a vector of length S over the alphabet $(0,1,*)$. Let Ω be the total number of unique patterns in the entire set of G alignments, denoted $\omega_1, \dots, \omega_\Omega$, and let n_{gp} count the number of times pattern ω_p is found in the multiple alignment of gene g . Assuming that the sites evolve

independently, the set $M_g = (n_{g1}, \dots, n_{g\Omega})$ fully characterizes the multiple alignment of the g th gene. Thus, all the relevant information about the multiple alignments is captured by the list of unique patterns $\omega_1, \dots, \omega_\Omega$, and the list of vectors M_1, \dots, M_G .

2.2. Phylogenetic Tree

Let T be a rooted bifurcating phylogenetic tree with S leaves (terminal nodes) corresponding to the S species above. The total number of nodes in T is $N = 2S - 1$, and we index them by $t = 0, 1, \dots, N - 1$, with the convention that zero is the root node. The state of node t is described by the random variable q_t , which can take the values 0 and 1 (and * in leaves). We use V_t for the set of all leaves such that node t is among their ancestors. The entire collection of leaves is, obviously, V_0 . The parent node of t is denoted $P(t)$. We use the special notations q_t^P and V_t^P for $q_{P(t)}$ and $V_{P(t)}$, respectively. Analogously, the two direct descendants of node t are denoted $L(t)$ and $R(t)$, and we use the special notations q_t^L , q_t^R , V_t^L , and V_t^R for $q_{L(t)}$, $q_{R(t)}$, $V_{L(t)}$, and $V_{R(t)}$, respectively. We index the branches by the node into which they are leading, and use Δ_t to denote the length (in time units) of the t th branch. We assume that the tree topology, as well as all the branch lengths $\Delta_1, \dots, \Delta_{N-1}$ are known.

3. Methods

3.1. The Probabilistic Model

A graphical model is a mathematical graph whose nodes symbolize random variables, and whose branches describe dependence relationships between them (22). A bifurcating phylogenetic tree, when viewed as a graphical model, depicts the probabilistic model

$$\Pr(q_0) \prod_{t=1}^{N-1} \Pr(q_t | q_t^P). \tag{1}$$

We use the notation $\pi_i = \Pr(q_0 = i)$ to describe the prior probability of the root, and $A_{ij}(g, t) = \Pr(q_t = j | q_t^P = i, g)$ to describe the transition probability for gene g along branch t . In our model, we assume that the transition probability depends on both the gene and the branch, and that it takes the explicit form

$$A(g, t) = \begin{pmatrix} 1 - \xi_t(1 - e^{-\eta_g \Delta_t}) & \xi_t(1 - e^{-\eta_g \Delta_t}) \\ 1 - (1 - \phi_t)e^{-\theta_g \Delta_t} & (1 - \phi_t)e^{-\theta_g \Delta_t} \end{pmatrix}. \tag{2}$$

Here, η_g and θ_g are nonnegative parameters, determining the intron gain and loss rates, respectively, of gene g . Complementarily, ξ_t and ϕ_t determine the intron gain and loss coefficients of branch t , respectively, and are bound to the range $0 \leq \xi_t, \phi_t \leq 1$.

The probability of an intron present in gene g at the beginning of branch t to be retained along the branch is $(1 - \phi_t)e^{-\theta_g \Delta_t}$, that is, it is retained only if the branch does not lose it (with probability $1 - \phi_t$), and also the gene does not lose it (with probability $e^{-\theta_g \Delta_t}$). This comes to reflect a reality where strong forces to strip a gene off its introns will be practically unaffected by the particular lineage, and, oppositely, strong forces to strip a lineage off its introns will be practically unaffected by the particular gene. In the same spirit, the probability of an intron to be gained in gene g along branch t is $\xi_t(1 - e^{-\eta_g \Delta_t})$, that is, it is gained only if both the branch “approves” it (with probability ξ_t) and the gene “approves” it (with probability $1 - e^{-\eta_g \Delta_t}$).

In other fields of molecular evolution, it was long realized that analysis precision improves if one allows for rate variability across sites (17, 18). Typically, such rate variability is modeled by introducing a *rate variable*, r , which scales, for each site, the time units of the phylogenetic tree, $\Delta_t \leftarrow r \cdot \Delta_t$. This rate variable is a random variable, distributed according to a distribution function with non-negative domain and unit mean, typically the unit-mean gamma distribution. The rate variability reflects the idea that sites differ in their rate of evolution. Specifically, there are fast-evolving sites ($r \gg 1$), as well as slow-evolving ones ($r < 1$). In our model of intron evolution we extend this idea by assuming that the gain and loss processes are subject to rate variability, independently of each other. Hence, a site can have any combination of gain and loss rates. To accommodate this idea, we use two independent rate variables, r^g and r^θ , that are used to scale, for each site, the gene-specific gain rate, $\eta_g \leftarrow r^g \cdot \eta_g$, and the gene-specific loss rate, $\theta_g \leftarrow r^\theta \cdot \theta_g$. We further assume that the distributions of these rate variables are independent of the genes, and are explicitly given by

$$\begin{aligned} r^g &\sim v\delta(\eta) + (1 - v)\Gamma(\eta; \lambda_\eta) \\ r^\theta &\sim \Gamma(\theta; \lambda_\theta). \end{aligned} \tag{3}$$

Here, $\Gamma(x; \lambda)$ is the unit-mean gamma distribution of variable x with shape parameter λ , $\delta(x)$ is the Dirac delta-function, and v is the fraction of sites that are assumed to have zero gain rate. These latter sites, denoted *invariant sites*, reflect these sites that are not a proto-splice site (19–21). Intron loss does not have an invariant counterpart, as the assumption is that once an intron is gained, it can always be lost. Therefore, the loss rate variable is assumed to be distributed according to a gamma distribution, which is by far the most popular in describing rate variability (17, 18, 23).

In practice, the rate distributions in Eq. [3] are rendered discrete (24). We assume that the gain rate variable can take K_η discrete values $r_1^g = 0, r_2^g, \dots, r_{K_\eta}^g$ with probabilities $f_1^g = v, f_2^g, \dots, f_{K_\eta}^g$ such

that $\sum_{k=1}^{K_\eta} f_k^\eta = 1$. Analogously, we assume that the loss rate variable can take K_θ discrete values $r_1^\theta, \dots, r_{K_\theta}^\theta$ with probabilities $f_1^\theta, \dots, f_{K_\theta}^\theta$ such that $\sum_{k=1}^{K_\theta} f_k^\theta = 1$. For a particular gain rate value r_k^η , we denote the actual gain rate $r_k^\eta \cdot \eta_g$ by η_{kg} . Similarly, for a particular loss rate value r_k^θ , we denote the actual loss rate $r_k^\theta \cdot \theta_g$ by θ_{kg} .

For notational clarity, we aggregate the model parameters into a small number of sets. To this end, let $\Xi_t = \{\zeta_t, \phi_t\}$ be the set of parameters that are specific for branch t , and let $\Xi = (\Xi_1, \dots, \Xi_{N-1})$ be the set of all branch-specific parameters. Similarly, let $\Psi_g = (\eta_g, \theta_g)$ be the set of parameters that are specific for gene g , and let $\Psi = (\Psi_1, \dots, \Psi_G)$ be the set of all gene-specific parameters. Additionally, we denote by $\Lambda = (v, \lambda_\eta, \lambda_\theta)$ the parameters that determine the rate variability. When the distinction between the different sets of parameters is irrelevant, we shall use $\Theta = (\Xi, \Psi, \Lambda)$ as the set of all the model's parameters. We achieve further succinctness in notations by denoting the actual gene-specific rate values for particular values r_k^η and r_k^θ of the rate variables as $\Psi_{kk'g} = (\eta_{kg}, \theta_{k'g})$.

3.2. The EM Algorithm

For each site, the S leaves form a set of observed random variables, their states being described by the corresponding pattern ω_p . The state of all the internal nodes, denoted σ , form a set of hidden random variables, that is, random variables whose state is not observed. In order to account for rate variability across sites, we associate with each pattern two hidden random variables, ρ_p^η and ρ_p^θ , that determine the value of the rate variables in that site. To sum up, the observed random variables are ω_p , and the hidden random variables are $(\sigma, \rho_p^\eta, \rho_p^\theta)$.

We assume that sites within a gene, as well as the genes themselves, evolve independently. Therefore, the total likelihood can be decomposed as

$$L(M_1, \dots, M_G | \Theta) = \prod_{g=1}^G L(M_g | \Xi, \Psi_g, \Lambda) = \prod_{g=1}^G \prod_{p=1}^{\Omega} L(\omega_p | \Xi, \Psi_g, \Lambda)^{n_{gp}}.$$

and so

$$\log L(M_1, \dots, M_G | \Theta) = \sum_{g=1}^G \sum_{p=1}^{\Omega} n_{gp} \log L(\omega_p | \Xi, \Psi_g, \Lambda). \quad [4]$$

According to the well-known EM paradigm (25) $\log L(M_1, \dots, M_G | \Theta)$ is guaranteed to increase as long as we maximize the auxiliary function

$$Q(\Theta, \Theta^0) = \sum_{g=1}^G \sum_{p=1}^{\Omega} n_{gp} Q_{gp}(\Xi, \Psi_g, \Lambda, \Xi^0, \Psi_g^0, \Lambda^0), \quad [5]$$

where

$$Q_{gp}(\Xi, \Psi_g, \Lambda, \Xi^0, \Psi_g^0, \Lambda^0) = \sum_{\sigma, \rho_p^\eta, \rho_p^\theta} \Pr(\sigma, \rho_p^\eta, \rho_p^\theta | \omega_p, \Xi^0, \Psi_g^0, \Lambda^0) \quad [6]$$

$$\log \Pr(\omega_p, \sigma, \rho_p^\eta, \rho_p^\theta | \Xi, \Psi_g, \Lambda).$$

Using some manipulations (see **Note 1**), this can be written as

$$Q_{gp}(\Xi, \Psi_g, \Lambda, \Xi^0, \Psi_g^0, \Lambda^0) = \sum_{k=1}^{K_\eta} \sum_{k'=1}^{K_\theta} \left[\Pr(\rho_p^\eta = k, \rho_p^\theta = k' | \omega_p, \Xi^0, \Psi_g^0, \Lambda^0) \right] \cdot \left[\sum_{\sigma} \Pr(\sigma | \omega_p, \Xi^0, \Psi_{gk k'}^0) \cdot \{ \log f_k^\eta + \log f_{k'}^\theta + \log \Pr(\omega_p, \sigma | \Xi, \Psi_{gk k'}) \} \right].$$

Denoting by $w_{gpk k'}$ and $Q_{gpk k'}$ the first and second square brackets, respectively, this expression becomes

$$Q_{gp}(\Xi, \Psi_g, \Lambda, \Xi^0, \Psi_g^0, \Lambda^0) = \sum_{k=1}^{K_\eta} \sum_{k'=1}^{K_\theta} w_{gpk k'} Q_{gpk k'}, \quad [7]$$

and consequently

$$Q(\Theta, \Theta^0) = \sum_{g=1}^G \sum_{p=1}^{\Omega} \sum_{k=1}^{K_\eta} \sum_{k'=1}^{K_\theta} n_{gp} w_{gpk k'} Q_{gpk k'}. \quad [8]$$

3.2.1. The E-Step

In this step we compute the function $Q(\Theta, \Theta^0)$, or, equivalently, the set of coefficients $w_{gpk k'}$ and $Q_{gpk k'}$. We accomplish this with the aid of an inward–outward recursion on the tree.

3.2.1.1. The Inward (γ) Recursion

Here we propose a variation on the well-known Felsenstein’s pruning algorithm (26). Let us associate with each node t (except for the root) a vector $\gamma_i^{gpk k'}(t) = \Pr(V_t | q_t^p = i, \Xi^0, \Psi_{gk k'}^0)$. In words, $\gamma_i^{gpk k'}(t)$ is the probability of observing the nodes V_t (which are a subset of the pattern ω_p) for a gene g , when the gain and loss rate variables are r_k^η and $r_{k'}^\theta$, respectively, and when the parent node of t is known to be in state i . By definition, this function is initialized at all leaves ($t \in V_0$) by

$$\gamma(t \in V_0) = \begin{cases} \begin{pmatrix} 1 - \xi_t(1 - e^{-\eta_{gk} \Delta_t}) \\ 1 - (1 - \phi_t)e^{-\theta_{gk'} \Delta_t} \end{pmatrix} & q_t = 0 \\ \begin{pmatrix} \xi_t(1 - e^{-\eta_{gk} \Delta_t}) \\ (1 - \phi_t)e^{-\theta_{gk'} \Delta_t} \end{pmatrix} & q_t = 1. \end{cases} \quad [9]$$

Here, and in the derivations to follow, we omit the superscript from γ . For all internal nodes (except for the root), γ is computed using the recursion

$$\gamma_i(t) = \sum_{j=0}^1 A_{ij}(g, t) \tilde{\gamma}_j(t), \quad [10]$$

where $\tilde{\gamma}_j(t)$ is defined as $\gamma_j[L(t)]\gamma_j[R(t)]$ (see **Note 2**).

The γ -recursion allows for computing the likelihood of any observed pattern ω_p , given the values of the rate variables:

$$\begin{aligned} \Pr(\omega_p | \Xi^0, \Psi_{gkk'}^0) &= \Pr(V_0 | \Xi^0, \Psi_{gkk'}^0) = \Pr(V_0^L, V_0^R | \Xi^0, \Psi_{gkk'}^0) = \\ &= \sum_{i=0}^1 \Pr(V_0^L, V_0^R, q_0 = i | \Xi^0, \Psi_{gkk'}^0) = \\ &= \sum_{i=0}^1 \Pr(q_0 = i | \Xi^0, \Psi_{gkk'}^0) \cdot \Pr(V_0^L | q_0 = i, \Xi^0, \Psi_{gkk'}^0) \cdot \\ &\quad \Pr(V_0^R | V_0^L, q_0 = i, \Xi^0, \Psi_{gkk'}^0). \end{aligned}$$

Given q_0 , V_0^R is independent of V_0^L , and so

$$\Pr(V_0^R | V_0^L, q_0 = i, \Xi^0, \Psi_{gkk'}^0) = \Pr(V_0^R | q_0 = i, \Xi^0, \Psi_{gkk'}^0),$$

and

$$\Pr(\omega_p | \Xi^0, \Psi_{gkk'}^0) = \sum_{i=0}^1 \pi_i \tilde{\gamma}_i(0). \quad [11]$$

This γ -recursion can be easily modified to incorporate missing data (*see Note 3*).

3.2.1.2. The Outward (α) Recursion

Once the γ -recursion is computed, we can use it to compute a second, complementary, recursion. To this end, let us associate with each node t (except for the root node) a matrix $\alpha_{ij}^{gpkk'}(t) = \Pr(q_t = j, q_t^P = i | \omega_p, \Xi^0, \Psi_{gkk'}^0)$. It is beneficial to define for each node t (except for the root node) a vector $\beta_j^{gpkk'}(t) = \sum_{i=0}^1 \alpha_{ij}^{gpkk'}(t) = \Pr(q_t = j | \omega_p, \Xi^0, \Psi_{gkk'}^0)$. Upon the computation of α , β is readily computed too. Again, omitting the superscripts, α can be initialized from its definition on the two direct descendants of the root,

$$\alpha(D(0)) = \frac{1}{\Pr(\omega_p | \Xi^0, \Psi_{gkk'}^0)} \begin{cases} \begin{pmatrix} \pi_0 \gamma_0(\bar{D}(0)) A_{00}(g, D(0)) & 0 \\ \pi_1 \gamma_1(\bar{D}(0)) A_{10}(g, D(0)) & 0 \end{pmatrix} & D(0) \in V_0, q_0^D = 0 \\ \begin{pmatrix} 0 & \pi_0 \gamma_0(\bar{D}(0)) A_{01}(g, D(0)) \\ 0 & \pi_1 \gamma_1(\bar{D}(0)) A_{11}(g, D(0)) \end{pmatrix} & D(0) \in V_0, q_0^D = 1 \\ \begin{pmatrix} \pi_0 \gamma_0(\bar{D}(0)) \tilde{\gamma}_0(D(0)) A_{00}(g, D(0)) & \pi_0 \gamma_0(\bar{D}(0)) \tilde{\gamma}_1(D(0)) A_{01}(D(0)) \\ \pi_1 \gamma_1(\bar{D}(0)) \tilde{\gamma}_0(D(0)) A_{10}(D(0)) & \pi_1 \gamma_1(\bar{D}(0)) \tilde{\gamma}_1(D(0)) A_{11}(D(0)) \end{pmatrix} & D(0) \notin V_0. \end{cases} \quad [12]$$

Here, $D(0)$ stands for any one of the direct descendants of the root, and $\bar{D}(0)$ is its sibling. For any other internal node, α is computed using the outward-recursion

$$\alpha(t) = \begin{pmatrix} \beta_0(P(t)) \tilde{\gamma}_0(t) A_{00}(g, t) / \gamma_0(t) & \beta_0(P(t)) \tilde{\gamma}_1(t) A_{01}(g, t) / \gamma_0(t) \\ \beta_1(P(t)) \tilde{\gamma}_0(t) A_{10}(g, t) / \gamma_1(t) & \beta_1(P(t)) \tilde{\gamma}_1(t) A_{11}(g, t) / \gamma_1(t) \end{pmatrix} \quad [13]$$

(see **Note 4**).

Finally, for each leaf that is not a descendant of the root,

$$\alpha(t) = \begin{cases} \begin{pmatrix} \beta_0(P(t)) & 0 \\ \beta_1(P(t)) & 0 \end{pmatrix} & q_t = 0 \\ \begin{pmatrix} 0 & \beta_0(P(t)) \\ 0 & \beta_1(P(t)) \end{pmatrix} & q_t = 1. \end{cases} \quad t \in V_0, P(t) \neq 0 \quad [14]$$

Again, this recursion can be straightforwardly modified when missing data are present (see **Note 5**).

These inward–outward recursions are the phylogenetic equivalent of the backward–forward recursions known from hidden Markov models, and other versions of it have already been developed (27, 28). The version that we developed here can be shown to be the realization of the junction tree algorithm (29) on rooted bifurcating trees (see **Note 6**).

3.2.1.3. Computing the Coefficients $w_{gpkk'}$

Here we show that the γ -recursion is sufficient to compute the coefficients $w_{gpkk'}$. From the definition, $w_{gpkk'} = \Pr(\rho_p^\eta = k, \rho_p^0 = k' | \omega_p, \Xi^0, \Psi_g^0, \Lambda^0)$. Using the Bayes formula $\Pr(x, y | z) = \Pr(x, y, z) / \sum_{x,y} \Pr(x, y, z)$, we can rewrite it as

$$\begin{aligned} w_{gpkk'} &= \frac{\Pr(\rho_p^\eta = k, \rho_p^0 = k', \omega_p | \Xi^0, \Psi_g^0, \Lambda^0)}{\sum_{b,b'} \Pr(\rho_p^\eta = b, \rho_p^0 = b', \omega_p | \Xi^0, \Psi_g^0, \Lambda^0)} = \\ &= \frac{\Pr(\rho_p^\eta = k | \Xi^0, \Psi_g^0, \Lambda^0) \cdot \Pr(\rho_p^0 = k' | \Xi^0, \Psi_g^0, \Lambda^0) \cdot \Pr(\omega_p | \Xi^0, \Psi_{gkk'}^0)}{\sum_{b,b'} \Pr(\rho_p^\eta = b | \Xi^0, \Psi_g^0, \Lambda^0) \cdot \Pr(\rho_p^0 = b' | \Xi^0, \Psi_g^0, \Lambda^0) \cdot \Pr(\omega_p | \Xi^0, \Psi_{gbb'}^0)}. \end{aligned}$$

But $\Pr(\rho_p^\eta = k | \Xi^0, \Psi_g^0, \Lambda^0)$ is just the current estimate of the probability of the gain rate variable to have the value r_k^η , namely $(f_k^\eta)^0$. Similarly, $\Pr(\rho_p^0 = k' | \Xi^0, \Psi_g^0, \Lambda^0)$ is just $(f_{k'}^0)^0$. Therefore, the expression for the coefficients $w_{gpkk'}$ is reduced to

$$w_{gpkk'} = \frac{(f_k^\eta)^0 (f_{k'}^0)^0 \Pr(\omega_p | \Xi^0, \Psi_{gkk'}^0)}{\sum_{b,b'} (f_b^\eta)^0 (f_{b'}^0)^0 \Pr(\omega_p | \Xi^0, \Psi_{gbb'}^0)}. \quad [15]$$

The function $\Pr(\omega_p | \Xi^0, \Psi_{gkk'}^0)$ is the likelihood of observing pattern ω_p for gain and loss rate variables r_k^η and $r_{k'}^0$, respectively. This is readily computed upon completion of the γ -recursion, using **Eq. [11]**.

3.2.1.4. Computing the Coefficients $Q_{gpkk'}$

Here we show that these coefficients require the α, β -recursion. By definition,

$$Q_{gpkk'} = \sum_{\sigma} \Pr(\sigma | \omega_p, \Xi^0, \Psi_{gkk'}^0) \cdot [\log f_k^\eta + \log f_{k'}^0 + \log \Pr(\omega_p, \sigma | \Xi, \Psi_{gkk'})].$$

The probability $\Pr(\omega_p, \sigma | \Xi, \Psi_{gkk'})$ is just the likelihood of a particular realization of the tree, thus from **Eq. [1]**

$$\begin{aligned} \log \Pr(\omega_p, \sigma | \Xi, \Psi_{gkk'}) &= \sum_{i=0}^1 \delta(q_0, i) \cdot \log \pi_i \\ &+ \sum_{i,j=0}^1 \sum_{t=1}^{N-1} \delta(q_t, j) \delta(q_t^P, i) \cdot \log A_{ij}(g, t). \end{aligned} \quad [16]$$

Here, $\delta(a, b)$ is the Kronecker delta function, which is 1 for $a = b$ and 0 otherwise. Denote the expectation over $\Pr(\sigma | \omega_p, \Xi^0, \Psi_{gkk'}^0)$ by E_σ . Applying it to **Eq. [16]**, we get

$$\begin{aligned} E_\sigma[\log \Pr(\omega_p, \sigma | \Xi, \Psi_{gkk'})] &= \sum_{i=0}^1 \log \pi_i \cdot E_\sigma[\delta(q_0, i)] \\ &+ \sum_{i,j=0}^1 \sum_{t=1}^{N-1} \log A_{ij}(g, t) \cdot E_\sigma[\delta(q_t, j) \delta(q_t^P, i)]. \end{aligned}$$

But $E_\sigma[\delta(q_0, i)] = \Pr(q_0 = i | \omega_p, \Xi^0, \Psi_{gkk'}^0) = \beta_i(0)$, and similarly $E_\sigma[\delta(q_t, j) \delta(q_t^P, i)] = \alpha_{ij}(t)$. Hence, $Q_{gpkk'}$ is given by

$$\begin{aligned} Q_{gpkk'} &= \sum_{\sigma} \Pr(\sigma | \omega_p, \Xi^0, \Psi_{gkk'}^0) [\log f_k^\eta + \log f_{k'}^\theta + \log \Pr(\omega_p, \sigma | \Xi, \Psi_{gkk'})] = \\ &= \log f_k^\eta + \log f_{k'}^\theta + \sum_{i=0}^1 \beta_i(0) \log \pi_i + \sum_{i,j=0}^1 \sum_{t=1}^{N-1} \alpha_{ij}(t) \log A_{ij}(g, t). \end{aligned} \quad [17]$$

3.2.2. The M-Step

Substituting **Eq. [17]** in **Eq. [8]**, we obtain an explicit form of the function whose maximization guarantees stepping up-hill in the likelihood landscape,

$$\begin{aligned} Q &= \sum_{g=1}^G \sum_{p=1}^{\Omega} \sum_{k=1}^{K_\eta} \sum_{k'=1}^{K_\theta} n_{gp} w_{gpkk'} (\log f_k^\eta + \log f_{k'}^\theta) + \\ &+ \sum_{g=1}^G \sum_{p=1}^{\Omega} \sum_{k=1}^{K_\eta} \sum_{k'=1}^{K_\theta} n_{gp} w_{gpkk'} [\beta_0^{gpkk'}(0) \log \pi_0 + \beta_1^{gpkk'}(0) \log \pi_1] + \\ &+ \sum_{g=1}^G \sum_{p=1}^{\Omega} \sum_{k=1}^{K_\eta} \sum_{k'=1}^{K_\theta} \sum_{t=1}^{N-1} n_{gp} w_{gpkk'} \alpha_{00}^{gpkk'}(t) \log [1 - \xi_t (1 - e^{-\eta_{gk} \Delta t})] + \\ &+ \sum_{g=1}^G \sum_{p=1}^{\Omega} \sum_{k=1}^{K_\eta} \sum_{k'=1}^{K_\theta} \sum_{t=1}^{N-1} n_{gp} w_{gpkk'} \alpha_{01}^{gpkk'}(t) [\log \xi_t + \log (1 - e^{-\eta_{gk} \Delta t})] + \\ &+ \sum_{g=1}^G \sum_{p=1}^{\Omega} \sum_{k=1}^{K_\eta} \sum_{k'=1}^{K_\theta} \sum_{t=1}^{N-1} n_{gp} w_{gpkk'} \alpha_{10}^{gpkk'}(t) \log [1 - (1 - \phi_t) e^{-\theta_{gk'} \Delta t}] + \\ &+ \sum_{g=1}^G \sum_{p=1}^{\Omega} \sum_{k=1}^{K_\eta} \sum_{k'=1}^{K_\theta} \sum_{t=1}^{N-1} n_{gp} w_{gpkk'} \alpha_{11}^{gpkk'}(t) [\log (1 - \phi_t) - \theta_{gk'} \Delta t]. \end{aligned} \quad [18]$$

Actually, any increase in Q is sufficient to guarantee an increase in the likelihood, suggesting that a precise maximization of Q is not very important. Therefore, we speed computations by performing low-tolerance maximization with respect to each of the parameters individually. Except for the parameters λ_η and λ_θ , it is easy to differentiate Q twice with respect to any parameter. This lends itself into using simple zero-finding algorithms; we chose the Newton-Raphson algorithm (30). Maximizing Q with respect to the shape parameters λ_η and λ_θ is more involved, as Q depends on these parameters only through the discrete approximation of the rate variability distributions, **Eq. [3]** (*see Note 7*).

4. Notes



1. If we replace the formal summing over all states of ρ_p^η and ρ_p^θ in **Eq. [6]** by a direct sum, we get

$$\begin{aligned} Q_{gp}(\Xi, \Psi_g, \Lambda, \Xi^0, \Psi_g^0, \Lambda^0) &= \sum_{k=1}^{K_\eta} \sum_{k'=1}^{K_\theta} \sum_{\sigma} \Pr(\sigma, \rho_p^\eta = k, \rho_p^\theta) \\ &= k' | \omega_p, \Xi^0, \Psi_g^0, \Lambda^0) \quad [19] \\ &\log \Pr(\omega_p, \sigma, \rho_p^\eta = k, \rho_p^\theta = k' | \Xi, \Psi_g, \Lambda). \end{aligned}$$

Using our notational conventions, we can write the first term in **Eq. [19]** as

$$\begin{aligned} \Pr(\sigma, \rho_p^\eta = k, \rho_p^\theta = k' | \omega_p, \Xi^0, \Psi_g^0, \Lambda^0) \\ = \Pr(\rho_p^\eta = k, \rho_p^\theta = k' | \omega_p, \Xi^0, \Psi_g^0, \Lambda^0) \quad [20] \\ \cdot \Pr(\sigma | \omega_p, \Xi^0, \Psi_{gk k'}^0), \end{aligned}$$

and the second term as

$$\begin{aligned} \log \Pr(\omega_p, \sigma, \rho_p^\eta = k, \rho_p^\theta = k' | \Xi, \Psi_g, \Lambda) \\ = \log \Pr(\rho_p^\eta = k | \Xi, \Psi_g, \Lambda) + \\ + \log \Pr(\rho_p^\theta = k' | \Xi, \Psi_g, \Lambda) \quad [21] \\ + \log \Pr(\omega_p, \sigma | \Xi, \Psi_{gk k'}) \\ = \log f_k^\eta + \log f_{k'}^\theta + \log \Pr(\omega_p, \sigma | \Xi, \Psi_{gk k'}). \end{aligned}$$

Substituting **Eqs. [20]** and **[21]** back in **Eq. [19]** gives the desired result.

2. We expand

$$\begin{aligned}
 \gamma_i(t) &= \Pr(V_t | q_t^P = i) = \Pr(V_t^L, V_t^R | q_t^P = i) \\
 &= \sum_{j=0}^1 \Pr(V_t^L, V_t^R, q_t = j | q_t^P = i) = \\
 &= \sum_{j=0}^1 \Pr(q_t = j | q_t^P = i) \cdot \Pr(V_t^L | q_t = j, q_t^P = i) \\
 &\quad \cdot \Pr(V_t^R | V_t^L, q_t = j, q_t^P = i).
 \end{aligned} \tag{22}$$

The first term is simply the definition of $A_{ij}(\mathcal{G}, t)$. Given q_t , V_t^L is independent on q_t^P , thus the second term is just $\Pr(V_t^L | q_t = j) = \gamma_j(t^L)$. By similar arguments the third term is just $\Pr(V_t^R | q_t = j) = \gamma_j(t^R)$. By substituting those results in **Eq. [22]**, we recover the recursion formula, **Eq. [10]**.

3. One of the appealing features of this recursion is that it allows to treat missing data fairly easily. Only a single option has to be added to the initialization phase **Eq. [9]**,

$$\gamma(t \in V_0) = \begin{pmatrix} 1 \\ 1 \end{pmatrix} \quad q_t = *.$$

4. To prove this recursion, let us start with the definition of α ,

$$\begin{aligned}
 \alpha_{ij}(t) &= \Pr(q_t = j, q_t^P = i | \omega_p) = \Pr(q_t = j, q_t^P = i | V_0) \\
 &= \Pr(q_t^P = i | V_0) \cdot \Pr(q_t = j | q_t^P = i, V_0) \\
 &= \beta_i(P(t)) \cdot \Pr(q_t = j | q_t^P = i, V_0).
 \end{aligned} \tag{23}$$

Let us make the decomposition $V_0 = V_t + \bar{V}_t$, with \bar{V}_t being the set of all leaves such that node t is not among their ancestors. But, given q_t^P , the state of node t is independent on \bar{V}_t , and therefore **Eq. [23]** becomes

$$\alpha_{ij}(t) = \beta_i(P(t)) \cdot \Pr(q_t = j | q_t^P = i, V_t). \tag{24}$$

From Bayes formula,

$$\begin{aligned}
 \Pr(q_t = j | q_t^P = i, V_t) &= \frac{\Pr(q_t = j, V_t | q_t^P = i)}{\Pr(V_t | q_t^P = i)} \\
 &= \frac{\Pr(q_t = j | q_t^P = i) \cdot \Pr(V_t | q_t = j, q_t^P = i)}{\gamma_i(t)} \tag{25} \\
 &= \frac{A_{ij}(\mathcal{G}, t)}{\gamma_i(t)} \cdot \Pr(V_t | q_t = j, q_t^P = i).
 \end{aligned}$$

But given q_t , V_t is independent of $P(t)$ and therefore

$$\Pr(V_t|q_t = j, q_t^P = i) = \Pr(V_t|q_t = j) = \tilde{\gamma}_j(t). \quad [26]$$

Combining **Eqs. [25]** and **[26]** in **Eq. [24]**, we get

$$\alpha_{ij}(t) = \frac{\tilde{\gamma}_j(t)\beta_i(P(t))}{\gamma_i(t)} A_{ij}(\mathcal{G}, p),$$

which is just another form of writing **Eq. [13]**.

5. When missing data are present, two simple modifications are required. First, we have to add to the initialization phase **Eq. [12]** an option

$$\alpha(D(0)) = \frac{1}{\Pr(\omega_p|\Xi^0, \Psi_{gkk}^0)} \left\{ \begin{array}{cc} \pi_0\gamma_0[\bar{D}(0)]A_{00}[\mathcal{G}, D(0)] & \pi_0\gamma_0[\bar{D}(0)]A_{01}[D(0)] \\ \pi_1\gamma_1[\bar{D}(0)]A_{10}[D(0)] & \pi_1\gamma_1[\bar{D}(0)]A_{11}[D(0)] \end{array} \right\} \quad D(0) \in V_0, q_0^D = *$$

Second, we have to add to the finalization phase **Eq. [14]** an option

$$\alpha(t) = \left\{ \begin{array}{cc} \beta_0[P(t)]A_{00}(\mathcal{G}, t) & \beta_0[P(t)]A_{01}(\mathcal{G}, t) \\ \beta_1[P(t)]A_{10}(\mathcal{G}, t) & \beta_1[P(t)]A_{11}(\mathcal{G}, t) \end{array} \right\} \quad q_t = *.$$

6. The junction tree algorithm is a scheme to compute marginal probabilities of maximal cliques on graphs by means of belief propagation on a modified junction tree. Indeed, the matrix α computes marginal probabilities of pairs $(t, P(t))$, but such pairs are nothing but maximal cliques on rooted bifurcating trees.
7. In our implementation, we used Yang's quantile method (24) to compute the discrete levels of the gamma distributions such that each level has equal probability. Formally, $f_1^\eta = v$, $f_k^\eta = (1 - v)/(K_\eta - 1)$ for $k = 2, \dots, K_\eta$, and $f_k^\theta = 1/K_\theta$ for $k = 1, \dots, K_\theta$. To perform the maximization in this case, we used Brent's maximization algorithm that does not require derivatives (30).

References

1. Nixon JE, Wang A, Morrison HG, McArthur AG, Sogin ML, Loftus BJ, Samuelson J. A spliceosomal intron in *Giardia lamblia*. *Proc Natl Acad Sci U S A* 2002; 99:3359–3361.
2. Vanacova S, Yan W, Carlton JM, Johnson PJ. Spliceosomal introns in the deep-branching eukaryote *Trichomonas vaginalis*. *Proc Natl Acad Sci U S A* 2005; 102:4430–4435.
3. Simpson AG, MacQuarrie EK, Roger AJ. Early origin of canonical introns. *Nature* 2002;419:270.
4. Collins L, Penny D. Complex spliceosomal organization ancestral to extant eukaryotes. *Mol Biol Evol* 2005;22:1053–1066.
5. Lynch M., Richardson AO. The evolution of spliceosomal introns. *Curr Opin Genet Dev* 2002;12:701–710.
6. Roy SW, Gilbert W. The evolution of spliceosomal introns: patterns, puzzles and progress. *Nat Rev Genet* 2006;7:211–221.
7. Rogozin IB, Wolf YI, Sorokin AV, Mirkin BG, Koonin EV. Remarkable interkingdom conservation of intron positions and

- massive, lineage-specific intron loss and gain in eukaryotic evolution. *Curr Biol* 2003;13:1512–1517.
8. Roy SW, Gilbert W. Complex early genes. *Proc Natl Acad Sci U S A* 2005;102:1986–1991.
 9. Roy SW, Gilbert W. Rates of intron loss and gain: implications for early eukaryotic evolution. *Proc Natl Acad Sci U S A* 2005;102:5773–5778.
 10. Csuros M. Likely scenarios of intron evolution, Lecture Notes in Bioinformatics (McLysaght, A. and Huson, D., editors): Proc. RECOMB 2005 Comparative Genomics International Workshop (RCG 2005) 2005;3678:47–60.
 11. Qiu WG, Schisler N, Stoltzfus A. The evolutionary gain of spliceosomal introns: sequence and phase preferences. *Mol Biol Evol* 2004;21:1252–1263.
 12. Fedorov A, Roy SW, Fedorova L, Gilbert W. Mystery of intron gain. *Genome Res* 2003;13:2236–2241.
 13. Cho S, Jin SW, Cohen A, Ellis RE. A phylogeny of caenorhabditis reveals frequent loss of introns during nematode evolution. *Genome Res* 2004;14:1207–1220.
 14. Roy SW, Hartl DL. Very little intron loss/gain in Plasmodium: intron loss/gain mutation rates and intron number. *Genome Res* 2006;16:750–756.
 15. Jeffares DC, Mourier T, Penny D. The biology of intron gain and loss. *Trends Genet* 2006;22:16–22.
 16. Nguyen HD, Yoshihama M, Kenmochi N. New maximum likelihood estimators for eukaryotic intron evolution. *PLoS Comput Biol* 2005;1:e79.
 17. Nei M, Chakraborty R, Fuerst PA. Infinite allele model with varying mutation rate. *Proc Natl Acad Sci U S A* 1976;73:4164–4168.
 18. Uzzell T, Corbin KW. Fitting discrete probability distributions to evolutionary events. *Science* 1971;172:1089–1096.
 19. Dibb NJ. Proto-splice site model of intron origin. *J Theor Biol* 1991;151:405–416.
 20. Dibb NJ, Newman AJ. Evidence that introns arose at proto-splice sites. *Embo J* 1989;8:2015–2021.
 21. Sverdlov AV, Rogozin IB, Babenko VN, Koonin EV. Reconstruction of ancestral protosplice sites. *Curr Biol* 2004;14:1505–1508.
 22. Jordan IM (ed.). *Learning in Graphical Models*. Kluwer Academic Publishers, Boston, MA, 1998.
 23. Jin L, Nei M. Limitations of the evolutionary parsimony method of phylogenetic analysis. *Mol Biol Evol* 1990;7:82–102.
 24. Yang Z. Maximum likelihood phylogenetic estimation from DNA sequences with variable rates over sites: approximate methods. *J Mol Evol* 1994;39:306–314.
 25. Dempster AP, Laird NM, Rubin DB. Maximum likelihood from incomplete data via the EM algorithm. *J R Statist Soc B* 1977;39:1–38.
 26. Felsenstein J. Evolutionary trees from DNA sequences: a maximum likelihood approach. *J Mol Evol* 1981;17:368–376.
 27. Friedman N, Ninio M, Pe’er I, Pupko T. A structural EM algorithm for phylogenetic inference. *J Comput Biol* 2002;9:331–353.
 28. Siepel A, Haussler D. Phylogenetic estimation of context-dependent substitution rates by maximum likelihood. *Mol Biol Evol* 2004;21:468–488.
 29. Castillo E, Gutierrez JM, Hadi AS. *Expert systems and probabilistic network models (Monographs in Computer Science)*. Springer, New York, 1996.
 30. Press WH, Flannery BP, Teukolsky SA, Vetterling WT. *Numerical recipes in C: The art of scientific computing*. Cambridge University Press, New York, 2nd ed., 1992.

Chapter 17

Enzyme Function Prediction with Interpretable Models

Umar Syed and Golan Yona

Abstract

Enzymes play central roles in metabolic pathways, and the prediction of metabolic pathways in newly sequenced genomes usually starts with the assignment of genes to enzymatic reactions. However, genes with similar catalytic activity are not necessarily similar in sequence, and therefore the traditional sequence similarity-based approach often fails to identify the relevant enzymes, thus hindering efforts to map the metabolome of an organism.

Here we study the direct relationship between basic protein properties and their function. Our goal is to develop a new tool for functional prediction (e.g., prediction of Enzyme Commission number), which can be used to complement and support other techniques based on sequence or structure information. In order to define this mapping we collected a set of 453 features and properties that characterize proteins and are believed to be related to structural and functional aspects of proteins. We introduce a **mixture model of stochastic decision trees** to learn the set of potentially complex relationships between features and function. To study these correlations, trees are created and tested on the Pfam classification of proteins, which is based on sequence, and the EC classification, which is based on enzymatic function. The model is very effective in learning highly diverged protein families or families that are not defined on the basis of sequence. The resulting tree structures highlight the properties that are strongly correlated with structural and functional aspects of protein families, and can be used to suggest a concise definition of a protein family.

Key words: Sequence–function relationships, functional prediction, decision trees, enzyme classification.

1. Introduction

To understand why predicting the Enzyme Commission number (EC number) of an enzyme is important, consider the related problems of pathway prediction and of filling “pathway holes”. With recent advances in sequencing technologies, the number of

genomes that are completely sequenced is increasing steadily, opening new challenges to researchers. One of the main challenges is to decipher the intricate network of cellular pathways that exist in each genome. Understanding this network is the key to understanding the functional role of individual genes and the genetic variation across organisms with respect to key processes and mechanisms that are essential to sustain life. Of special interest are metabolic pathways that consist mostly of enzymatic reactions and are responsible for nucleotide and amino acid synthesis and degradation, energy metabolism, and other functions. Many metabolic pathways have been studied and documented in the literature, usually in specific model organisms such as yeast or *Escherichia coli*.

Since experimental verification of pathways is a time-consuming and expensive process, there is great interest in computational methods that can extend existing knowledge about pathways to newly sequenced genomes. It is often the case that the same metabolic pathway exists in multiple organisms. The different pathways are “homologous” to each other in the sense that they have the same overall structure. In each organism, different enzymes fill the various roles in the pathway. At the same time, different enzymes from different organisms that appear in the same location in a pathway have the same or similar functions. Hence, a pathway can be viewed as a generic graph diagram, where the nodes of the graph are the locations in the pathway, and are annotated with just the function associated with that location. For example, this is the way pathways are represented in the pathway databases KEGG (1) and MetaCyc (2), where EC numbers are used to denote the function of each node in the graph (see Fig. 17.1).

The most popular approach for *pathway prediction* is to use pathway diagrams that were experimentally determined in one organism and map genes from a newly sequenced genome onto these diagrams (3, 4). This approach requires a functional association of genes (and their protein products) with enzymatic reactions or enzyme families. If, for a particular organism, the enzyme corresponding to a node in the pathway diagram is unknown, this is called a “*pathway hole*” for that organism. Only proteins from that organism with the proper enzymatic activity can be candidates for filling a hole.

To identify the set of candidate enzymes various approaches have been employed. For example, Green and Karp (5) used BLAST to identify the best enzyme from among all candidate genes. Specifically, for each pathway hole, all proteins in the genome of interest were BLAST-ed against a set of “query” proteins, which consisted of proteins with the correct EC number taken from other genomes. Those proteins most similar to the query set were viewed as the most promising candidates. Chen and Vitkup (6) fill pathway holes by choosing an enzyme for each hole which

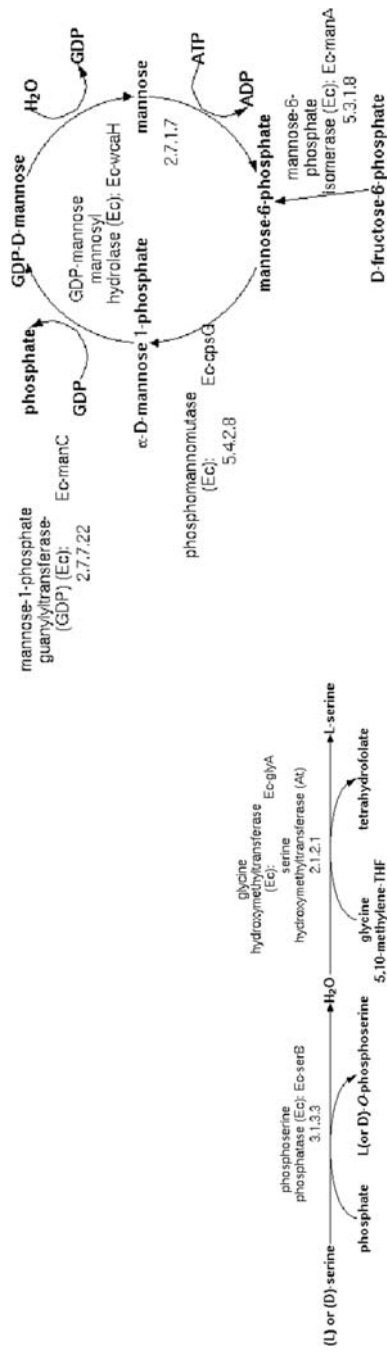


Fig. 17.1. **Left: Glycine biosynthesis pathway. Right: 1/GDP-mannose metabolism.** (Pathway layouts were retrieved from the MetaCyc database (2)). Metabolic pathways are sequences of consecutive enzymatic reactions. Each reaction starts with a certain metabolite and produces another metabolite through synthesis, break down, or other transformations. The Glycine biosynthesis pathway produces the amino acid glycine from the amino acid serine. The GDP-mannose metabolism is one of the pathways that produce the compounds necessary to build cellular structures and organelles, such as cell walls.

has a similar phylogenetic profile as nearby enzymes in the pathway. Kharchenko et al. (7) took an integrative approach to filling holes. They used phylogenetic profiles, expression profiles, physical interaction data, gene fusion data, and chromosomal clustering data to compare each candidate with the hole's neighbors in the pathway. A different integrative approach is used in (8) where genes were selectively assigned to pathways so as to maximize the co-expression of genes that were assigned to the same pathway while minimizing conflicts and shared assignments across pathways. This work was later extended in (9) where deterministic assignments were replaced with probabilistic assignments that reflect the affinity of the different enzymes with different cellular processes.

All these methods require a set of candidates for each hole. And while they can use the trivial candidate set consisting of all of the proteins in the subject organism, their performance is likely to significantly improve if the candidate sets are small and focused. For example, Yaminishi et al. (10) found in experiments with their pathway inference algorithm that constraining pathways so that they are consistent with EC annotations "improves the [performance] in all cases." Similar results were reported in (9). Hence, accurate EC annotation of proteins is an important step toward pathway prediction and functional annotation of genes in general. Producing effective mappings from genes to enzyme families is the focus of this chapter.

1.1. Enzyme Class Prediction: Survey

1.1.1. The EC Hierarchy

Traditionally, enzymes have been organized into a systematic, hierarchical classification scheme devised by the IUB Enzyme Commission (11). The EC number of an enzyme has the form A.B.C.D, where the first digit A indicates the enzyme's major functional family: oxidoreductase, transferase, hydrolase, lyasase, isomerase, or ligase. The second digit B places the enzyme in a functional subfamily, the third into a sub-subfamily, and so on. For example, the EC number 1.2.3.4 designates enzymes which are oxidoreductase (the first digit) that act on the aldehyde or oxo group of donors (the second digit) and have oxygen as an acceptor (the third digit). In this case, the last digit specifies the particular reaction that is catalyzed by this family of enzymes. Some enzymes whose functions are poorly understood have incomplete EC numbers, i.e., they have only been assigned the first few digits of an EC number. Also, some multifunction enzymes have more than one EC number.

1.1.2. Prediction Based on Sequence Similarity

Given a new protein sequence, the first step toward predicting its function is usually through sequence analysis. The most basic form of sequence analysis is sequence comparison, in search of sequence similarity. This analysis is still the most common way of functional prediction today, and the majority of sequences in the protein databases are annotated using just sequence comparison.

However, in many cases sequences have diverged to the extent that their common ancestry cannot be detected even with the most powerful sequence comparison algorithms. This is true for many enzyme families; although the proteins that share a particular EC number all possess the same function, they can nevertheless have very different sequences. As a result, predicting the EC number of a protein, even the first few digits, can be very challenging, and using only sequence alignment techniques is often insufficient.

For example, Shah and Hunter (12) used BLAST and FASTA to examine how well sequence alignment can predict the EC number of an enzyme. They argued that if sequence alignment was enough to completely characterize an EC family, then every enzyme should have a higher alignment score with members of its own family than with members of other families. However, they found that this is the case for only 40% of EC families; for some families, the connection between sequence similarity and functional similarity was very weak. They noted several reasons for this, including the prevalence of multi-domain proteins among enzymes, as well as convergent and divergent evolution.

1.1.3. Structure-Based Approaches

When the structure of a protein is known, sequence analysis can be followed by structure comparison with the known structures in the PDB database. Since structure is more conserved than sequence, structural similarity can suggest functional similarity even when sequences have diverged beyond detection. However, enzyme families do not necessarily correlate with groups of structurally similar proteins. Moreover, proteins can have similar functions even if their structures differ (13, 14).

For example, Todd et al. (15) showed that the same enzyme family can be mapped to different structural folds. A clear mapping in the opposite direction also cannot be established. The authors showed that 25% of CATH superfamilies have members of different enzyme types and above 30% sequence identity is necessary to deduce (with 90% accuracy) the first three digits of the EC number. Similar results are reported in (16), based on pairs of enzymes from FSSP (17), a database of structural alignments. The authors found that among structurally similar proteins, above 50% sequence identity usually implied that the two enzymes shared the first three digits of their EC number, and above 80% sequence identity was enough to preserve all four digits. However, they also found that it was difficult to classify enzymes correctly below 50% identity, while below 30% was “problematic.”

Wilson et al. (18) performed a similar analysis of the SCOP database (19). Among enzymes sharing a particular SCOP fold, they found that the first three digits of the EC number were not preserved below 40% identity, and even the first digit was not preserved below 25% sequence identity.

While these results were not very promising, Rost (20) argued that they were actually too optimistic since the distribution of enzymes in curated databases is not representative of the true distribution of enzymes in nature. He proposed to correct for this bias by reducing the redundancy in those databases, and after doing so found that even a very low BLAST e-value between two proteins did not always imply that they had identical EC numbers. He did show, however, that structural similarity predicts EC number much better than sequence similarity, though still not especially well. But since structural data are sparse, and are not available for newly sequenced genes, structure-based methods are not very useful for genome-wide enzyme prediction.

1.1.4. Approaches Based on Alternate Representation of Proteins

Instead of traditional sequence or structural alignment techniques, several groups have proposed comparison algorithms that rely on an alternative representation of proteins. These representations are usually derived from sequence motifs (i.e., short patterns), simple physiochemical properties that can easily be computed from sequence, or annotations that are independent of sequence but can be found in databases for many proteins, e.g., subcellular location.

For example, desJardins et al. (21) represented each protein as a feature vector where the features were the composition of amino acids and the composition of secondary structure elements like helix, loop, etc. Note that this representation completely ignores the order of the amino acids in the sequence. However, a large number of generic machine learning algorithms require that training data instances be described as feature vectors, so the major advantage of this type of representation is that it allows these algorithms to be straightforwardly applied. DesJardins et al experimented with the naive Bayes classifier, decision trees, and nearest neighbor approaches. Although their results were inferior to sequence alignment methods, they were surprisingly good considering the simplicity of the features they used.

It is also possible to extract features from the molecular 3D structure of proteins. These can include per-residue properties like the contact energy or the distance from the center of mass of the protein. Obviously, this can only be applied to proteins whose structure is already known. Recently, Borro et al. (22) applied a naive Bayes algorithm to this kind of representation, and were able to predict the first digit of the EC number with 45% accuracy.

Cai and Chou (23) used a feature vector consisting of the amino acid composition, as well as the correlation of amino acid properties between residues near each other (but not necessarily adjacent) on the sequence. Their feature vectors also incorporated the GO (24) annotations associated with the motifs contained in the sequence. In order to focus their efforts on difficult cases, they restricted their data set to proteins that had less than 20% sequence

identity to each other. Still, they were able to predict the first digit of the EC number of an enzyme with 85% accuracy. However, as there are only six possible values for the first EC digit, their method can uncover only a broad indication of the function of the enzyme.

A related approach is described in (25). This work uses an elaborate algorithm for predicting protein function, which integrates several types of data. Given a query protein, they computed several sequence-derived and database-derived properties for it. They also performed a PSI-BLAST (26) search to find homologous proteins, and computed the same properties for those homologs as well. Next, they used inductive logic programming to find frequently occurring patterns among all these properties, and let each pattern be a binary feature. Finally, they used those features to train decision trees. There are several similarities between their algorithm and ours, but a major difference is that their decision trees are deterministic, whereas ours are stochastic (as described in **Section 4.1**). Although they did not attempt to predict EC numbers, they were able to infer annotations from the MIPS classification scheme (27), with accuracies ranging from 38% to 75%.

An important subgenre within these methods consists of those that use Support Vector Machines (SVMs) (28). Alternate protein representations often reside in very high-dimensional spaces, and SVMs are particularly well suited for learning in these kinds of spaces, provided an appropriate kernel can be devised. Jaakola et al. (29) introduced the idea of using SVMs in biosequence analysis. They focused on detecting remote protein homologs, and devised SVM classifiers for SCOP families using an HMM-based kernel, where the feature vectors were generated by computing the derivatives of the sequence's likelihood with respect to the HMM parameters. Their study was followed by many others who applied SVMs to various classification tasks, including EC prediction. For example, Han et al. (30) formed a feature vector from several basic physio-chemical properties of proteins, and used a Gaussian kernel in the resulting feature space. They applied their algorithm to a set of 50 enzymes that have no known sequence homologs (the set was constructed using PSI-BLAST). On this difficult set, they were able to predict the first two digits of the EC number with 72% accuracy.

A novel kernel function, the "mismatch" kernel, was introduced in (31) and tested on the SCOP classification. They defined their feature space such that the inner product of two vectors in the space is large only when the two sequences corresponding to those vectors contain many of the same k -mers (i.e., amino acid sequences of length k). Ben-Hur and Brutlag (32) explored a very similar approach as (31), but used sequence motifs instead of k -mers in their mismatch kernel. They tried to predict EC numbers and reported better results than simpler methods such as the nearest neighbor algorithm.

1.1.5. Other Approaches

Some authors have proposed novel data sources to predict EC numbers. For example, in prokaryotic organisms, it is well known that functionally related genes are often grouped together on the genome. Kolesov et al. (33) extended this concept with the following hypothesis: if the orthologs of two genes tend to be near each other on many genomes, then the two genes themselves are likely to have a similar function.

Other ideas can be viewed as advanced sequence similarity techniques. For instance, Tian et al. (34) used HMMs to determine the “functionally discriminating residues” of each EC family. According to their definition, these are the minimal set of residues needed to distinguish members of an EC family from non-members. Levy et al. (35) proposed a Bayesian method to combine BLAST scores for the purpose of EC classification. They extend the simple approach of assigning each enzyme to the same EC class as its BLAST best hit. Instead, for each enzyme, they estimate its membership probability in each EC class based on the EC classes of other enzymes with similar BLAST best-hit scores.

1.2. Our Approach

In view of the previous sections, it is clear that multiple aspects of protein similarity should be considered, beyond sequence and structure, when functional similarity is sought. Features such as the domain content, subcellular location, tissue specificity, pairwise interactions, and expression profiles may indicate common context and may suggest functional similarity even in the absence of clear sequence or structure similarity. These protein attributes and others are usually ignored, either because data are not available or because it is not clear how to use them to quantify functional similarity.

In this chapter we describe a method to predict the function of a protein based on basic biochemical properties augmented with (partial) data available from database records of biological properties that may hint at the biological context of the protein. Our goal is to identify the most relevant and informative features and combination of features that best characterize protein families (e.g. enzyme families) and to create a model for each protein family, which can be used for classification and function prediction. Our approach hinges on algorithms for decision tree building (36, 37, 38), but further expands the traditional work on decision trees, by introducing a **mixture model of stochastic decision trees (SDT)**. The trees handle missing attributes, ambiguities, and fuzzyness, which are to be expected when analyzing database records of proteins by using a probabilistic framework and are optimized to ensure high generalization power by testing several validation techniques.

The resulting tree structure indicates the properties that are strongly correlated with the class of proteins modeled. This set of properties depends on the classification task. For example, the set of properties most strongly correlated with structural aspects are

not necessarily the same as the properties that are most relevant for functional aspects. Clearly, this set can also change from one protein family to another. To study these correlations, trees were created and tested on two different types of known protein classifications: a sequence-based classification of proteins (Pfam (39)) and a functional classification of proteins (the enzyme classification system (11)).

The learning system consists of two main parts: the first is the feature extraction system. Much thought was given to selecting an extensive set of attributes that would create reliable representations of protein families. The second part is the model used to identify regularities in the patterns of these attributes for each family – the set of attributes that best correlate with functional aspects of proteins and can most accurately predict membership in the family. We first turn to describing the feature extraction system and then to the decision tree learning model and our new SDT model.

2. Methods I – Data Preparation and Feature Extraction

Our basic data set is the SWISSPROT database release 39.3 and the TrEMBL database (40) release 14.4 as of July 15, 2000, with 464744 proteins (the dataset is available at <http://biozon.org/ftp/data/papers/ec/>). A total of 453 features are used to describe each protein. The features are divided into three sets of features: features that can be calculated directly from the sequence, features that are predicted from the sequence, and features that are extracted from database records.

2.1. Sequence Features

These features include composition percentages for the 20 individual amino acids, as well as for 16 amino acid groups adopted from (41) and (42). The groups are as follows: charged (DEHIKLRV), positively charged (HKR), negatively charged (DE), polar (DEHKNQRSTWY), aliphatic (ILV), aromatic (FHWY), small (AGST), tiny (AG), bulky (FHRWY), hydrophobic (ILMV), hydrophobic aromatic (FWY), neutral and weakly hydrophobic (AGPST), hydrophilic acidic (EDNQ), hydrophilic basic (KRH), acidic (ED), and polar and uncharged (NQ).

Despite the overlap between these amino acid groups, each one has a unique characteristic. It is this characteristic that may play a role in defining and distinguishing family members from non-members, and therefore all the groups were considered to be distinct features.

We also calculated composition percentages for each of the 400 possible dipeptides. However, to save computation time, only the most informative dipeptides were used during learning. The

selection of dipeptides was done dynamically, during the training phase of the algorithm, on a per-family basis. We discuss this refinement further in **Note 1a**.

In addition to sequence composition, we also computed the average hydrophobicity, average isoelectric point, and average molecular weight of the protein sequence, as well as the overall sequence length. The hydrophobicities used are the scaled values, as described in (43). The values for isoelectric point can be found at (44).

2.2. Predicted Features

These represent the percentage of each protein that is predicted to be a coil, helix, or strand, as computed by PSIPRED (45), a secondary structure prediction program. We ran PSIPRED in “single” mode for each sequence, meaning that the program did not search for sequence homologs to assist in its prediction.

2.3. Database Features

We extracted nine database features from SWISSPROT: three binary, five nominal, and one numerical (integer). The three binary features respectively indicate the presence of alternative products, enzyme cofactors, and catalytic activity in each protein, as determined from the optional fields of the CC section of each SWISSPROT protein record (see SWISSPROT user manual (46)). A lack of annotation was interpreted as a “0” value; otherwise, the value was set to “1”.

The nominal features (tissue specificity, subcellular location, organism classification, and species) are more involved. Each protein is defined over zero or more values for each nominal feature: if $Values(A)$ is the set of all possible values for nominal attribute A , then each protein is defined over a subset of $Values(A)$. In all cases, a complete lack of information for A in a protein record was interpreted as an “unknown” value for that attribute, and was treated specially by the decision tree algorithm (described in the **Note 1e**). Two definitions of tissue specificity were included, one derived from the RC:TISSUE field and one from the CC:TISSUE SPECIFICITY field. The subcellular location of the protein was derived from the CC:SUBCELLULAR LOCATION field, while the organism classification and species were taken from the OC and OS fields, respectively.

Lastly, we also computed the number of PROSITE patterns exhibited by each protein by consulting the appropriate DR line of each record. Specifically, for each protein, we summed the number of certain and uncertain hits over all PROSITE patterns. This attribute is an indication of the domain/motif “complexity” of the protein. (Alternatively, one could use an attribute for each possible motif, or a single motif attribute that indicates the set of motifs that is present in the protein. However, with more than 1300 PROSITE patterns, the set of possible values or combinations of values is too large to handle efficiently during learning.)

It should be noted that we intentionally skipped the keywords in the KW field of SWISSPROT records. These keywords are strongly linked with protein function. However, as opposed to the other database attributes, these keywords explicitly describe protein functionality and are based on human annotation and interpretation. If ours was purely a classification task, the performance clearly could have improved by integrating SWISSPROT keywords. However, since we are interested in learning the mapping between proteins' properties and their function, we decided to exclude these keywords. We selected only the database attributes that are based on experimentally verified information.

3. Methods II - The Decision Tree Learning Model

To model protein families, we developed a novel learning algorithm, which we call a **mixture model of stochastic decision trees (SDT)**. Each family is modeled by a collection of decision trees that capture the salient features which are common to the members of the family. Our choice of the model was motivated by the nature of the data. Decision trees are useful when the data are nominal, i.e., when there is no natural notion of similarity or even ordering between objects. Such is the case for some of the attributes we associate with proteins, for example, tissue specificity and subcellular location. These attributes rule out the use of many other popular machine learning models, such as Support Vector Machines. Other attractive aspects of decision trees are robustness to errors both in classification and in attribute values. Moreover, decision trees can be used when some of the data are missing, and this is often the case with database attributes. In the next sections, we first describe the traditional decision tree model, and then introduce the mixture model and other variations. The reader who is not interested in the details of the learning model can skip to **Section 6**. However, we recommend reading **Sections 3.1** through **5** to get an idea of the main components of the model.

3.1. The Basic Decision Tree Model

Decision trees classify instances by sorting them down the tree from the root to a leaf node, which provides the classification of the instance. Each leaf node is either associated with a category label or probabilities over different categories. Each internal node specifies a test of some attribute of the instance and each branch descending from that node corresponds to a subset or range of the possible values of this attribute. An example decision tree is given in **Fig. 17.2**. An instance is classified by testing the attributes of the instance, one at a time, starting with the one defined by the root

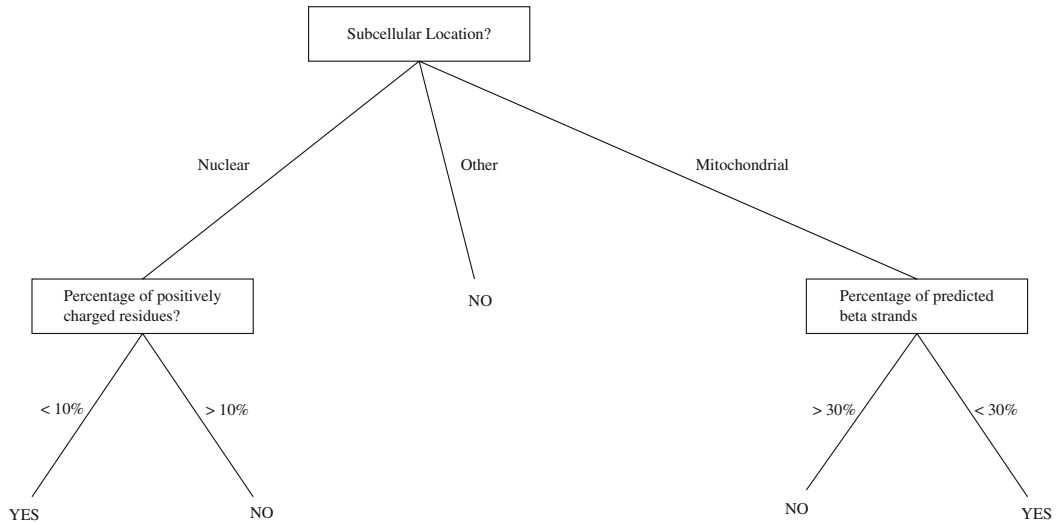


Fig. 17.2. **A sample decision tree for a specific class of proteins.** An instance is classified by testing its attributes, starting from the test at the root node. If the final answer is yes, then the instance is classified to the class.

node, and moving down the tree along the branch corresponding to the specific value of the attribute in that instance, until it reaches a leaf node. Note that this kind of scenario does not require a metric over instances.

3.2. The Traditional Training Procedure for Decision Trees

When training decision trees from sample data S , the goal is to create a concise model that is consistent with the training data. Most learning algorithms use variations on a core algorithm that employs a top-down, greedy search through the space of all possible decision trees. Trees are grown from the root to the leaves, and the algorithm progressively splits the set of training examples into smaller and purer subsets by introducing additional nodes to the bottom of the growing tree with tests over new properties.

Usually, the criterion according to which the next attribute is selected is based on the reduction in *impurity* (class mixup) when testing on the attribute: the difference between the impurity before and after the split. One possible measure of the impurity or uncertainty is the entropy of the sample set at that node. The entropy impurity is also called the information impurity and is defined as

$$i(S) = Entropy(S) \equiv - \sum_{j=1}^c p_j \log_2 p_j$$

where p_j is the fraction of examples in the set S , which are in category j . The common strategy is to select the attribute that decreases the impurity the most, where the drop in impurity is defined as

$$\Delta i(S, A) = i(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} i(S_v)$$

where $Values(A)$ is the set of all possible values for attribute A , and S_v is the subset of samples in S that have value v for attribute A . When the measure used is the entropy impurity, the drop in impurity is called information gain and is denoted by $Gain(S, A)$

$$Gain(S, A) = Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

The ID3 learning algorithm (47) uses this criterion to decide on which attribute to test next. The outline of the algorithm is given below:

1. Initialize a root node with the set of all samples S .
2. If no node reduces the node impurity
 - a) Then node is a leaf.
3. Else
 - a) Select best test for decision node.
 - b) Partition instances by test outcome.
 - c) Construct one branch for each possible outcome.
 - d) Build subtrees recursively.

After the learning procedure has terminated, the learned tree can be used to classify new instances, as depicted in **Fig. 17.2**. During classification, an example is repeatedly subjected to tests at decision nodes until it reaches a leaf, where it is assigned a probability equal to the percent of training examples at the leaf that are class members.

3.3. The Extended Training Procedure for Decision Trees

The ID3 algorithm may result in trees that over-fit the training data, capturing coincidental regularities that are not necessarily characteristic of the class being modeled. Such trees will perform poorly on new unseen examples. The C4.5 algorithm (48) is an improvement over ID3 by using rule **post-pruning** to prevent overfitting and several other modifications to handle missing attributes and to account for bias in information gain introduced by multi-value attributes (using the *GainRatio* and *Mantaras* functions described in **Note 1f**).

Our basic learning procedure (which is one component of our new learning system) is similar to the C4.5 algorithm. We make intensive use of validation-based post-pruning in our algorithm. We refer to the set of samples available for learning as the **learning set** (the remaining samples are compiled into the **test set** used later for performance evaluation). The proteins in the learning set are divided into 10 equal parts. Let n be the size of the learning set. Ten different trees are generated for each family, and each one uses $\frac{9}{10}n$ as its **training set** and $\frac{1}{10}n$ as its **validation set**. Each tree is built using the training set and pruned using the validation set. Pruning is done by running an exhaustive search over all tree

nodes. Each node is temporarily pruned (i.e., converted to a leaf) and the performance of the new tree is evaluated over the validation set. The node that produces the largest increase in performance is selected and the corresponding subtree is pruned. The process continues as long as it does not cause a decrease in performance over the validation set (we evaluate performance using a variety of different *evaluation functions*, which are discussed in **Section 4.2**).

Cross-validation pruning improves consistency and provides a more balanced model, which learns the regularities that are common to different subsets of the data rather than those that are coincidental and occur only in a specific data set.

Eventually, the ten trees are combined to obtain a single prediction. Each tree is weighted according to its performance over the learning set, where the performance is measured in terms of the evaluation function, denoted by Q (*see Section 4.2*). The final output is a performance-weighted average of the probabilities returned by each of the ten trees. Denote by $P_i(+|x)$ the class probability of the sample x according to the i th tree, then the total probability assigned by the set of trees is given by

$$P_{\text{Trees}}(+|x) = \frac{\sum_{i \in \text{Trees}} Q(i) P_i(+|x)}{\sum_{i \in \text{Trees}} Q(i)} \quad [1]$$

where $Q(i)$ is the quality (performance) of the i th tree over the learning set (the higher, the better the tree separates the samples). It should be noted that the ten trees that are learned from the ten sets are generally very similar to each other, and their similar core structure is a reflection of the properties that truly characterize the protein family. However, because of the greedy nature of the search for the optimal tree, other competing tree structures may be missed. This issue is addressed in the next section where we introduce the concept of a stochastic decision tree.

In addition to cross-validation based post-pruning, our algorithm includes other variations and enhancements that were adopted from different papers in this field, as well as several novel elements that we introduced into the learning procedure. Among the enhancements are

- Dynamic attribute filtering (*see Note 1a*).
- An effective procedure for discretizing numerical features (*see Note 1b*).
- An efficient algorithm for finding binary partitions of multi-valued attributes (*see Note 1c*).
- Methods to handle instances that have missing or multiple values for an attribute (*see Notes 1d and 1e*).
- Various impurity measures for attribute selection (*see Note 1f*).

- Different weighting functions and adjusted impurity measures (such as **mixed entropy** and **interlaced entropy**) to handle skewed distributions in which negative samples far outnumber positive samples, such as in our protein classification problem (*see* **Notes 1g** and **1h**).

For more details on these and other elements of the extended learning algorithm, the interested reader is referred to **Note 1**.

4. Methods III – The Mixture of Stochastic Decision Trees

The next elements of our model address some of the fundamental problems with traditional decision tree learning algorithms. Specifically, we address three elements: optimization, evaluation, and model selection. More precisely, we first propose an effective method of searching the hypothesis space that overcomes the pitfalls of the deterministic learning algorithms. Secondly, we introduce some novel criterion functions to evaluate decision tree performance. Thirdly, we propose an alternative method (MDL-based pruning) for deciding on the most probable model that is especially effective for small data sets. The first two elements are described next. The third is described in **Note 2**.

4.1. The Stochastic Framework

The basic procedures for building decision trees (such as those that are described in **Section 3**) are deterministic. They employ a greedy approach that always selects the attribute which maximizes the information gain. However, this local maximization is not guaranteed to produce the “best” decision tree that describes the data. It happens quite often that several attributes have very similar information gain values. The choice of the one that marginally outperforms the others at some point in the training process may prove to be less advantageous later on. Even if the chosen attribute has a significantly better information gain, it still does not guarantee that in the optimal tree this attribute is indeed used at this point. To address this limitation imposed by the classical decision tree learning algorithm, we switched to a stochastic (probabilistic) framework in which the attribute is selected with probability that depends on its information gain.

$$P, (A) = \frac{Gain(S, A)}{\sum_i Gain(S, A_i)}$$

Thus, attributes with higher information gain have higher probability to be selected, but even attributes with small information gain have a non-zero probability to get selected. To diversify the

composite model of a protein family, a total of ten trees are learned for each training set; one deterministic (using the traditional approach) and nine stochastic trees. With ten training sets per family (see Section 3.3), the final mixture model has a total of 100 trees per family. As before, denote by $P_i(+ | x)$ the class probability of the sample x according to the i th tree, then the total probability assigned by the hybrid mixture of trees is given by

$$P_{\text{Mixture}}(+ | x) = \frac{\sum_{\text{training set } j} \sum_{i \in \text{Trees}(j)} Q(i) P_i(+ | x)}{\sum_{\text{training set } j} \sum_{i \in \text{Trees}(j)} Q(i)}, \quad [2]$$

where $\text{Trees}(j)$ is the set of trees learned from the j th training set.

Our model was originally introduced in (49). It is similar to other methods that use ensembles of decision trees, such as (50) and (51). In particular, Breiman (52) introduced the *random forests* model in which a large mixture of decision trees is trained and where the best split at each node is chosen from among a randomly selected subset of all the attributes. The parallels with our method are clear, but a major difference is that our method biases the selection toward higher quality attributes, instead of randomly constraining the selection to a small set. Moreover, some of the trees in our mixture are trained using the traditional deterministic method. Breiman's analysis suggests that these changes may yield improved performance, since he proves that the generalization error of a random forest decreases as the quality of the individual trees in the forest increase.

4.2. Evaluation of Decision Trees

The evaluation function Q is a key element of our learning model (see Eq. [1] and [2]) as well as for performance assessment. A common measure of performance that is used in many machine learning applications (including decision tree learning algorithms such as C4.5) is the **accuracy**, i.e., the percentage of correctly classified examples

$$\text{accuracy} = \frac{tp + tn}{tp + tn + fp + fn} = \frac{tp + tn}{\text{total}}$$

where tp is the number of true positives, tn is the number of true negatives, fp is the number of false positives, and fn is the number of false negatives. However, with the majority of the samples being negative examples, the accuracy may not be a good indicator of the discriminating power of the model. When the task is to discern the members of a specific class (family) from a large collection of negative examples, better measures of performance are the **sensitivity** and **selectivity** (the positive predictive power), defined as

$$\text{sensitivity} = \frac{tp}{(tp + fn)} \quad \text{selectivity} = \frac{tp}{(tp + fp)}$$

The categorization of samples as true (false) positives/negatives depends on the model and on the output it assigns to the samples. Since decision trees output probabilities, we need a way to interpret these real numbers as either “positive” or “negative”. Usually, one sets a threshold score T , a probability above which samples are predicted to be positive. Thus, if a sample reaches a leaf node j with membership probability $P_j(+)$ (defined based on the relative fraction of positive samples in this node), then it will be classified as positive if $P_j(+)$ > T (see **Note 3** for discussion on methods for setting the threshold).

The ROC measure. An alternative that does not require defining a threshold first is the Receiver Operating Characteristic (ROC) score. To compute this score, one first has to sort all samples according to their probabilities (as assigned by the model), then plot the number of true positives as a function of the number of false positives, and finally measure the area under the curve. This measure will be maximized when all the true positives are assigned higher scores than the negative samples. A variation on this score is ROC50, which only measures the area under the curve up to the first 50 false positives. The idea behind this plot is that in scanning database search results one may be willing to overlook a few errors if additional meaningful similarities can be detected. The area under the curve can be used to compare the overall performance of different methods.

The Jensen-Shannon measure. We propose a new evaluation function that takes into account the complete distributions of positives and negatives induced by the decision tree model, and accounts for their *statistical distance*. Specifically, we use the Jensen-Shannon (JS) divergence between probability distributions (53). Given two probability distributions p and q , for every $0 \leq \lambda \leq 1$, the λ -JS divergence is defined as

$$D_{\lambda}^{JS}[p||q] = \lambda D^{KL}[p||r] + (1 - \lambda) D^{KL}[q||r]$$

where $D^{KL}[p||q]$ is the relative entropy of p with respect to q (also called the KullbackLeibler divergence (54)) and

$$r = \lambda p + (1 - \lambda)q$$

can be considered as the most likely common source distribution of both distributions p and q , with λ as a prior weight (here we set $\lambda = 1/2$ since the weighted samples are divided equally between the two categories; see **Note 1g**). In our case, p and q are the empirical distributions of the positive and negative examples with respect to the class probabilities assigned to them by the decision tree model. We call the corresponding measure simply the **divergence score** and denote it by D^{JS} . This measure is symmetric and ranges between 0 and 1, where the divergence for identical distributions is 0.

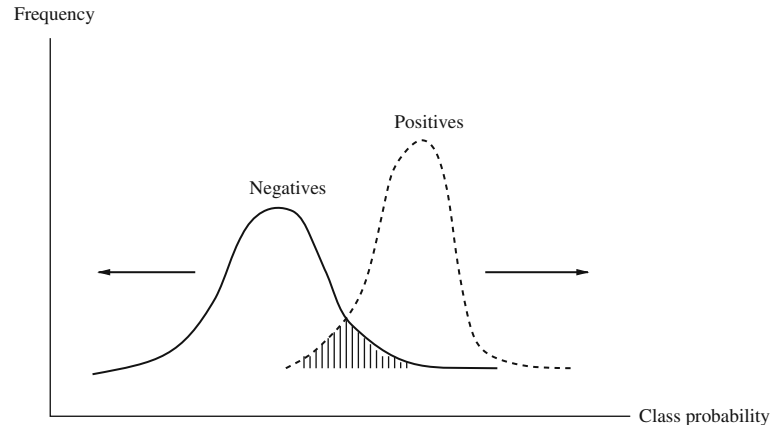


Fig. 17.3. **Optimizing performance with the Jensen-Shannon measure.** Ideally, the model should assign high probability to class members and low probability to non-class members. The goal is to maximize the statistical distance between the distribution of positives and the distribution of negatives and minimize the overlap.

All the measures provide rough estimates of the expected accuracy. However, unlike the other measures, the divergence score is less sensitive to outliers and is more likely to reflect the true estimate when applied to future examples. Moreover, there is no need to define a threshold T . Our goal is to maximize the statistical distance between the distribution of positives and the distribution of negatives, as exemplified in **Fig. 17.3**, and minimize the overlap. The smaller the overlap, the better the expected accuracy.

All four performance measures were tested during post-pruning (a node is pruned if the probabilities induced by the pruned tree are better at separating the positives from the negatives in terms of the evaluation function selected; see **post-pruning** in **Section 3.3** for more details). Also, while testing a particular pruning technique, we always used that same evaluation function to compute $Q(i)$ in Eq. [1] and [2].

5. Methods IV - Optimization of the Learning Algorithm

There are many strategic decisions that one can take throughout the learning process. To find a good choice of parameters, we first optimized the learning strategy. Having incorporated the modifications described in **Sections 3.3** and **4** into our model and the tree-generating program, we converged to a locally optimal learning strategy by conducting a rough greedy search over the **space of decision tree learning algorithms**. All performance evaluations

are done over the test set using the Pfam data set (*see Section 6.1*) and averaged over all families. A step-by-step description of that search procedure is as follows.

The initial basic configuration is very similar to the C4.5 learning algorithm (48). It uses a multiple (variable) branching factor, unweighted entropy-based gain ratio, five cross-validation sets, accuracy-based post-pruning (with the equivalence-point-based threshold as described in **Note 3**), unweighted leaf nodes in prediction, and a smaller subset of 53 features after removing all dipeptide information. The performance of this configuration was poor (sensitivity of 0.35). Introducing dipeptide information and increasing the number of cross validation sets to ten resulted in a significant improvement in performance (0.55). This setup was gradually improved by considering each one of the modifications discussed in the previous sections. The modifications were added one by one, and were accepted only if they improved performance over the previous configuration of the parameters. Each configuration includes only modifications that were previously accepted during the optimization procedure. However, not all previously accepted modifications are included in each configuration, since some of them override each other. This is the case, for instance, with the various pruning strategies, since only one pruning strategy is used at a time. The results of the optimization procedure are summarized in **Table 17.1**.

Switching to binary splitting, weighted entropy and JS-based post-pruning, and the introduction of the stochastic decision trees improved performance and were accepted. Especially noticeable is the improvement due to the replacement of a single deterministic tree with a mixture of stochastic trees. Increasing the number of cross-validation sets did improve the performance, but only until a certain point (ten sets) beyond which no further improvement was observed. Some of the weighting procedures (such as mixed-entropy and leaf weighting) were rejected, as well as other modifications that were later outperformed by alternative strategies (e.g., sensitivity/selectivity based pruning was selected first during the search but was then outperformed by Jensen-Shannon based pruning). It should be noted that very good performance was also obtained with MDL-based pruning (*see Note 2*). The MDL approach is especially effective for small families (*see Section 6.1*). However, because of its dependency on an external parameter that requires additional tuning we focus here on the simpler Jensen-Shannon-based pruning. The final average performance, after adding the stochastic trees, was the same with Jensen-Shannon and MDL-based pruning (in both cases it was 0.81).

The final configuration was set to the mixture of stochastic decision trees, including information on dipeptides (dynamically selected on a per-family basis, during training as described in

Table 17.1

Optimization of learning strategy. We started with a basic learning algorithm that is very similar to the C4.5 learning algorithm. Then, one step at a time, we introduced another modification to the learning algorithm and tested its impact on the performance. Performance was measured in terms of the prediction sensitivity over the test set using the equivalence point criterion (see Note 3). At the equivalence point, the sensitivity equals the selectivity

Step	Modification	Sensitivity (=selectivity)	Accepted/rejected
1	Initial configuration (C4.5)	0.35	Initial
2	Dipeptides + 10 cross-validation sets	0.55	Accepted
3	Mantaras metric	0.56	Accepted
4	Binary splitting	0.66	Accepted
5	Weighted entropy	0.69	Accepted
6	Confidence-based threshold	0.63	Rejected
7	Sen/sel post-pruning	0.69	Rejected
8	JS-based post-pruning	0.7	Accepted
9	Roc50 post-pruning	0.7	Rejected
10	20 cross-validation sets	0.69	Rejected
11	MDL post-pruning (entropy based)	0.71	Accepted
12	MDL post-pruning (probability based)	0.73	Accepted
13	Weighted leafs	0.72	Rejected
14	Mixed entropy	0.67	Rejected
15	Interlaced entropy	0.73	Rejected
16	Stochastic trees	0.81	Accepted

Note 1a), with binary splitting, weighted entropy, ten sets of cross-validation, and JS-based pruning and evaluation. The probability assigned by the mixture model to a sample x is defined as

$$P_{\text{Mixture}}(+/x) = \frac{\sum_{\text{training set } j} \sum_{i \in \text{Trees}(j)} Q_{JS}(i) P_i(+/x)}{\sum_{\text{training set } j} \sum_{i \in \text{Trees}(j)} Q_{JS}(i)}$$

where $Q_{JS}(i)$ is the divergence/separation score of the i th tree over the validation set. Note that the divergence score does not require a method to choose a threshold.

6. Results

We used two types of known protein classifications, the Pfam database of protein families and the EC database of enzyme families. We tested our model over both databases by training models for all families using the optimal learning strategy, as described in the previous section, and testing them over unseen examples.

6.1. The Pfam Classification Test

Our first test was on the Pfam database of protein domains and families. Of the 464744 proteins in the SWISSPROT and TrEMBL databases, 237925 proteins are classified into 2128 families (Pfam release 5.2). Most of the families in the Pfam database are domain families that characterize only a small part of the member proteins (these domains usually appear in conjunction with other domains). To prevent ambiguity in the class labels, we included in our data set only those proteins that were labeled as belonging to exactly one family. In particular, since sequence databases are constantly growing, the absence of a label does not necessarily mean that the protein does not belong to a functional family. Since the properties we use in our model are global (calculated based on the complete sequence), only protein families that cover most of the sequence were considered for learning and evaluation. Specifically, we used a subset of 233 families with more than 80% coverage of the sequence and at least 50 members. Because of the high coverage, we expect that for these families their functionality is indeed associated with the complete protein chain.

Each family was randomly partitioned into a learning set and a test set in the ratio of 3:1, so that 75% of the member proteins were used for learning and 25% for testing, and a model was trained for each family using the optimal learning strategy as described in **Section 5**. The mixture model was then used to assign probabilities to each sample in the test set and the performance was evaluated by measuring sensitivity at the equivalence point (i.e., the point where the selectivity equals the sensitivity; *see Note 3* for details).

In all cases, the mixture model performed better than the best individual tree. Interestingly, for almost half the families, the best stochastic tree performed better (by up to 10%) than the greedy, deterministic tree, which is evidence for the usefulness of the stochastic approach. In **Fig. 17.4** we show five trees for the Pfam

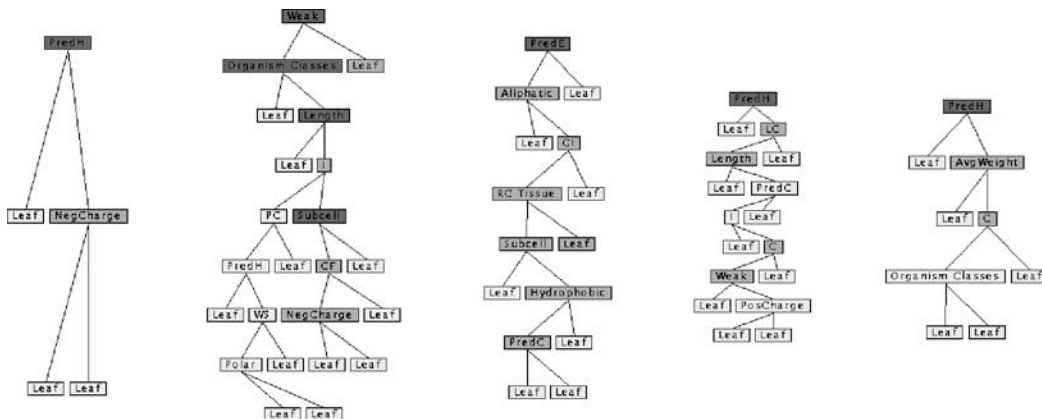


Fig. 17.4. **Alternative decision trees for Pfam family Apolipoprotein.** The leftmost tree is a deterministic tree. The others are stochastic. The rightmost tree performed the best. The shading of each decision node is proportional to the (weighted) entropy at that node. Darker shades indicate higher entropy values (high class impurity).

family Apolipoprotein (proteins whose function is to carry cholesterol and other fats through the blood and help in maintaining normal levels of these molecules). The first one is deterministic while the other trees were built using the probabilistic algorithm. The performance of the five trees is 0.96, 0.85, 0.84, 0.89, and 0.99, with the fifth stochastic tree outperforming the deterministic tree.

To assess the power of our new model we compared it with BLAST (26), the most popular database search algorithm. Since BLAST is a pairwise comparison algorithm, we repeated the search for all query sequences in the family. Specifically, for each query, we sorted all proteins by their BLAST score against the query, and then measured sensitivity for detecting the family at the equivalence point (i.e., the threshold where the selectivity equals the sensitivity; *see Note 3*). For each family, we report the best performing query, as well as the performance of a typical BLAST search, by averaging over all queries. **Table 17.2** gives the results for the 30 largest families. The average performance of the SDT model over all families was 81%. A typical BLAST search detected 86% of the member proteins, while the best BLAST search detected 94% of the proteins on average (in a blind prediction scenario one does not have a knowledge of which is the best query, as this is determined based on the performance over the test set, and therefore the average performance should be considered as the typical performance). Note that for many families, especially the larger ones, the SDT model outperformed a typical BLAST query, and in some cases the best BLAST query (e.g., picornavirus coat proteins [rhv], Hemagglutinin-neuraminidase [HN] and histones). This is very encouraging considering that our model does not use any information about the order of amino acids in the sequences beyond pair statistics.

Our analysis suggests three main reasons why our model is short of detecting all member proteins for some of these sequence-based protein families. First, since Pfam is not necessarily a functional classification but rather a domain classification, many of the features that we associate with complete protein chains may not be correlated with the specific domain. It is possible to “localize” some features and calculate them just along the domain region. However, some features, such as database attributes, cannot be localized.

The second reason is the use of weighted entropy. Although weighted entropy definitely improves performance (*see Table 17.1*), it can also stop the learning too early, leaving some of the leaf nodes impure and thus affecting performance (*see Fig. 17.5*). Specifically, because of the skewed background distributions, positives are assigned a much higher weight than negatives. Take for example the UbiA family. There are 58 family members and 237867 negative examples, of which 43 and 178400 are in the learning set, respectively. The weight of each one of the positive examples is more than 4000 times the weight of a negative example. Consider a node with 20 positive examples and 1000 negative

Table 17.2
SDT performance for the 30 largest Pfam families. Performance is evaluated using the equivalence point criterion (see text). For BLAST two numbers are given, for the best query and for a typical query (in parenthesis). The average performance of the SDT model over all 233 families was 0.81

Family name	Family size	Test set size	Percent test set detected by	
			SDT	BLAST
GP120	21644	5411	1.00	–
COX1	3397	850	0.97	1.00 (0.73)
MHC_II_beta	2348	587	0.97	1.00 (0.97)
F-protein	1637	410	0.99	0.99 (0.96)
Hemagglutinin	1360	340	0.99	1.00 (0.97)
p450	1328	332	0.78	0.97 (0.86)
globin	1064	266	0.89	0.87 (0.66)
adh_short	1012	253	0.63	0.97 (0.79)
rhv	895	224	0.95	0.84 (0.58)
vMSA	890	223	1.00	1.00 (0.91)
ras	793	199	0.86	0.99 (0.98)
NADHdh	786	197	0.94	0.99 (0.76)
Tat	707	177	0.99	1.00 (0.99)
adh_zinc	695	174	0.76	0.99 (0.84)
VPR	645	162	0.99	1.00 (0.98)
sugar_tr	624	156	0.65	0.91 (0.61)
Vif	604	151	0.99	1.00 (1.00)
fer4_NifH	561	141	0.85	0.96 (0.84)
actin	555	139	0.90	1.00 (0.97)
histone	536	134	0.95	0.54 (0.35)
cpn60_TCP1	516	129	0.85	0.95 (0.84)
HSP70	511	128	0.89	1.00 (0.99)
Gram-ve_porins	495	124	0.96	0.94 (0.82)
late_protein_L1	482	121	0.93	1.00 (0.76)
COX3	480	120	0.95	1.00 (0.94)
fusion_gly	442	111	0.99	1.00 (0.83)
HN	414	104	0.95	0.82 (0.51)
REV	375	94	0.95	0.97 (0.87)
serpin	353	89	0.78	0.95 (0.86)
COesterase	347	87	0.79	0.99 (0.92)

examples. If the total weight of the initial set of positives and negatives is 100 (50 each), then the sum of the weighted positive examples in that node is 26 while the negative samples sum to 0.3, resulting in weighted entropy of 0.09. However, this node is far from being pure, and although we set a very low impurity threshold some nodes do end being somewhat impure. In the case of the UbiA family, one of the nodes has an entropy of 0.15535 with 0.27 weighted negative examples and 11.8 weighted positive examples. However, the number of actual negatives and positives is 876 and 9, respectively. This node clearly needs refinement. Employing other weighting protocols can resolve this problem (see Note 1g). Indeed, switching to unweighted sample entropy after exhausting weighted sample entropy (interlaced entropy) improved performance over the 25 worst performing families by several percent.

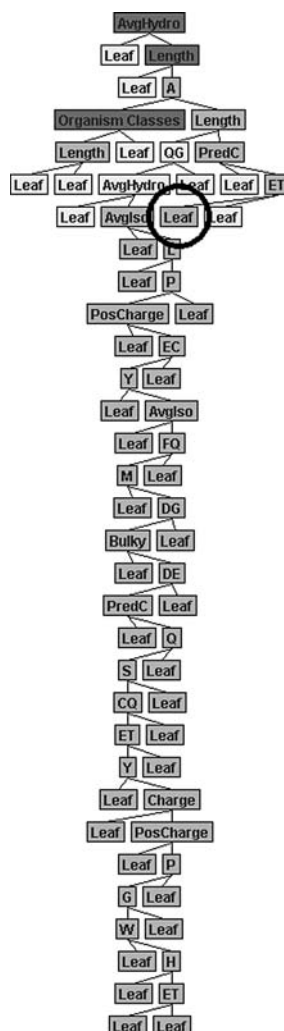


Fig. 17.5. **An example tree for the UbiA family.** Our model did not perform well over this family. The main reason is demonstrated in this tree. One of the leaf nodes (*circled*) has an entropy of 0.15535 with 2727.95 weighted negative examples and 118421 weighted positive examples. However, the number of actual negatives and positives is 876 and 9, respectively.

But perhaps a more significant flaw is that validation-based pruning is overly aggressive for small families. This is because, for these families, the validation set typically contains only a handful of family members. It is unlikely that such a small sample will be representative of the family overall. So during pruning, many branches of the tree (which was trained on a much larger sample) will be deemed irrelevant and pruned away. For these families, MDL-based pruning (*see Note 2*) may be a better alternative, as it seeks to balance tree size with the performance of the tree over the entire learning set. We tried MDL-based pruning on a subset

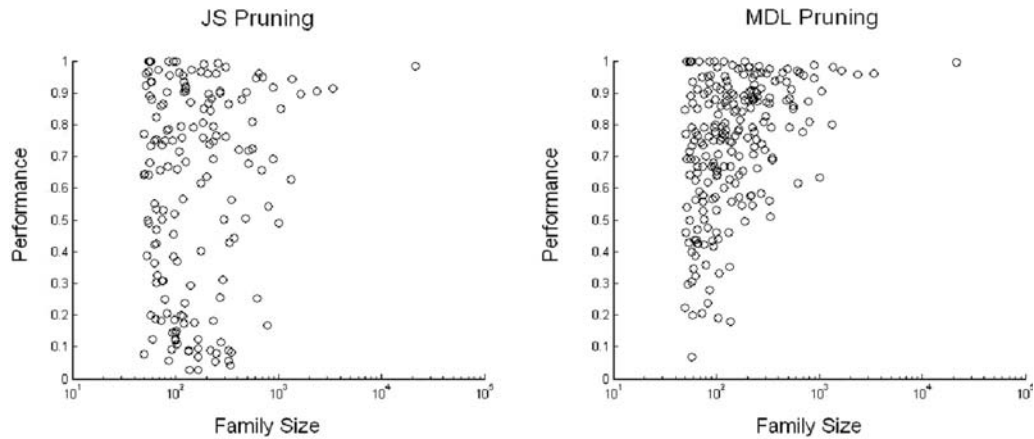


Fig. 17.6. **Improvement in performance with MDL-based pruning.** The *left plot* shows performance (on the y-axis) vs. family size (on the x-axis) under JS-based pruning. Each *circle* represents a family. The *right plot* shows the improvement from MDL-pruning. Note that the mass of families clustered at the *bottom-left* of the JS plot shifted upward in the MDL plot, while the rest of the distribution seems relatively unchanged.

of 50 small families. Compared to JS pruning, the lowest performing families seem to make the biggest gains, with performance boosting over this subset from an average of 0.34 to 0.47 (Fig. 17.6).

6.2. The Information Content of Features

In our representation, each protein sequence is represented by a set of 453 attributes. However, not all of the attributes are equally important and not all of them are necessarily used in the model for each protein family. Those that are used are not exclusive in that they allow even proteins with unknown values for these attributes to be classified to the family. This is because the learning system can accommodate ambiguities by defining multiple rules for each protein family during the learning process.

To evaluate the quality of each feature as a protein family predictor, and to quantify its information content, we trained 453 trees for each family, one for each feature. The tree was trained using the optimal configuration as described in the previous section. (A simple way to estimate the information content of an attribute would be to calculate the class entropy for each feature, and average over all families. However, this calculation ignores some of the aspects of decision trees that are addressed in the Notes.) The results were averaged over all families and are given in Table 17.3. Results for just the dipeptide attributes are available in Table 17.4.

Clearly, no single attribute can serve as a good predictor in general. Performance varies between the different attributes and database attributes seem to have less information content

Table 17.3

Attribute information – part 1 (excludes dipeptides). Attributes are sorted by decreasing information gain with respect to that attribute only (second column). The results are averaged over all families. Since throughout the learning phase we use the weighted entropy (*see Note 1g*) the information gain in this case is given over the weighted samples. These numbers are also more effective for identifying informative attributes; because the skewed distribution of positives and negatives, the background unweighted entropy is very small to begin with and significant improvement in prediction power will translate only to slight improvements in the unweighted entropy (the non-linearity of the entropy function further obscures significant changes in classification accuracy). The third column is the average depth in the trees at which the attribute is used. The depth of the root is 0; the average depth of the bottom of the trees is 16.83

Attribute	Information gain	Depth
OrganismClasses	0.390	5.6
AvgHydro	0.289	6.4
Length	0.259	6.3
NegCharge	0.253	6.5
Acid	0.253	6.7
Aromatic	0.245	6.9
HydrophobicArom	0.238	7.0
HydrophilicAcid	0.225	7.0
AvgWeight	0.223	6.9
Bulky	0.210	7.2
PosCharge	0.209	6.9
HydrophilicBase	0.209	6.9
E	0.209	7.2
Charge	0.207	6.8
Hydrophobic	0.202	6.9
PredH	0.200	6.7
AvgIso	0.196	7.2
Polar	0.191	6.7
W	0.190	7.4
R	0.190	7.4
Aliphatic	0.187	6.8
PrositeDomains	0.185	7.5
PredC	0.184	6.9
D	0.182	7.4
Y	0.179	7.6
Tiny	0.176	7.0
L	0.172	7.4
G	0.168	7.3

(continued)

Table 17.3 (continued)

K	0.166	7.5
C	0.163	7.4
PredE	0.162	7.0
Weak	0.161	7.2
Small	0.160	7.3
F	0.155	7.5
Proline	0.147	7.4
P	0.147	7.4
A	0.142	7.3
V	0.141	7.6
T	0.135	7.7
PolarUn	0.127	7.5
N	0.121	7.7
H	0.121	7.8
I	0.110	7.7
S	0.099	7.9
Species	0.098	6.7
Q	0.098	7.9
M	0.098	7.9
Catalytic	0.089	8.4
Cofactor	0.039	9.7
Subcell	0.035	6.8
RCTissue	0.011	7.5
Alternate	0.006	10.7
TissueSpec	0.002	6.9

than the predicted and calculated attributes, perhaps because little information is currently available in database records. For most proteins these attributes are undefined and fractions of these proteins reach many leaf nodes (following the ratios for proteins with known values for these attributes, as described in **Note 1e**), thus affecting the discriminative power of the attribute.

Surprisingly, the organism classification attribute is the best predictor (information gain of 0.390). Note also its depth (5.6), meaning that it is usually used higher in the decision trees. Indeed many protein families exist only in a single kingdom or subphyla. The success of this attribute suggests that other database attributes can play a more significant role, as more data become available. The organism class is followed by the average hydrophobicity and the length. Other informative attributes are the relative compositions of different groups of amino acids, such as acidic, negatively charged, aromatic, and so on.

We also recorded the usage frequency of attributes. The usage frequency of an attribute (**Table 17.5**) is an indication of its role in classification at a more general level. Such attributes are the length, the organism class, predicted secondary structure, and the frequency of different amino acid groups.

Table 17.4
Attribute information – part 2 (dipeptides only). Missing depth information means that the dipeptide was never used in the mixture of stochastic decision trees. Only the top 40 dipeptide attributes are listed

Attribute	Information gain	Depth
DP	0.192	7.9
GG	0.180	6.6
YG	0.171	4.8
EL	0.167	5.5
LD	0.166	—
GA	0.166	5.4
DG	0.164	8.4
GL	0.163	6.6
PL	0.162	—
IG	0.161	5.9
GR	0.160	6.9
LY	0.157	—
AG	0.157	6.9
LG	0.156	4.0
FL	0.155	7.1
FF	0.154	8.0
RR	0.149	6.6
EV	0.149	—
EG	0.149	9.2
EE	0.149	6.6
WL	0.148	7.7
QL	0.148	5.9
DV	0.148	—
VD	0.147	—
NG	0.147	3.8
LE	0.147	6.1
KK	0.147	7.0
EA	0.147	7.0
AV	0.147	—
NV	0.146	5.4
GK	0.146	7.9
DL	0.145	—
VR	0.144	5.2
IL	0.144	—
AI	0.144	7.1
AA	0.144	6.4
MP	0.143	7.1
GT	0.143	5.6
NP	0.142	8.6
LR	0.142	6.6

6.3. The EC Classification Test

While the Pfam test establishes the validity and proves the potential of our method, it is the EC classification test (which was independent of the optimization of the learning strategy) that shows the true prediction power in cases where sequence-based methods can fail.

The EC classification system is an extensive collection of functionally based protein families. Enzyme families are defined based on the functional role of the enzymes in the cell rather than on common evolutionary ancestry, and enzymes that perform similar functions are not necessarily homologs and might not exhibit any sequence or structure similarity. Clearly, detecting this kind of similarity is much more difficult, as it involves inferring the functionality of the proteins in vivo.

Table 17.5

Attribute usage. Percentage of *all splits* in the mixture of stochastic decision trees that use each attribute. The third column lists the percentage of *all trees* that used the attribute, and the fourth is the ratio between the two. A higher ratio of the percentage of splits to the percentage of trees indicates multiple tests over the same attribute in the same tree (usually because of a range of values or multiple values where the information about family association is not limited to a single transition point)

Attribute	Usage frequency (splits)	Usage frequency (trees)	Ratio
OrganismClasses	0.030	0.686	0.044
Length	0.033	0.646	0.051
PredH	0.023	0.547	0.042
AvgHydro	0.023	0.544	0.042
PredE	0.022	0.527	0.042
PredC	0.020	0.507	0.039
AvgIso	0.019	0.494	0.038
Hydrophobic	0.019	0.490	0.039
AvgWeight	0.019	0.490	0.039
Aromatic	0.019	0.489	0.039
Polar	0.019	0.478	0.040
HydrophobicArom	0.018	0.478	0.038
Charge	0.018	0.474	0.038
G	0.018	0.473	0.038
Aliphatic	0.018	0.467	0.039
NegCharge	0.017	0.462	0.037
L	0.017	0.462	0.037
Tiny	0.017	0.454	0.037
PosCharge	0.016	0.449	0.036
C	0.017	0.449	0.038
R	0.016	0.448	0.036
PrositeDomains	0.016	0.442	0.036
K	0.016	0.438	0.037
HydrophilicAcid	0.016	0.436	0.037
P	0.016	0.435	0.037
A	0.016	0.434	0.037
Acid	0.016	0.431	0.037
W	0.015	0.430	0.035
Small	0.016	0.429	0.037
Weak	0.015	0.423	0.035
E	0.015	0.422	0.036
HydrophilicBase	0.016	0.420	0.038
V	0.015	0.416	0.036
F	0.015	0.413	0.036
Y	0.015	0.412	0.036
I	0.015	0.411	0.036
D	0.015	0.409	0.037
Proline	0.015	0.406	0.037
S	0.014	0.404	0.035
PolarUn	0.014	0.404	0.035
T	0.014	0.401	0.035
H	0.014	0.399	0.035
N	0.014	0.393	0.036
Q	0.013	0.385	0.034
M	0.013	0.371	0.035
Species	0.009	0.291	0.030
Catalytic	0.007	0.256	0.027
Bulky	0.008	0.208	0.038
Cofactor	0.004	0.140	0.029
Subcell	0.002	0.075	0.027
RCTissue	0.001	0.057	0.018
Alternate	0.001	0.024	0.042
TissueSpec	0.0005	0.016	0.031

To test our model on this functional classification we extracted all enzyme families and trained models for all families with more than 50 members, for a total 229 families. In our training and test sets, we only included proteins that belonged to exactly one EC family. Models were trained as described in **Section 6.1**. As with the Pfam classification test, we compared our results to BLAST. We also compared our model with another statistical model, the HMM-based alignment program SAM (55). Each SAM model was trained first over the set of unaligned training sequences and then used to search the database and assign probabilities to each sample in the test set. The performance was evaluated as before using the equivalence point criterion.

The average performance of the SDT model over all 229 families was 71%. Many of these were “high-level” EC families (e.g., 1.-.-.-), so we compared our performance against other methods over just the 122 families with complete, four-digit EC numbers (e.g., 1.2.3.4). (This data set is available at <http://biozon.org/ftp/data/papers/ec/>.) SAM performance over this subset was 89%. The performance of our model over the same subset was 76%. The results for the 30 largest families in this subset are listed in **Table 17.6**. Note that in this case the SDT model outperformed the best BLAST in many cases, especially when the enzyme family is composed of several sequence-based families. As most members of the same EC family have similar sequences, the success of this model is not surprising. However, for extremely diverse and large families, our model was more effective in detecting family members (*see* 1.9.3.1, 3.6.1.34, and 2.7.7.48). Again, one has to remember that the SDT model was trained on features rather based on the sequences, and yet it was surprisingly successful in classification. Integrating this model with other models can be expected to boost the overall performance by detecting remote homologies. For example, the mixtures of trees from the SDT model might be used to refine sequence similarity-based models, combining the strengths of both approaches. One can also expect integration of higher-order significant sequence elements (conserved k -tuples) to improve performance as well. Indeed, k -tuples were successfully used to classify proteins into families using support vector machines with a string kernel (31).

It should be noted that for the EC classification task we used the same final configuration of the learning system which was optimized over the Pfam data set (*see* **Table 17.1**). By optimizing the parameters of our learning system over a separate classification, we avoided the danger of overfitting the learning strategy to the irregularities of the EC families.

6.4. Interpretability of Decision Trees – Decision Rules

A decision tree represents a disjunction of conjunctions of constraints on the attribute values of instances, where each path (each decision) from the tree root to a leaf corresponds to one such

Table 17.6

SDT performance for the 30 largest EC families that have complete EC numbers. Performance is evaluated using the equivalence point criterion (see text). For BLAST two numbers are given, for the best query and for a typical query (in parenthesis). The fifth column is the number of different sequence subfamilies in each family (Chau and Yona, unpublished work). The last column shows the performance of SAM. The average performance of the SDT model over all 122 EC families that have complete EC numbers was 0.76, while the performance of SAM over the same set was 0.89

Family number	Family description	Family size	Test set size	Number of subfamilies	Percent test set Detected by		
					SDT	BLAST	SAM
4.1.1.39	Ribulose-bisphosphate carboxylase	3879	970	2	0.99	0.95 (0.91)	0.99
1.9.3.1	Cytochrome-c oxidase	1904	476	18	0.95	0.85 (0.47)	0.92
3.6.1.34	H ⁺ -transporting two-sector ATPase	1612	403	26	0.87	0.42 (0.23)	0.69
2.7.7.48	RNA-directed RNA polymerase	488	122	21	0.95	0.37 (0.14)	0.78
2.7.7.6	DNA-directed RNA polymerase	473	119	26	0.64	0.42 (0.18)	0.76
2.7.7.7	DNA-directed RNA polymerase	461	116	13	0.67	0.45 (0.24)	0.90
3.1.3.48	Protein-tyrosine-phosphatase	395	99	3	0.85	0.83 (0.59)	0.97
1.14.14.1	Unspecific monooxygenase	382	96	1	0.76	0.70 (0.58)	0.77
2.7.1.112	Protein-tyrosine kinase	367	92	2	0.79	0.92 (0.82)	0.89
1.1.1.1	Alcohol denhydrogenase	360	90	2	0.91	0.77 (0.61)	0.92
5.99.1.3	DNA topoisomerase (ATP-hydrolyzing)	320	80	2	0.81	0.92 (0.66)	0.92
1.2.1.12	Glyceraldehyde-3-phosphate dehydrogenase	318	80	1	0.88	0.98 (0.94)	0.98
1.6.99.3	NADH dehydrogenase	315	79	37	0.68	0.40 (0.18)	0.69
2.7.7.49	RNA-directed DNA polymerase	284	71	1	0.77	0.80 (0.57)	0.82
5.2.1.8	Peptidylprolyl isomerase	245	62	2	0.76	0.72 (0.55)	0.95
2.5.1.18	Glutathione transferase	240	60	5	0.77	0.77 (0.49)	0.91
3.2.1.14	Chitinase	222	56	2	0.73	0.51 (0.31)	0.90
6.3.1.2	Glutamyl-ammonia ligase	213	54	1	0.87	0.97 (0.86)	0.98
3.1.3.16	Phosphoprotein phosphatase	211	53	9	0.77	0.67 (0.46)	0.95
3.2.1.4	Cellulase	207	52	2	0.40	0.43 (0.23)	0.89
1.11.1.7	Peroxidase	179	45	5	0.76	0.82 (0.66)	0.83
3.1.1.4	Phospholipase A ₂	175	44	3	0.93	0.91 (0.82)	0.94
1.11.1.6	Catalase	170	43	3	0.91	0.92 (0.82)	0.99
3.2.1.1	α-Amylase	168	42	2	0.79	0.76 (0.57)	0.85
1.18.6.1	Nitrogenase	168	42	2	0.74	0.57 (0.42)	0.92
3.4.99.46	Proteasome endopeptidase complex	161	41	1	0.88	0.98 (0.73)	1.00
2.7.2.3	Phosphoglycerate kinase	157	40	2	0.80	0.97 (0.93)	0.95
3.1.26.4	Calf thymus ribonuclease H	154	39	2	0.69	0.50(0.33)	0.32
4.1.1.31	Phosphoenolpyruvate carboxylase	150	38	2	0.92	0.99(0.85)	0.96
3.5.2.6	β-Lactamase	150	38	3	0.74	0.73(0.52)	0.96

conjunction. Thus, a decision tree can be converted to an equivalent rule set that can be considered a logical description of a category. Each one of our decision trees was converted into a rule set, with rules sorted by their accuracy over the test set. Some of these rules are especially interesting as they might suggest a concise description of a family in terms of a few elements or principles. For example, one of the features that characterizes EC family 1.14.14.1 of monooxygenases (including cytochrome P450) is subcellular location. Many members of this family are located in the membrane-bound endoplasmic reticulum (see Fig. 17.7). Note that after the initial split according to subcellular location, other features refine the definition of family members. One rule given by this tree indicates that proteins located in the membrane-bound endoplasmic reticulum, which have above-

average content of F residues, and are approximately 500 residues long, have a 75% probability of belonging to family 1.14.14.1 (i.e., selectivity = 0.75), with 52% coverage of the family (i.e., sensitivity = 0.52). For comparison, the probability to select a member of this family by chance (the background probability) is 0.01

Interestingly, one of the stochastic trees highlights another feature that characterizes the family: members are more abundant in several tissues (blastocyst, mycelium, prostate, liver, adrenal, lung, hepatopancreas, blood, and ovary) while rare in others. The split by this attribute reduces the impurity by 30%. However, this is just one element of a more complex set of dependencies, and to understand the role of tissue specificity in characterizing family members it is necessary to look at the complete set of rules.

Another example where subcellular location is useful for prediction is family 1.10.2.2 (including the ubiquinol-cytochrome-c reductases). Family members tend to be located in the inner mitochondrial membrane, particularly the matrix side, and half of the deterministic trees in the mixture for the family use this subcellular location in the root split of the tree. Yet another example is enzyme family 2.7.7.6 (RNA polymerases). It turns out that, among proteins that have low content of A, G, W, and E residues, and have fewer than 1000 residues, proteins in the nucleus are disproportionately likely to be members of 2.7.7.6 (selectivity = 0.12, compared to background probability of 0.013, and sensitivity = 0.11). Tissue specificity plays a role in defining the trees of the enzyme family 1.6.99.3 (NADH dehydrogenases), as proteins in the thorax muscle disproportionately belong to the family (selectivity = 0.21, compared to background probability of 0.004, and sensitivity = 0.77). Twenty percent of deterministic root splits in the mixture for 1.6.99.3 are based on tissue specificity.

Some EC families are naturally annotated as having co-factors. For example, members of the 1.18.99.1 hydrogenases often have iron-sulfur and nickel as cofactors. These co-factors in themselves are not enough to distinguish family members from non-members (in fact, a search in Biozon (56) reveals that there are more than 30 different enzyme families that use Nickel as co-factor), but inclusion of this information in decision rules can greatly increase their accuracy. For example, proteins that have iron-sulfur as co-factors have high CT dipeptide content, are found in bacteria or viruses, and are significantly enriched for membership in family 1.18.99.1 (*see Fig. 17.8*). Another example is the co-factor thiamine pyrophosphate. There are 30 enzyme families from four different (out of six) major EC groups that use thiamine pyrophosphate as a co-factor (source: Biozon (56)). However, when this fact is combined with information about amino acid and Prosite domain content, it yields an effective classifier for family 1.2.4.1 (ROC50 score of 0.65 vs. 0.51 for

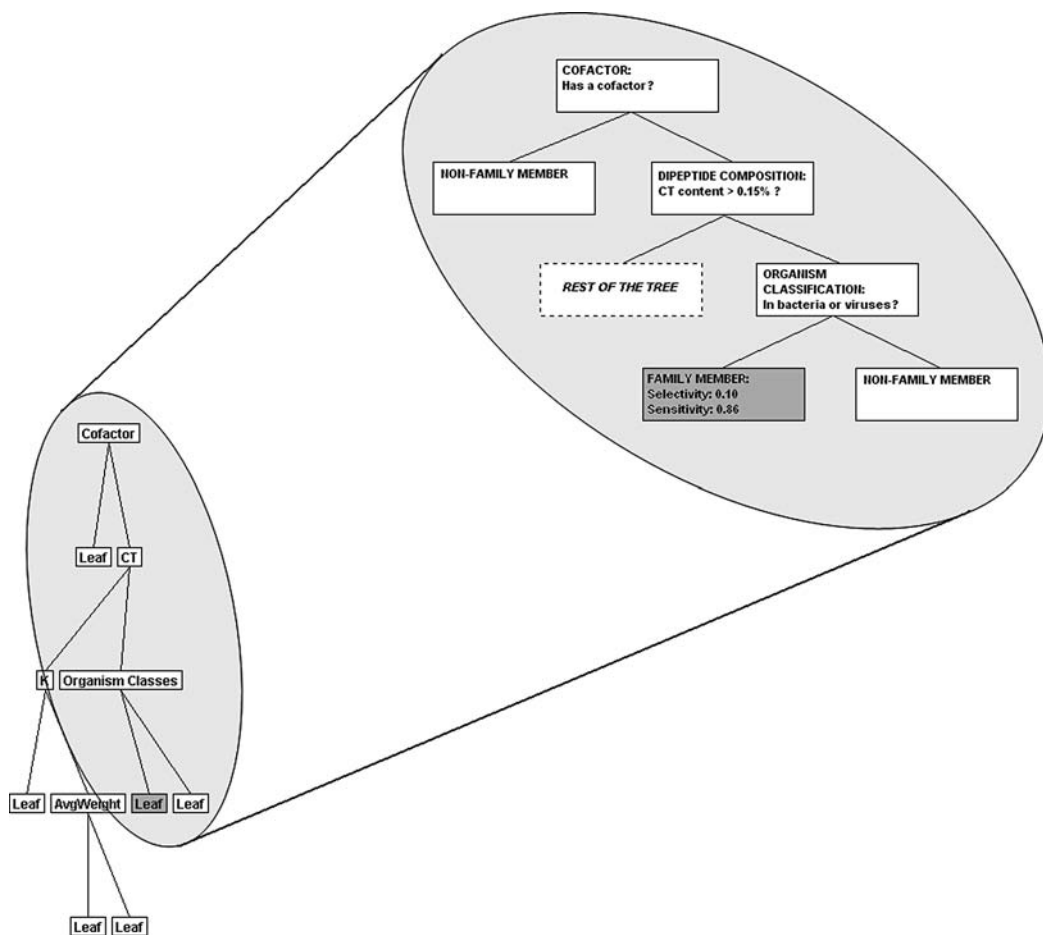


Fig. 17.8. **A decision tree for EC family 1.18.99.1.** The complete tree is shown at lower-left. The part of the tree discussed in the text has been enlarged. The *highlighted rule* detects members of 1.18.99.1 with 10% selectivity (compared to background probability of 0.001) and 86% sensitivity.

best BLAST). Twenty percent of trees in the mixture for this family use the co-factor attribute in the root split of the tree, including 90% of the deterministic trees.

Another type of information that can be extracted from these trees is structural information. For example, family members of 1.10.3.2 (including laccases and urishiol oxidases) tend to have high predicted beta sheet content. Indeed, members of 1.10.3.2 with solved 3D structure all belong to “mainly beta” class in CATH and “all beta” class in SCOP. Again, by itself, this information is clearly inconclusive. However, proteins that have high predicted beta sheet content, above-average content of P and N residues, and are present in flowering plants, are significantly enriched for membership in 1.10.3.2 (selectivity = 0.30, compared to background probability of 0.001, and sensitivity =

0.97). A similar observation can be made about enzyme family 3.6.1.23. In this case, proteins low in predicted helix content, high in tryptophan, and present in flowering plants are disproportionately likely to be members of 3.6.1.23 (selectivity = 0.10, compared to background probability of 0.002, and sensitivity = 0.85).

Many such rules can be extracted from the set of trees. It must be emphasized that only on rare occasions is a single rule sufficient to produce a highly accurate classifier (such as with family 1.14.14.1, described above). It is the combination of many rules in the mixture of stochastic decision trees that yields superior performance over BLAST. However, the examples given in this section illustrate that the non-sequence attributes used in our algorithm do contribute substantially to the overall performance of the mixtures.

7. Conclusions

The gene data that are available today present us with a major challenge: how to decipher the rules that define protein functionality. Obviously, the sequences of these proteins have a bearing on this question. According to the central dogma of molecular biology, it is the protein sequence that dictates the structure of the protein and this structure in turn prescribes the biological function of the protein. Moreover, sequence similarity can suggest common heritage and similar functions. Traditionally, sequence analysis has played a significant role in predicting gene function. However, there are a growing number of protein families that cannot be defined based on sequence similarity, and are linked and grouped based on other features. Some may argue that the tools that are available today for sequence analysis already exhaust sequence information. Indeed, in some cases it seems that new evidence must be presented, either in the form of structural similarity, expression data or protein interaction data before relationships can be established.

Here we argue that there is more information to utilize from sequence data. Our ability to analyze an entity and explore its relationships with other entities really depends on the representation we employ for that entity. Traditionally, proteins were analyzed as sequences of amino acids. This fairly simple representation was complex enough to detect many relationships. But once the order of amino acids was eliminated much of the information was lost. In this paper we present a new representation for proteins, through an extensive set of attributes, and introduce a novel model to decipher the regularities that distinguish family members from non-members. The model uses only sequence data and features

extracted from database records, but can detect subtle principles that cannot always be established based on pure sequence similarity.

The set of attributes covers a broad range of protein features, from basic compositional properties to physiochemical features, predicted features that hint at the topology and the shape of the protein structure, and database attributes that provide additional information about the subcellular location of the protein, its tissue preferences and other features that are likely to be linked to protein functionality. This set of features can be extended as more data become available. The combination of nominal attributes with numeric attributes in our model suggested the use of decision trees, one of the very few machine learning techniques that can handle mixed data. In addition to being suited for our learning task, the model can tolerate noise and missing data, as is often the case with database attributes. However, even the most recent learning algorithms for decision trees proved unsuccessful and the accuracy of the learned models was insufficient for effective prediction. Therefore, we embarked on developing a new model based on the decision tree model, which we call stochastic decision trees. In addition, we introduce several other components and options in the learning algorithm, such as different pruning methods and different weighting schemes. To find the optimal learning strategy we search through the space of decision tree learning algorithms until we converge to a locally optimal strategy. Four main elements characterize our final model: (1) dynamic feature and value selection for large feature and value sets, (2) probabilistic attribute selection protocol, (3) the use of a mixture of trees, and (4) the use of positive and negative sample divergence properties in all stages of tree learning, from pruning to tree weighting and evaluation. We use the Jensen-Shannon divergence measure to assess the trees in terms of the separation they induce over the data set between the positive and the negative samples (members and non-members). Trees that maximize the separation are assigned a higher score (weight) and play a more dominant role in prediction. Thus the learning process essentially maximizes the buffer between the positive and the negative samples. One might notice the similarity with kernel-based methods such as Support Vector Machines that try to maximize the margins around the separating hyper-planes. A good alternative to the JS-based pruning is the MDL-based approach that is described in the **Note 2**. This approach compensates for some of the deficiencies of validation-based post-pruning and is especially useful with small datasets, where not enough samples are available for training and validation.

To assess our model we evaluated it over two well-known classifications; the Pfam sequence-based classification and the EC function-based enzyme classification. The results were compared

to the popular BLAST algorithm and SAM, an HMM-based alignment program. Indeed our model compares favorably with BLAST and SAM over the Pfam data set, but the power of our model is more pronounced when the protein family is defined based on function, as in the EC database, rather than just based on sequence. Although for most EC families sequence similarity is still the most dominant factor in their definition, these families are less conserved than Pfam families, and in some cases are composed of multiple sequence subfamilies. Learning such families is a complex task, especially when the subfamilies do not seem to be related and cannot be reliably aligned. As was demonstrated here, our method can be applied successfully to predict the enzyme class of a protein, a task that sequence similarity-based methods often perform poorly on.

One of the advantages of our method is that the sequences need not be aligned. The model can learn the features common to a diverse set of proteins of shared function, sometimes without even evident sequence similarity. When these features are clearly a property of the protein family and are not found in other sequences, then they serve as good predictors and are integrated into the model.

There are several modifications that may improve performance. One modification that we are considering is to modify the pruning algorithm and switch from a local approach to a global approach (where all nodes are considered before deciding on the node to be pruned). Another modification would be to assign probabilities to attribute values, since for some nominal attributes it is possible to quantify the likelihood of the different values. Clearly, integration of other features can refine the models and, finally, boosting techniques can also help to improve the performance.

8. Notes



1. *The extended training procedure for decision trees.* Our extended decision tree learning algorithm includes the following elements:
 - a. *Dynamic attribute filtering.* In the case of dipeptide composition, the number of attributes is too large to be used exhaustively during training. To narrow the scope of data that will be input to the decision tree algorithm, we need a filtering criterion that can examine the training data and retain only those attributes that seem important. This importance is estimated by computing the ratio of the frequency of a dipeptide among family members to its frequency among all non-family members. For example, for an attribute like “AA”,

this ratio is the percent composition of “AA” in family member sequences divided by the percent composition of “AA” in non-family member sequences. Note that these calculations are restricted to the training set only. We retain only those dipeptides for which this ratio is one standard deviation from the mean. However, to keep the computation feasible, we used at most the top 30 such dipeptides.

- b. *Discretizing numerical features.* In a decision tree, numerical features must be divided into subranges before they can be used to split a node. Many researchers have approached this problem and the best solution is by no means clear. The Fayyad-Irani heuristic (57) is an efficient and flexible method for discretizing numerical attributes. It is a greedy recursive protocol, inspired by the Minimum Description Length principle, which produces a partition of k -arity, where k is selected by the protocol itself. Given a k -ary partition, the algorithm splits it into a $(k+1)$ -ary partition by examining every remaining cut point and selecting the one that maximizes an MDL-like criterion function. If its value is less than zero for every remaining cut point, the protocol terminates and the current partition is used. The specific formula for scoring cut points is

$$Gain(S, A, T) = \frac{\log_2(N-1)}{N} - \frac{\Delta(S, A, T)}{N}$$

where N is the number of samples, T is the cut point and

$$\begin{aligned} \Delta(S, A, T) = & \log_2(3^k - 2) - [k \times Entropy(S) - k_1 \\ & \times Entropy(S_1) - k_2 \times Entropy(S_2)] \end{aligned}$$

with k , k_1 , and k_2 being the number of classes in each set (2 in our case). For more details see (57).

Although others have used this protocol to globally partition numerical attributes (58), we use it at every decision node to induce local partitions of the numerical attributes based on the given subset of sample points at that node, thus adjusting the decision to the subset of data available. The protocol can also be constrained to produce only binary splits (see **Binary splitting**, below).

- c. *Binary splitting.* While many of our attributes are naturally suited to having multiple values, we wanted to test the impact of forcing all splits to be binary. This preserves training data for later splits, and consequently may improve accuracy, particularly since a very small fraction of our data set represents class (family) members. Moreover, binary splits are not susceptible to the biased entropy of many-way splits (see **Selection measures** below). With numerical attributes, the most straightforward procedure is also the

most efficient and exhaustive: simply restrict discretization to exactly two subranges (i.e., a single cut point). For a particular attribute, the optimal split can be found in $O(N)$ time, where N is the number of possible cut points (57). However, for nominal attributes, since there is no ordering over the values, the exhaustive procedure of testing every possible binary partition of the values leads to a time complexity of $O(2^d)$ where d is the number of possible values for the attribute $d = \text{Values}(A)$. There are at least two ways to deal with this problem. The most naive way is to examine only those binary partitions of $\text{Values}(A)$ in which one value is assigned to one branch and all other values are assigned to the other branch. This is functionally equivalent to converting each value of A into a separate, purely binary attribute, and it reduces the time complexity to $O(d)$.

A more sophisticated partitioning algorithm is found in CART (59). Our problem can be expressed as having to find a subset of $\text{Values}(A)$ to assign to one child, such that $\text{Gain}(S, A)$ is maximized. Let each value $v_i \in \text{Values}(A)$ correspond to a subset S_i of S , which will consist of the examples that have v_i as a value. Sort the v_i 's in order of increasing class purity of the S_i 's. In this new order, the optimal subset is the union of S_0, \dots, S_k , where k is some number in $\{0, \dots, d\}$. Therefore, one can find the optimal binary split in only $O(d)$ time, excluding sorting time. This optimal partitioning algorithm is the one we use during tree training.

- d. *Multiple values for attributes.* The nominal database attributes that we use are somewhat different than those typically found in the literature on decision trees. Each example can take on several values for each nominal attribute, instead of just one. For example, a protein can be abundant in the liver as well as in heart tissues, and the number of possible combinations is huge. During training, when a node is split using a nominal attribute A , the set $\text{Values}(A)$ is partitioned across the node's children. With traditional nominal attributes, each example would be assigned to the unique child that corresponds to its value. In our model, however, each example may correspond to several children. To overcome this problem we divide the example itself across all children, weighting each child's portion by the number of values the example has in common with that child.
- e. *Missing attributes.* Sometimes, an individual example will lack the attribute being tested by a decision node. We adopt the approach implemented in the C4.5 algorithm to handle these samples, and partition them across the child nodes maintaining the same proportions as the rest of the training examples (i.e., the proportions are

calculated using only samples that have reached the splitting node and that have known values for the test attribute). These fractions can be partitioned again and again with every unknown attribute. Consequently, the ambiguous examples will reach several leaf nodes during classification. In these cases, the final probability is defined as the average of all leaf probabilities, weighted by the fraction of the example that reached each leaf.

- f. *Selection measures.* The C4.5 algorithm improves upon ID3 by using $\text{GainRatio}(S, A)$ (48), which is $\text{Gain}(S, A)$ normalized by the *split information*:

$$\text{SplitInfo}(S, A) = - \sum_{v \in \text{Values}(A)} \frac{|S_v|}{|S|} \log_2 \frac{|S_v|}{|S|}$$

Note that this expression is just the information content of the attribute itself. This modification is meant to compensate for the bias of $\text{Gain}(S, A)$, which tends to be higher for attributes with more values. However, $\text{GainRatio}(S, A)$ is itself less than ideal, since it may be biased toward attributes with very low $\text{SplitInfo}(S, A)$ instead of high $\text{Gain}(S, A)$. To counter this, we also tested a closely related distance metric introduced by Mantaras (60), who showed formally that his measure does not suffer from the limitations of $\text{Gain}(S, A)$ or $\text{GainRatio}(S, A)$ (though there is empirical evidence that it is not guaranteed to eliminate the bias altogether (61)). To simplify notation, in the following sections we refer to both GainRatio and the *Mantaras* selection functions as **information gain**, unless specified otherwise. Note that these measures are only used when computing multi-branch splits, since their purpose is to combat high arity splitting. The basic $\text{Gain}(S, A)$ is sufficient when dealing with binary splits.

- g. *Handling skewed distributions.* When trying to discern protein families, the overwhelming majority of the training set is composed of negative examples (usually more than 90%). As a result, one may run into problems when trying to learn models of small families. Because of its greedy nature, most of the learning phase will concentrate on learning the regularities in the negative examples that result in significant information gain. Not only will this process cause extended learning times but it may also totally miss subtle regularities observed in the positive samples. The simple solution of presenting only a small set of negative samples equal in size to the set of positive samples is insufficient because some regularities observed in the positive set may be mistakenly considered as discriminating regularities, while if considered along with the whole set of negative examples they may prove to be uninformative.

To overcome this problem, we adopted a different approach where all the negative examples are presented, but their sum is given the same weight as the sum of all the positive examples (as suggested in (62)). Thus, every positive example contributes $1/\#positives$ and every negative example contributes $1/\#negatives$ to the total counts, and are weighted accordingly in the calculations of the information gain.

However, by doing so we introduce another problem. It may happen that we will end with a tree of high purity (when using the weighted samples), but of high impurity when considering the original unweighted samples, thus increasing significantly the rate of false positives. We tested two possible solutions that attempt to balance the two approaches. In the first, the criterion function for splitting and pruning is the average of the weighted-sample information gain with the unweighted-sample information gain (this protocol is called **mixed entropy**). The second solution starts by training the trees using the weighted samples. The usage of unweighted-sample entropy is delayed until the maximal information gain over all possible attributes at a given node is below a certain threshold. At that point each attribute is reconsidered using unweighted entropy and the training proceeds for as long as the information gain is below the threshold (we call this protocol **interlaced entropy**).

Note that the weighted samples are used only during the calculations of the information gain. For all other purposes, including pruning and prediction, the probabilities based on the unweighted samples are used.

- h. *Leaf weighting*. Not all leaf nodes are good predictors and different nodes may have different accuracies, when tested on new samples. This is especially true if the number of samples in a node is small. To address that we tested also a variant where weights are associated with leaf nodes, and the final prediction is adjusted accordingly. The weights are trained using the perceptron learning algorithm (36), until a local minimum of the performance is achieved over the training set.
2. *Post-pruning by the MDL principle*. One of the major problems of validation-based post-pruning methods is that a sufficiently large number of positives in the validation set are required to ensure high generalization power. Otherwise, many true regularities learned from the positives in the training set may be pruned, since they are not supported by the small validation set. The effect of post-pruning on decision trees in such cases can be drastic.

With very small data sets, where every single sample is important, one might want to consider alternative pruning methods that use all the given data. The chi-squared test, used in a pre-pruning context, is one possibility. However, it may suffer from what is called the *horizon effect*. (This refers to the phenomenon where decision node splits that initially appear poor in fact lead to better performance later on during the training process. Any pre-pruning technique will be susceptible to this kind of short-sightedness (36)). Here we use an alternative method based on the **minimum description length principle** (63), abbreviated as MDL.

The MDL principle is a common heuristic for selecting the most probable or valid model. It is based on an argument that was postulated by Occam in the 14th century. According to this argument (called *Occam's razor*), given two models that explain an observation equally well, it is advised to select the simpler model, under the assumption that it will generalize better to new examples. Therefore, the most probable model (of all possible models) for a given data set is the one that minimizes the complexity of the model while maximizing its ability to explain the data. This concept is captured by the **description length**. The description length of a given **model (hypothesis) and data** is defined as the description length of **the model** plus the description length of the **data given the model**.

One way to define the description length of a model is by its algorithmic complexity (Kolmogorov complexity). However, this approach is not very practical, since to define the algorithmic complexity of a model, one has to find the shortest program that encodes this model, something that is hardly ever possible.

A more practical method that formulates the same principle in probabilistic terms is the Bayesian approach. Under this approach, the most probable model is the one that maximizes the posterior probability of the model given the data

$$P(h|D) = \frac{P(h)P(D|h)}{P(D)}$$

The optimal hypothesis h^* is the one that maximizes $P(h | D)$ or, equivalently, minimizes $-P(h | D)$

$$\begin{aligned} h^* &= \arg \min_h \left[\frac{-P(h)P(D|h)}{P(D)} \right] \\ &= \arg \min_h [-P(h)P(D|h)] \\ &= \arg \min_h [-\log_2 P(h) - \log_2 P(D|h)] \end{aligned}$$

By Shannon's theorem, $-\log_2 P(x)$ is related to the shortest description of a string x (in our case the model h), and the likelihood $-\log_2 P(D | h)$ is a measure for the description of the data given the model. Therefore, the MDL principle and the Bayesian approach as formulated above are practically the same.

To approximate the first term we calculate the binary encoding of a tree. For each node j we devote $\log N + \log k_j$ bits, where N is the number of attributes in our model and k_j is the number of possible values for the attribute associated with node j . In this way, the total description length of a tree is approximated by the sum of the descriptions of its nodes. Note that the number of possible trees increases exponentially with the description length; therefore, the probability of each tree decreases exponentially. Thus, the logarithm as applied to the probability of the tree generates a term that is linear in the binary encoding of the tree, as desired.

To calculate the second term we used two different approaches. The first, called **probability based MDL post-pruning**, calculates $P(D | h)$ based on the probabilities the model assigns to each sample. Specifically,

$$P(D|h) = \prod_{i=1}^n P(+|x_i) = \prod_{i=1}^n \sum_{j \in \text{leaves}(T)} f_j(x_i) \text{Prob}_j(x_i)$$

where $f_j(x)$ is the fraction of x_i that reaches leaf j and $\text{Prob}_j(x_i)$ is the probability assigned to x_i by that leaf. (The reason that a fraction of a sample can reach a node is related to the problem of missing values and is explained in **Note 1e**.) Intuitively, the likelihood term $\log P(D | h)$ accounts for the uncertainty that is left in the tree. With probability 1 the uncertainty is zero, or in other words, the tree describes the data perfectly. Otherwise, some uncertainty exists and we must include in the description the variance or the deviation of the data from the tree, since only a combined description of the tree and the variance will enable complete recovery of the data set. Our second approach, called **entropy-based MDL post-pruning**, attempts to directly estimate the total uncertainty that remains in the tree by measuring the total entropy in all the leaf nodes. Our tests indicate that the probability-based approach outperforms the entropy-based approach.

Finally, the two terms are combined together into a single measure. Although the MDL heuristic relies on sound arguments, the exact formulation requires some adjustment and one usually needs to introduce a factor α to fix the scale such that

$$\text{DescLen} = \text{ModelDesc} + \alpha \cdot \text{Likelihood}$$

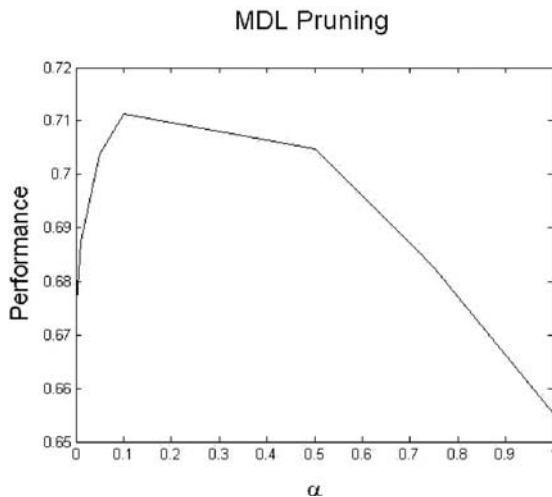


Fig. 17.9. **The minimum description length.** The description length consists of two elements, and the relative contribution of each is determined by the weight α (*see text*). We tested a range of possible values for over a random subset of 50 PFAM families to determine the one that maximizes the performance. For each value of α we trained a new set of trees and measured the average performance over the subset. The graph plots the average performance as a function of the value of α .

To determine α , we study the distribution of uncertainties in the trained trees versus the description lengths of the nodes, and set the range of possible values for α so that they scale similarly. The exact value is selected based on performance optimization (*see Fig. 17.9*).

To find the optimal tree, using this new pruning method, we train a complete tree and then prune it in an exhaustive fashion, similar to the post-pruning method described in **Section 3.3**, but using this new MDL criterion function. The elimination of the validation set provides us with more training samples at our disposal. However, it also eliminates a stochastic element that can increase robustness, as it unifies all ten training sets used by the post-pruning procedure into one. To compensate for the missing validation set in that aspect, we re-introduce ten training sets by using the bootstrapping sampling method (64) of the learning set. Specifically, each training set of size m is generated by selecting instances from the learning set at random, with repetitions. Theoretical results (65) indicate that models estimated from bootstrap data sets will converge to the optimal model at the limit of infinite bootstrap data sets. Moreover, the bootstrap method was proved to be more robust in providing reliable estimates of performance in the presence of small data sets (66).

With the MDL-based pruning and evaluation, the final output of the mixture model is given by

$$P_{\text{Mixture}}(+|x) = \frac{\sum_{\text{training set } j} \sum_{i \in \text{Trees}(j)} Q_{\text{MDL}}(i) P_i(+|x)}{\sum_{\text{training set } j} \sum_{i \in \text{Trees}(j)} Q_{\text{MDL}}(i)}$$

where $Q_{\text{MDL}}(i)$ is defined as $1/\text{MDL}(i)$ and $\text{MDL}(i)$ is the description length of the i th tree.

3. *Setting the threshold for evaluation and prediction.* When evaluating the performance of a model that assigns probabilities to samples, one needs a clear definition of positives and negatives. Usually a threshold T is defined, and if the sample probability exceeds this threshold, then the sample is defined as positive.

How should one define the threshold T ? We tested two different approaches. The naive approach would be to take a majority vote, so that all samples with probability higher than 0.5 are defined as positive. However, this may be misleading, since the number of positives and negatives may differ significantly to begin with. A more sensible approach would be to take into account the prior probabilities of positives $P_0(+)$ and negatives $P_0(-)$, and set the threshold to $P_0(-)$. To account for random fluctuations due to the varying sample sizes at the nodes, a different significance threshold T_j is calculated for each node. Specifically, if the total number of samples in a leaf node j is N_j , then the number of positives that will reach this node by chance (the null hypothesis) is expected to follow a binomial distribution with mean $N_j \times P_0(+)$ and variance $N_j \times P_0(+)$ $\times P_0(-)$. Using the normal approximation for the binomial distribution, we set the threshold at the 95th percentile of the distribution, and if the node probability $P_j(+)$ exceeds this threshold then we define this instance to be positive. Thus, each example is classified unambiguously in a discrete manner, and the accuracy is computed thereby. If the sample reaches a single leaf node, then only the confidence interval for this node is used to assign a label. If it reaches more than one node, a “combined” node is created, where the datasets are collected from the different leaf nodes, weighted by the fractions of the sample that reach each leaf node, and the sample probabilities, the threshold and the label are calculated accordingly. We refer to this approach as the **confidence-based** approach.

The second approach uses the equivalence point criterion. All sequences in the database are sorted according to the output assigned by the model. The equivalence point is the point where the number of false positives equals the number of false negatives (67). All proteins that are predicted with

higher probability are labeled as positives. In other words, the equivalence point is the point that balances sensitivity and selectivity. We refer to this approach as the **equivalence point** approach.

Acknowledgments

This work is supported by the National Science Foundation under Grant No. 0133311 to Golan Yona.

References

1. Kanehisa, M. and Goto, S. (2000) KEGG: Kyoto encyclopedia of genes and genomes. *Nucl. Acids Res.* **28**, 27–30.
2. Caspi, R., Foerster, H., Fulcher, C.A., Hopkinson, R., Ingraham, J., Kaipa, P., Krummenacker, M., Paley, S., Pick, J., Rhee, S. Y., Tissier, C., Zhang, P., and Karp, P. D. (2006) MetaCyc: a multiorganism database of metabolic pathways and enzymes. *Nucl. Acids Res.* **34**, D511–D516.
3. Paley, S. M. and Karp, P.D. (2002) Evaluation of computational metabolic-pathway predictions for *Helicobacter pylori*. *Bioinformatics* **18**, 715–724.
4. Bono, H., Ogata, H., Goto, S., and Kanehisa, M. (1998) Reconstruction of amino acid biosynthesis pathways from the complete genome sequence. *Genome Res.* **8**, 203–210.
5. Green, M. and Karp, P. D. (2004) A Bayesian method for identifying missing enzymes in predicted metabolic pathway databases. *BMC Bioinformatics* **5**, 76.
6. Chen, L. and Vitkup, D. (2006) Predicting genes for orphan metabolic activities using phylogenetic profiles. *Genome Biol.* **7**, R17.
7. Kharchenko, P., Chen, L., Freund, Y., Vitkup, D., and Church, G. M. (2006) Identifying metabolic enzymes with multiple types of association evidence. *BMC Bioinformatics* **7**, 177.
8. Popescu, L. and Yona, G. (2005) Automation of gene assignments to metabolic pathways using high-throughput expression data. *BMC Bioinformatics* **6**, 217.
9. Popescu, L. and Yona, G. (2006) Expectation-maximization algorithms for fuzzy assignment of genes to cellular pathways. *In proceedings of the 2006 Computational Systems Bioinformatics Conference.*
10. Yaminishi, Y., Vert, J., and Kanehisa, M. (2005) Supervised enzyme network inference from the integration of genomic data and chemical information. *Bioinformatics* **21**, i468–i477.
11. <http://www.chem.qmw.ac.uk/iubmb/enzyme/>
12. Shah, I. and Hunter, L. (1997) Predicting enzyme function from sequence: a systematic appraisal. *In the Proceedings of the 5th International Conference on Intelligent Systems for Molecular Biology* 276–283.
13. Wilson, D. B. and Irwin, D. C. (1999) Genetics and properties of cellulases. *Adv. Biochem. Eng.* **65**, 2–21.
14. Stawiski, E. W., Baucom, A. E., Lohr, S. C., and Gregoret, L. M. (2000) Predicting protein function from structure: unique structural features of proteases. *Proc. Natl. Acad. Sci. U.S.A.* **97**, 3954–3958.
15. Todd, A. E., Orengo, C. A., and Thornton, J. M. (2001) Evolution of function in protein superfamilies, from a structural perspective. *J. Mol. Biol.* **307**, 1113–1143.
16. Devos, D. and Valencia, A. (2000) Practical limits of function prediction. *Prot. Struct. Func. Genet.* **41**, 98–107.
17. Holm, L. and Sander, C. (1994) The FSSP database of structurally aligned protein fold families. *Nucl. Acids Res.* **22**, 3600–3609.
18. Wilson, C. A., Kreychman, J., and Gerstein, M. (2000) Assessing annotation transfer for genomics: quantifying the relations between protein sequence, structure and function

- through traditional and probabilistic scores. *J. Mol. Biol.* **297**, 233–249.
19. Murzin A. G., Brenner S. E., Hubbard T., Chothia C. (1995) SCOP: a structural classification of proteins database for the investigation of sequences and structures. *J. Mol. Biol.* **247**, 536–540.
 20. Rost, B. (2002) Enzyme function less conserved than anticipated. *J. Mol. Biol.* **318**, 595–608.
 21. desJardins, M., Karp, P. D., Krummenacker, M., Lee, T. J., and Ouzounis, C. A. (1997) Prediction of enzyme classification from protein sequence without the use of sequence similarity. *In the Proceedings of the 5th International Conference on Intelligent Systems for Molecular Biology* 92–99.
 22. Borro, L. C., Oliveira, S. R. M., Yamagishi, M. E. B., Mancini, A. L., Jardine, J. G., Mazoni, I., dos Santos, E. H., Higa, R. H., Kuser P. R., and Neshich G. (2006) Predicting enzyme class from protein structure using Bayesian classification. *Genet. Mol. Res.* **5**, 193–202.
 23. Cai, Y-D. and Chou, K-C. (2004) Using functional domain composition to predict enzyme family classes. *J. Proteome Res.* **4**, 109–111.
 24. The Gene Ontology Consortium. (2000) Gene ontology: tool for the unification of biology. *Nat. Genet.* **25**, 25–29.
 25. Clare, A. and King R. D. (2003) Predicting gene function in *Saccharomyces cerevisiae*. *Bioinformatics* **19**, ii42–ii49
 26. Altschul, S. F., Madden, T. L., Schaffer, A. A., Zhang, J., Zhang, Z., Miller, W., and Lipman, D.J. (1997) Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucl. Acids Res.* **25**, 3389–3402.
 27. Mewes, H. W., Heumann, K., Kaps, A., Mayer, K., Pfeiffer, F., Stocker, S., and Frishman, D. (1999) MIPS: a database for genomes and protein sequences. *Nucl. Acids Res.* **27**, 44–48.
 28. Burges, C. J. C. (1998) A tutorial on support vector machines for pattern recognition. *Data Min. Knowl. Discov.* **2**, 121–167.
 29. Jaakola, T., Diekhans, M., and Haussler, D. (1999) Using the Fisher kernel method to detect remote protein homologies. *In the Proceedings of the 7th International Conference on Intelligent Systems for Molecular Biology* 149–158.
 30. Han, L. Y., Cai, C. Z., Ji, Z. L., Cao, Z. W., Cui, J., and Chen, Y. Z. (2004) Predicting functional family of novel enzymes irrespective of sequence similarity: a statistical learning approach. *Nucl. Acids Res.* **32**, 6437–6444.
 31. Leslie, C., Eskin, E., Cohen, A., Weston, J., and Noble, W. S. (2004) Mismatch string kernels for discriminative protein classification. *Bioinformatics* **1**, 1–10.
 32. Ben-Hur, A. and Brutlag, D. L. (2006) Sequence motifs: highly predictive features of protein function, in *Feature Extraction, Foundations and Applications* (Guyon, I., Gunn, S., Nikravesh, M., and Zadeh, L. eds.), Springer Verlag, New York.
 33. Kolesov, G., Mewes, H. W., and Frishman, D. (2001) SNAPping up functionally related genes based on context information: a colinearity-free approach. *J. Mol. Biol.* **311**, 639–656.
 34. Tian, W., Arakaki, A. K., and Skolnick, J. (2004) EFICAZ: a comprehensive approach for accurate genome-scale enzyme function inference. *Nucl. Acids Res.* **32**, 6226–6239.
 35. Levy, E. D., Ouzounis, C. A., Gilks, W. R., and Audit, B. (2005) Probabilistic annotation of protein sequences based on functional classifications. *BMC Bioinformatics* **6**, 302.
 36. Duda, R. O., Hart, P. E., and Stork, D. G. (2000) *Pattern Classification*. John Wiley and Sons, New York.
 37. Mitchell, T. M. (1997) *Machine Learning*. McGraw-Hill, New York.
 38. Breiman, L., Friedman, J. H., Olshen, R.A., and Stone, C. J. (1993) *Classification and Regression Trees*. Chapman and Hall, New York.
 39. Bateman, A., Birney, E., Durbin, R., Eddy, S. R., Finn R. D., and Sonnhammer E. L. (1999) Pfam 3.1: 1313 multiple alignments and profile HMMs match the majority of proteins. *Nucl. Acids Res.* **27**, 260–262.
 40. Bairoch, A. and Apweiler, R. (1999) The SWISS-PROT protein sequence data bank and its supplement TrEMBL in 1999. *Nucl. Acids Res.* **27**, 49–54.
 41. Hobohm, U. and Sander, C. (1995) A sequence property approach to searching protein database. *J. Mol. Biol.* **251**, 390–399.
 42. Ferran, E. A., Pflugfelder, B., and Ferrara P. (1994) Self-organized neural maps of human protein sequences. *Protein Sci.* **3**, 507–521.
 43. Black, S.D. and Mould, D.R. (1991) Development of hydrophobicity parameters to analyze proteins which bear post or cotranslational modifications. *Anal. Biochem.* **193**, 72–82.

44. <http://www.ionsource.com/virtit/VirtualIT/aainfo.htm>
45. McGuffin, L. J., Bryson, K., and Jones, D. T. (2000) The PSIPRED protein structure prediction server. *Bioinformatics* **16**, 404–405.
46. <http://www.expasy.org/sprot/userman.html>
47. Quinlan, J.R., (1986) Induction of decision trees. *Mach. Learn.* **1**, 81–106.
48. Quinlan, J.R., (1993) *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Mateo, CA.
49. Syed, U. and Yona, G. (2003) Using a mixture of probabilistic decision trees for direct prediction of protein function. *In the Proceedings of the 7th Annual International Conference on Research in Computational Molecular Biology* 289–300.
50. Dietterich, T. G. (2000) An experimental comparison of three methods for constructing ensembles of decision trees: bagging, boosting, and randomization. *Mach. Learn.* **40**, 139–157.
51. Ho, T. K. (1998) The random subspace method for constructing decision forests. *IEEE Trans. Pattern Anal. Mach. Intell.* **20**, 832–844.
52. Breiman, L. (2001) Random forests. *Mach. Learn.* **45**, 5–32, 48
53. Lin, J. (1991) Divergence measures based on the Shannon entropy. *IEEE Trans. Inf. Theory* **37**:1, 145–151.
54. Kullback, S. (1959) *Information Theory and Statistics*. John Wiley and Sons, New York.
55. Hughey, R., Karplus, K., and Krogh, A. (1999) SAM: sequence alignment and modeling software system. Technical report UCSC-CRL-99-11. University of California, Santa Cruz, CA.
56. Birkland, A. and Yona, G. (2006) The BIOZON database: a hub of heterogeneous biological data. *Nucl. Acids Res.* **34**, D235–D242.
57. Fayyad, U. M. and Irani, K. B. (1993) Multi-interval discretization of continuous-valued attributes for classification learning. *In the Proceedings of the 13th International Joint Conference on Artificial Intelligence* 1022–1027.
58. Kohavi, R. and Sahami, M. (1996) Error-based and entropy-based discretization of continuous features. *In the Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining* 114–119.
59. Breiman, L., Friedman, J. H., Olshen, R.A., and Stone, C. J. (1984) *Classification and Regression Trees*. Wadsworth Int. Group, Belmont, CA.
60. Mantaras, R. L. (1991) A distance-based attribute selection measure for decision tree induction. *Mach. Learn.* **6**, 81–92.
61. Kononenko, I. (1995) On biases in estimating multi-valued attributes. *In the Proceedings of the 14th International Joint Conference on Artificial Intelligence* 1034–1040.
62. Eskin, E., Grundy, W. N., and Singer, Y. (2000) Protein family classification using sparse Markov transducers. *In the Proceedings of the 8th International Conference on Intelligent Systems for Molecular Biology* 20–23.
63. Rissanen, J. (1989) *Stochastic Complexity in Statistical Inquiry*. World Scientific, Singapore.
64. Hjorth, J. S. U. (1994) *Computer Intensive Statistical Methods: Validation, Model Selection, and Bootstrap*. Chapman and Hall, London.
65. Jain, A. K., Dubes, R. C., and Chen, C. (1998) Bootstrap techniques for error estimation. *IEEE Trans. Pattern Anal. Appl.* **9**, 628–633.
66. Shakhnarovich, G., El-Yaniv, R., and Baram, Y. (2001) Smoothed bootstrap and statistical data cloning for classifier evaluation. *In the Proceedings of the 18th International Conference on Machine Learning* 521–528.
67. Pearson, W. R. (1995) Comparison of methods for searching protein sequence databases. *Protein Sci.* **4**, 1145–1160.

Chapter 18

Using Evolutionary Information to Find Specificity-Determining and Co-evolving Residues

Grigory Kolesov and Leonid A. Mirny

Abstract

Intricate networks of protein interactions rely on the ability of a protein to recognize its targets: other proteins, ligands, and sites on DNA and RNA. To recognize other molecules, it was suggested that a protein uses a small set of specificity-determining residues (SDRs). How can one find these residues in proteins and distinguish them from other functionally important amino acids? A number of bioinformatics methods to predict SDRs have been developed in recent years. These methods use genomic information and multiple sequence alignments to identify positions exhibiting a specific pattern of conservation and variability. The challenge is to delineate the evolutionary pattern of SDRs from that of the active site residues and the residues responsible for formation of the protein's structure. The phylogenetic history of a protein family makes such analysis particularly hard. Here we present two methods for finding the SDRs and the co-evolving residues (CERs) in proteins. We use a Monte Carlo approach for statistical inference, allowing us to reveal specific evolutionary patterns of SDRs and CERs. We apply these methods to study specific recognition in the bacterial two-component system and in the class Ia aminoacyl-tRNA synthetases. Our results agree well with structural information and the experimental analyses of these systems. Our results point at the complex and distinct patterns characteristic of the evolution of specificity in these systems.

Key words: Specificity-determining residues, co-evolving residues, correlated mutations, mutual information, Monte Carlo, protein evolution, two-component system, aminoacyl tRNA synthetase.

1. Introduction

The structure of complex biological networks is determined by a large number of protein–ligand interactions. Each interaction involves a protein precisely recognizing and binding its target: a protein, a site on DNA, or a small molecule. What determines the

specificity of these interactions on a molecular level? How does specificity change during evolution? How can we alter the existing or engineer novel specificity of a protein?

It was suggested that for many proteins specificity is determined by a small set of amino acids, so-called specificity determining residues (SDRs) (1, 2). Mutations of SDRs can lead to altered (not necessarily diminished) specificity, i.e., one can mutate SDRs to make proteins bind new targets.

It is important to make a distinction between SDRs and other amino acids forming a binding pocket or a protein interface. While the binding interface of a protein can be very extensive and mutations of this interface can affect binding, only a small group of SDRs is responsible for *specific* interactions. For example, a DNA-binding protein can have a large non-specific DNA-binding interface through electrostatic interactions with the DNA backbone, while only a few amino acids (SDRs) form specific interactions with the nucleotides and determine the sequence of DNA to be recognized by the protein (1). Mutations of amino acids forming the binding interface can affect the affinity of binding, while mutations of SDRs can affect the specificity of recognition.

Finding SDRs is an important problem in molecular biology. While crystallographic and NMR structures of proteins and protein complexes reveal organization of the binding interface, they provide little information on the relative importance and energetics of interactions of individual amino acids with the target. Usually, SDRs are determined by trial-and-error (through structure-guided) mutations of the protein. While finding mutations that diminish binding can be relatively easy, identifying mutations that alter specificity may require a large amount of experimental work.

Our approach, in contrast, is based on the use of evolutionary information as “lab notebooks from the nature’s laboratory”. Indeed, a large number of mutations have been made, and only those that provide viable and fit phenotypes have been recorded in genomes of different organisms. Here we show how this information can be used to understand protein specificity and detect SDRs.

We use two approaches to identify SDRs. Both methods look for specific patterns of amino acid evolution as a signature of SDRs. One method uses proteins grouped by specificity and finds residues that are the same within a group (among proteins of the same specificity) and are different between the groups (among proteins of different specificities). Another method uses co-evolving residues in interacting proteins and identifies positions that co-evolve, i.e., a substitution of an amino acid in one protein correlates with a specific substitution of an amino acid in its interaction partner.

Importantly, our methods rely on a novel method to calculate the statistical significance of detected residues. A statistical control takes into account potential biases due to specific evolutionary history and structure of considered proteins. Neither approach requires the knowledge of the proteins' structure, thus allowing us to use structure to validate our predictions.

Several similar techniques have been developed in the field. Different techniques require different inputs (e.g., some need structures), rely on somewhat different set of assumptions, and use different methods to compute statistical significance of their predictions.

Livingstone and Barton (3) grouped sequences according to overall sequence similarity, functional similarity, origin or other criteria. Conservation scores took into account physicochemical properties of each amino acid. This technique has been applied to SH2 domain family, and it has been shown to correctly identify locations of core α -helices. No scheme for the calculation of statistical significance has been reported.

Lichtarge et al. (4) further developed this idea into a method dubbed evolutionary tracing (ET). In ET, sequences are divided into subgroups by cutting phylogenetic tree at different cutoffs. Each cutoff provides a grouping, which in turn provides a list of putative specificity determining residues. SDRs are detected as residues that are 100% conserved within every subgroup and not conserved across the whole family. Such definition is sensitive to small rare substitutions and does not require the sought residue to be different in different subgroups. To filter-out false positives emerging in such selection process, 3D structure is employed. Residues forming surface patches are predicted as SDRs. It has been demonstrated by Lichtarge et al. that the different partitionings obtained by applying different cutoffs on the tree of SH2 domains indeed results in finding residues of different specificity levels. The method has been applied to several protein families, including SH2, SH3, nuclear hormone receptors, G-proteins and G-protein coupled receptors, zinc-binding domains, and the RGS/G-protein interaction. The main drawbacks of the method are that (1) it requires the knowledge of a 3D structure, (2) it does not evaluate of statistical significance of predicted residues, (3) SDRs not located in the surface patches (e.g., specificity pockets) are missed.

Hannenhalli and Russel presented another method of detection of SDRs using specificity subclasses (5). Unlike Lichtarge et al., the major focus of their work was on the problems of identifying regions that confer specificity of subtypes already known and of predicting subtypes for "orphan" sequences. Given a multiple sequence alignment and a classification of different subtypes (e.g., difference in enzyme specificity), the method utilizes the relative

entropy value to determine such positions in hidden Markov profiles of subtypes that are most discerning of each subtype. Also hidden Markov profiles were used to predict the subtype for unclassified proteins. Authors performed a large-scale assessment of their method by applying it to PFAM collections of multiple alignments partitioned into subtypes by using SWISS-PROT functional assignments. The method has been shown to be able to detect positions known to confer specificity in close agreement with experiment. Good performance of the method in terms of predicting protein subclass has been demonstrated. The method suffers similar drawbacks: (1) division into sub-grouping is arbitrary user-defined and (2) statistical significance is not evaluated. Statistical significance is important not only for the elimination of false predictions but for the discrimination between several evolutionary patterns. Kalinina et al. (6) used substitution matrices to account for varying rates of substitution of different amino acids and, following our earlier work (7), utilized shuffled MSAs and a linear correction technique to compute statistical significance. In addition, they have developed method to select cutoff values, which uses a Bernoulli estimator. Recently, Pei et al. (8) have developed a method called SPEL that using log-likelihood ratios finds positions which evolve significantly differently from a random model of evolution and, correspondingly, from the phylogenetic tree for a given MSA. The latter two methods have been shown to perform well on the LacI protein family.

CERs, which are also called “correlated mutations”, are residues in the protein sequence where a mutation in one region is compensated by mutation at a certain position(s) in the same or other protein. CERs have been shown often to participate in contact regions of the proteins (9, 10). They can be also be used to improve the prediction of protein structure and docking (11).

Several approaches have been developed to detect CERs using MSAs. For example, Göbel et al. (10) used correlation coefficient matrices to detect CERs. At the same time Shindyalov et al. (12) introduced a phylogeny reconstruction-based method to take into account phylogenetic relations. Pollock et al. used a maximum likelihood method to detect CERs (13). Numerous other approaches have been developed including machine learning-based approaches (14). Yu et al. (15) tried to combine CERs and SDRs in their surface patch ranking (SPR) method, which uses a support vector machine (SVM) to find the correlated positions that best discriminate between protein domains grouped according to their function. Their method, however, requires knowledge of the protein structure and cannot distinguish between functional and phylogenetic signal.

2. Model of Protein Evolution

2.1. Sources of Conservation and Variation

Protein sequences evolve under several constraints. A globular protein must fold into a 3D structure, get transported into the right compartment, bind its targets, catalyze reactions, etc. Mutations that disrupt some of these functions lead to lower fitness of the organism and thus are eliminated in evolution. Each constraint leads to a specific pattern of conservation and variation in protein sequences. Here we focus on four major sources (evolutionary signals): structural, non-specific functional, specificity-determining, historic.

Structural signal leads to higher conservation of buried, mostly hydrophobic residues, while letting non-functional solvent-exposed residues to vary to some extent among hydrophilic ones. The *non-specific functional* constraint causes amino acids of the active site to be highly conserved, while keeping amino acids involved in the formation of binding interfaces to be somewhat conserved as well (e.g., binding the DNA backbone or formation of a hydrophobic-pocket to recognize a protease target). *Specificity-determining* amino acids are expected to be conserved among homologs of the same specificity, while being different among proteins of different specificity. Finally, *historic* signal is a reflection of the individual phylogenetic history of a protein family, leading to a higher preservation of amino acid sequence among proteins that diverged more recently (e.g., orthologous proteins that were separated by recent speciation). Proteins that have diverged earlier in the history are expected to be more diverged in sequence as well (e.g., distant homologs).

In analyzing patterns of protein evolution, we should take into account all these factors and try to separate them to find the signal of interest. Here we focus on revealing *specificity-determining signal*.

To separate the signals we use the following strategy. First, we compute the measurement of SDR signal in the alignment of natural proteins. Second, we obtain the parameters of this alignment, such as amino acid composition and the phylogenetic tree of the family and generate pseudo-random protein sequences that have the same amino acid composition at each position and the same phylogenetic tree of the whole family. Third, we compute the probability of obtaining observed natural signal in the pseudo-random proteins. Residues that have very low such probability are considered as statistically significant predictions (*see Section 3* below).

2.2. Consequences of Protein Duplication

Upon duplication, proteins can diverge in their specificity and function (16, 17). We take advantage of this divergence and assume that paralogous proteins in closely related genomes have

changed their specificity but did not have sufficient time to significantly change the type of binding partners or alter the geometry of the binding site.

2.3. Using Genomic Proximity

In identifying CERs, we use the genomic proximity of genes of bacterial two-component system as a predictor that these genes participate in the same pathway and interact with each other (18, 19).

3. Methods

3.1. Outline

Here we briefly outline our procedure. Details of the method are presented in the following sections.

3.1.1. Finding SDRs

1. *Data collection and alignment.* We start from a single sequence, collect groups of orthologous and paralogous proteins, and build a multiple alignment of them. The alignment is grouped such that orthologs are put together, while paralogs in separate groups.
2. *Calculation of mutual information.* Calculate mutual information for each position in the alignment and for the group index.
3. *Estimation of statistical significance.* Construct a pseudo-random multiple sequence alignment that has the same amino acid composition of each column and the same phylogenetic tree as the real alignment used above. Calculate the distribution of the mutual information in each position of the alignment. Compare random and real mutual information and identify statistically significant SDRs
4. *Validation.* Use known structures of proteins and their complexes with ligands to test (i) clustering of obtained SDRs in space; (ii) surface/pocket locations of SDRs; (iii) proximity between SDRs and ligands, active sites, etc.

3.1.2. Finding CERs

1. *Data collection and alignment.* To find CERs, we start from a pair of interacting proteins, collect and align homologs for each of them, and establish pairwise interactions between collected proteins (e.g., by using proximity in a bacterial genome as an evidence for interaction). This produces a double alignment, i.e., side-by-side alignments of interacting proteins.
2. *Calculation of mutual information.* Calculate mutual information for each pair of positions in the two proteins (alignments).
3. *Estimation of statistical significance.* Construct a pseudo-random multiple sequence alignment for both participating proteins. Each pseudo-random alignment has the same

amino acid composition of each column and the same phylogenetic tree as the real alignments used above. Calculate the distribution of the mutual information in each pair of position of the alignments. Compare random and real mutual information and identify statistically significant CERs.

4. *Validation*. Use known structures of proteins and their complexes to test (i) clustering of obtained SDRs in space; (ii) surface/pocket locations of SDRs; (iii) proximity of SDRs to the protein–protein interface.

3.2. Finding SDRs

3.2.1. Pattern of SDR Evolution

Due to their nature, SDRs exhibit a particular pattern of evolution; they can be highly conserved among proteins carrying out the same function in related organisms (e.g., orthologs). On the other hand, SDRs are typically different in homologous proteins that have different functions (e.g., paralogs) (Fig. 18.1).

3.2.2. Ortholog Detection and Clustering

If specificities of proteins of interest are known (e.g., from experiment), one can simply group proteins by specificity and apply our algorithm for the identification of SDRs. Unfortunately, such information is not available for most of the proteins. Instead, we use orthology and paralogy relationships between proteins to group them and detect SDRs. Each group consists of orthologs from closely related species. Paralogs, in contrast, are placed into separate

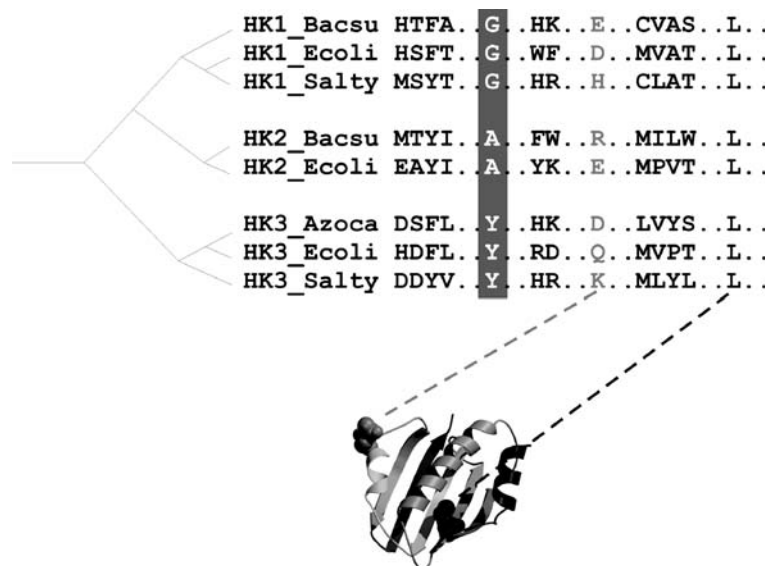


Fig. 18.1. Example of a multiple sequence alignment of three orthologous groups. Variable residues on the surface of the protein are depicted in *light gray*, conserved residue in hydrophobic core in *dark gray* and putative SDRs in *white on gray* background. Correlation of the group of orthologs and type of amino acid at putative SDR positions may arise from the phylogenetic history of the family, depicted as a phylogenetic tree on the *left*.

groups. By assuming that orthologs have the same specificity and paralogs have different specificities, we obtain the sought grouping of proteins by specificity.

In this paper we focus on prokaryotic genomes, where, in general, the degree of duplication of specific function is typically negligible, and best-to-best similarity matches are often used as orthologs. We apply the following simple strategy to derive groups of orthologs. (1) A number of genomes are scanned with an HMM domain profile and domain sequences are extracted; (2) domain sequences are aligned all-against-all using BLAST or PSI-BLAST program (3) best-to-best hits are identified and a graph is constructed where proteins are the nodes and best-to-best relations are the edges (4) starting from a protein in the largest genome the graph is traversed depth-first with the depth 2. All proteins (domains) encountered in the traversal are assigned to the same orthologous groups. Groups of size one are eliminated.

3.2.3. Multiple Alignments

Accurate multiple alignments are crucial for our method. We used the recently developed MAFFT (20) and MUSCLE (21) algorithms to build multiple alignments.

Due to extensive domain shuffling and gene fusion events, it is impossible to construct reasonably good multiple alignments if whole protein sequences are used. Therefore, we extract a particular domain type from the set of given proteins using profile models such as PFAM or CDD and then use the extracted domains for multiple alignments.

3.2.4. Data Collection and Alignments for Bacterial Two-Component System

Twenty bacterial genomes were scanned with PFAM HMM profiles for corresponding domains: RR (Response_reg), DD (HisKA), ATPase (HATPase_c). It is known that often cognate HKs and RRs are located next to each other inside the operon. Triples of domains (RR, DD and ATPase) that were nearest to each other and occurring within the same putative operons were detected. Putative operons were loosely defined as the strings of genes on the same strand of DNA separated by no more than 400 bp. The domain triples were then considered as interacting domains later in the correlated mutations analysis. Sets of each domain type were then aligned using MUSCLE multiple alignment program.

To obtain clusters of orthologs, triples of domains were concatenated to form pseudo- proteins and then aligned pairwise all-against-all using BLAST program (22). Best-to-best matches were then identified. For each pseudosequences in the largest genome (*Bradyrhizobium japonicum*) best-to-best matching pseudosequences in the other genomes were extracted. Such clusters were considered to be groups of orthologs. Starting the clustering from the pseudosequences in the same genomes ensures that clusters do not contain paralogous sequences. This assumes that

prokaryotic genomes rarely contain close gene duplicates that carry out the same function. The largest genome was chosen in order to get the largest number of orthologous clusters. Clustering depth of 1 guarantees that clusters do not overlap. To ensure quality of the clusters, pseudosequences that were linked by just one best-to-best match to the clusters were recursively deleted from the cluster.

Concatenation of the triples of domains into pseudosequence implicitly incorporates genome neighborhood information into orthology clustering.

3.2.5. Calculation of Mutual Information

To identify residues that tend to be the same within groups and different between the groups, we use mutual information as a measure of association between a residue and group index. Mutual information is frequently used in computational biology for co-variational analysis in RNA and proteins.

If $x = 1, \dots, 20$ is a residue type, $y = 1, \dots, Y$ is the specificity index, which is the same for all proteins of the same specificity group and is different for different groups, and Y is the total number of specificity groups, then the mutual information at position i of the MSA is given by

$$I^i = \sum_{\substack{x=1, \dots, 20 \\ y=1, \dots, Y}} f_i(x, y) \log \frac{f_i(x, y)}{f_i(x) f(y)},$$

where $f_i(x)$ is the frequency of residue type x in position i of the MSA, $f(y)$ is the fraction of proteins belonging to the group y , and $f_i(x, y)$ is the frequency of residue type x in the group y at position i . Note that although $f(y)$ does not depend on position i in practice, $f(y)$ need to be recomputed for each position i since proteins that contains gaps at each position are skipped in calculations of $f_i(x, y)$. Use of mutual information requires $f(y)$ and $f_i(x)$ to be marginal distributions of a joint distribution $f_i(x, y)$, i.e.,

$$f_i(x) = \sum_{y=1, \dots, Y} f_i(x, y) \quad f(y) = \sum_{x=1, \dots, 20} f_i(x, y).$$

Using these relationships to calculate $f(y)$ and $f_i(x)$ avoids mistakes due to gaps in the alignments. Positions that have gaps in more than 20% of sequences are neglected.

Mutual information has several important properties: (1) it is non-negative; (2) it equals zero if and only if x and y are statistically independent; and (3) a large value of I^i indicates a strong association between x and y . Unfortunately, a small sample size and a biased composition of each column in the MSA influences I^i a lot. For example, positions with less conserved residues tend to have higher mutual information. Hence, we cannot rely on the value of I^i as a sole indicator of specificity, instead we estimate the statistical significance of I^i .

3.2.6. Estimating Statistical Significance

Orthologs are in general more similar to each other than to paralogs due to the topology of the underlying phylogenetic tree. This represents a problem for any method used to detect SDRs: on one hand, we are trying to find positions where the sequence correlates with tree topology (with orthologous groups); on the other hand, we can expect correlation of all residues and tree topology purely from the random mutation process (**Fig. 18.1**).

The goal is to distinguish an evolutionary pattern of SDRs from the background of signals arising due to phylogenetic history of the protein family and due to constraints imposed on protein sequence to enable it to fold into a stable structure (see above).

What are the characteristics of a purely phylogenetic process? (1) Probability of substitution depends on the branch length, or the time to the common ancestor of the two proteins. (2) Substitutions are chosen randomly in accordance with the amino acid substitution matrix. Due to various physical and structural constraints that can be imposed on the particular position in the sequence, each column of the MSA would have a unique substitution matrix.

In contrast to this, SDRs (ideally) would have different characteristics. (1) Within orthologous group SDR is conserved regardless of how far the sequences have diverged. (2) Between paralogs, SDRs would have to be different however phylogenetically close the sequences are. Numerous ways have been proposed to estimate the influence of phylogeny on a particular position. Here we describe a Monte Carlo approach developed in our group.

3.2.6.1. Monte Carlo

The purpose of our Monte Carlo (MC) algorithm is to generate MSAs that would have tree topology and amino acid composition (entropy) inside each MSA column identical to original alignment while otherwise being random (**Fig. 18.2**). By generating a number of such alignments, we can estimate the influence of phylogeny and physicochemical constraints on a sequence position and therefore compute expected mutual information and statistical significance of the mutual information in each position for original MSA (**Fig. 18.3**).

We do not explicitly construct and use a phylogenetic tree. Instead, we utilize a matrix M_0^{mn} that contains mean pairwise sequence identity between proteins of group m and group n , i.e.,

$$M^{mn} = \sum_{i \in m, j \in n} D_{ij} / N_n N_m,$$

where D_{ij} is the sequence identity between sequences i and j , and N_n is the number of proteins in group n . Such a sequence identity matrix M_0^{mn} is an approximation of the phylogenetic tree (this is similar to a tree obtained by the distance matrix method, e.g., by neighbor joining). The MC algorithm aims to generate sequences that have a pairwise identity matrix M_{mc}^{mn} as close to M_0^{mn} as possible. Simultaneously, it aims to produce an MSA that has

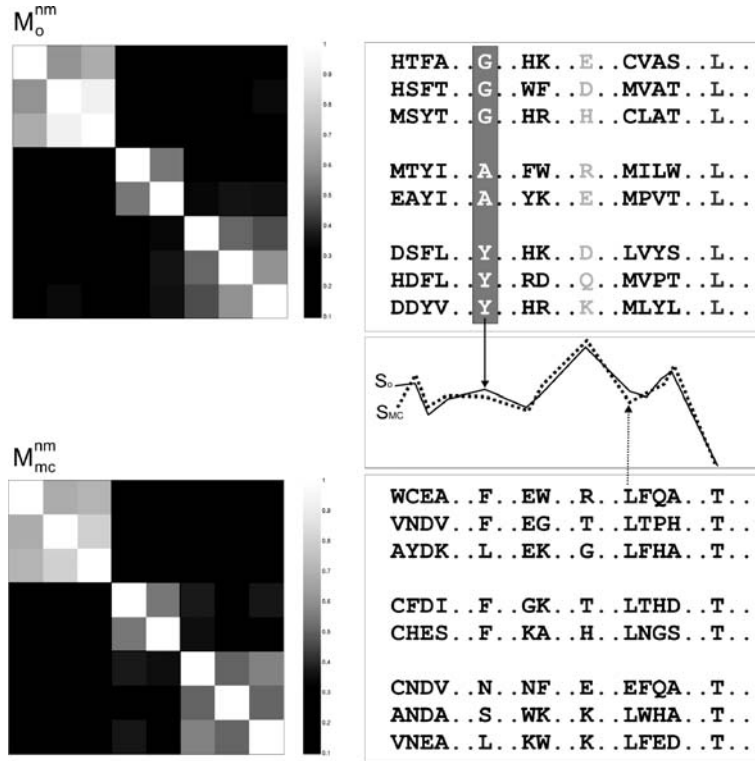


Fig. 18.2. A demonstration of MC procedure. Given an MSA (*top right*), the algorithm finds a random MSA (*bottom right*) that has the pairwise identity matrix M_{mc}^{nm} as close to the original M_0^{nm} as possible and entropy in each position S_{mc}^i as close to the entropy of the original alignment S_0^i as possible.

conservation in each column (as measured by entropy S_{mc}^i) close to the entropy in the original alignment S_0^i . Entropy of a column i of the MSA is given by

$$S^i = - \sum_{x=1, \dots, 20} f_i(x) \log(f_i(x)).$$

Taken together, these constrains lead to the following function to be minimized in MC simulations

$$F(S, M) = \alpha \sum_{\substack{n, m=1, \\ n \neq m}}^Y (M_0^{nm} - M_{mc}^{nm})^2 + \beta \sum_{n=1}^Y (M_0^{nn} - M_{mc}^{nn})^2 + \gamma \sum_{i=1}^L (S_0^i - S_{mc}^i)^2, \quad [1]$$

where $i=1..L$ is the index of position in MSA, α , β , and γ are empiric weights. Parameters α , β , and γ are chosen in such way that corresponding terms in Eq. [1] are roughly (same order of magnitude) equal when F converges.

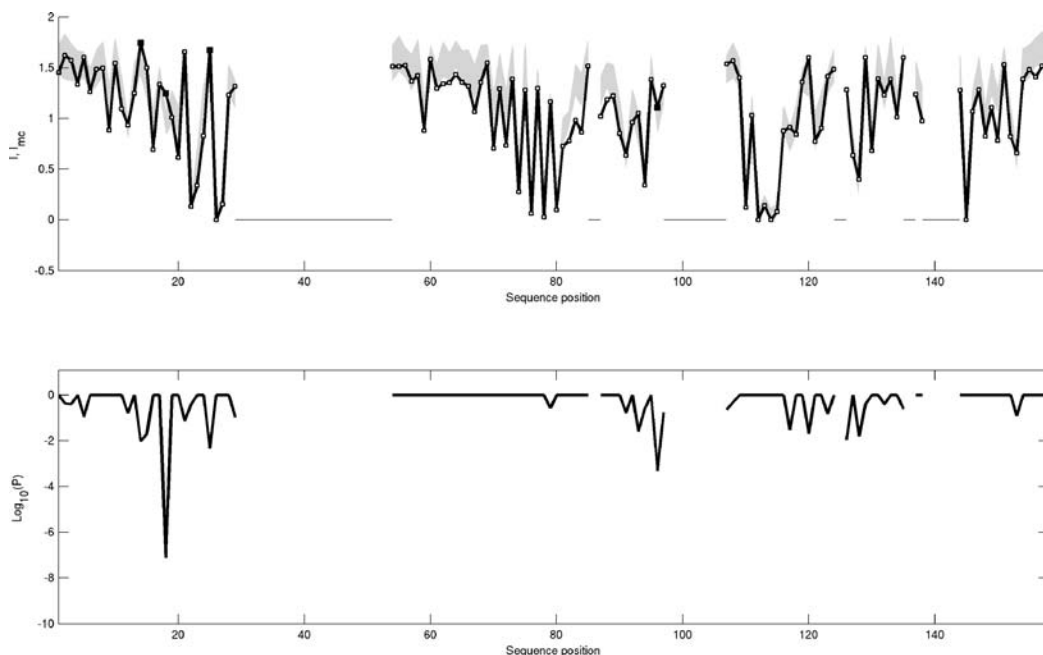


Fig. 18.3. (a) Observed mutual information I (solid black line) and expected from MC MSAs. The expected MC mutual information I_{mc} corresponds to the central points of the thick gray line with the width of the line corresponding to 4σ ($I_{mc} \pm 2\sigma$) at each position of the alignment. Positions with P -value $< 10^{-2}$ are marked with black squares. (b) Probabilities of observed I in position calculated from MC MSAs.

We use standard Metropolis MC (23) with the energy given by function F and moves that consist of swapping two randomly chosen letters in the MSA or replacing a randomly chosen residue with another one (chosen uniformly at random). All moves are made on positions that carry amino acids and exclude gaps, i.e., gaps in the MC-generated alignment are in the same position as in the original one. The algorithm starts with a MSA obtained from original MSA by shuffling each column. The temperature of the Metropolis procedure T is chosen as the maximum temperature that allows F to converge.

3.2.6.2. Calculating Statistical Significance and Predicting SDRs

The mean expected mutual information I_{mc} and variance σ are derived from a number of MC alignments. The probability to observe mutual information greater than or equal to I_o in the alignment is calculated from Gaussian cumulative distribution function:

$$P(I_o) = \int_{I_o}^{+\infty} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(I - I_{mc})^2}{2\sigma^2}\right) dI \quad [2]$$

The positions that have probability P below a certain cutoff are putative SDRs. The cutoff is usually chosen based on the effective length of the alignment (length of the alignment minus number of positions with >20% of gaps) and is 0.02–0.001 for a typical alignment.

3.3. Locating CERs in Double Alignments

3.3.1. Co-Evolving Residues

Groupings by specificity and orthology may be hard to obtain and/or hard to automate. In such cases, we identify co-evolving residues (CERs, also known as “correlated mutations”). Our method to identify CERs does not require grouping by specificity, it rather requires knowledge of binding partners. We take advantage of known binding interactions between pairs of homologous proteins to identify CERs. For example, bacterial histidine-kinases (HKs) are known to bind corresponding response regulators (RRs). Genomic information allows us to find interacting HK-RR pairs and to construct a “dual alignment” needed to identify CERs (*see Fig. 18.4*).

To identify residues that display correlated evolution, we also use mutual information as a measure of their correlation. For example, residues in dark gray columns on **Fig. 18.4** are highly correlated between two sets of homologous proteins, but residues in light gray columns (conserved residues) and light gray residues (highly variable residues) are not. To quantify the extent of their correlated behavior we use mutual information. Given two multiple sequence alignments (MSAs), each corresponding to one of two sets of homologous proteins and arranged in such a way that interacting proteins are located in the same row of the alignments, mutual information is computed as

$$I^{ij} = \sum_{x,y=1,\dots,20} f_{ij}(x,y) \log \frac{f_{ij}(x,y)}{f_i(x)f_j(y)},$$

where $f_{ij}(x,y)$ is joint distribution of residues x and y in positions i and j of the two proteins, $f_i(x)$ is the marginal frequency of x in column i . Gaps are not counted and distributions are properly normalized (as described above for SDRs).

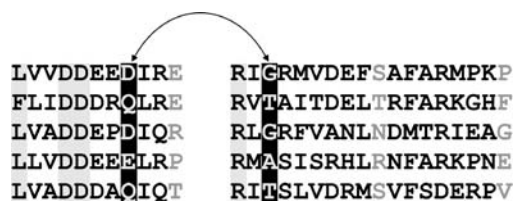


Fig. 18.4. Two hypothetical MSAs that have two co-evolving positions. The pair of CERs is shown in white on dark gray background, conserved residue in dark gray on light gray background and variable residues in light gray. While positions with variable residues do not correlate with each other, certain amino acids in a CER position of one alignment always correspond to certain amino acids of another in the ideal case.

3.3.2. Estimation of Statistic Significance of CERs

Similar to SDR analysis, identification of CERs is prone to errors due to a bias introduced by other evolutionary signals: structural signal, a signal from functional amino acids, and a signal originating in the phylogenetic history of the two families. Often interacting proteins are co-evolving and as a result may have similar topology of phylogenetic trees. Such shared history can make all positions in these proteins to be somewhat correlated. Often CERs are sought within one protein where all positions share the same phylogenetic tree.

To take these factors into account we conduct MC procedure similar to the one described above. For CERs, we set all sequence group sizes to 1 and $\beta = 0$. Thus sequence identity between individual random sequences is optimized until the exact tree topology of original set is reached.

Similar to SDRs, this is the crucial step in our method of CER identification. High correlation does not necessarily imply co-evolution and most position pairs with high mutual information are discarded by statistical significance procedure, because MC can reproduce the values of mutual information given a phylogenetic history and entropy at each position.

3.3.2.1. Computing Statistical Significance

The probability P for value I_o^{ij} to occur for two given columns i, j in two MSAs is computed using formula (2), where expected value and variance of I_{mc}^{ij} is derived from MC MSAs.

A position i is defined as a CER in the first MSA if it occurs at least in one position pair i, j with the probability P below cutoff. Likewise, position j is defined as CER in the second MSA. The cutoff for the probability P is usually chosen as reverse proportional to the product of both MSA's effective lengths.

4. Results and Discussion

4.1. Bacterial Two- Component System

Signal transduction in bacteria often involves two-component systems. The system consists of two parts: sensor histidine kinase (*HK*) and a response regulator protein. Upon receiving an external signal (such as a small molecule, e.g. fumarate), the *ATPase* domain of an active homodimerized HK autophosphorylates the second monomer in conserved His residue, which is usually located in the dimerization domain (DD) (24) (**Fig. 18.6**).

Upon binding a response regulator protein, the phosphoryl group is then transferred to its conserved Asp residue. The response regulator protein usually consists of two domains: phosphate-accepting response regulator domain (*RR*) and effector domain (**Fig. 18.5**). The effector domain is typically one of a broad variety of DNA-binding domains that regulates the transcription of response genes.

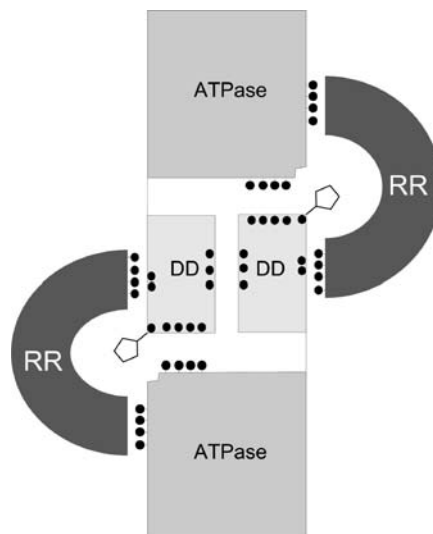


Fig. 18.5. A scheme of interactions between domains in two-component systems. After dimerization via the DD and activation by extraneous signal, the ATPase autophosphorylates the DD by the conserved His residue (shown as a pentagon on the DD). Active dimer is then bound by RR that accepts phosphoryl group from the DD. SDRs and CERs on domain interfaces are shown as *black circles*.

Despite a large number of different two-component systems in bacteria and a significant sequence similarity between their components, the signaling is very specific: there is virtually no cross-talk between different signal transduction pathways *in vivo* (19). We applied our method to locate the residues that determine this specificity of HK-RR and HK-ATPase recognition.

4.1.1. Results

Predicted SDRs and CERs are schematically summarized on **Fig. 18.5**. Detailed predictions are presented in **Table 18.1** and on (**Fig 18.6a**) for the RR domain; in **Table 18.2** and on **Fig. 18.6b** for the ATPase domain; and in **Table 18.3** and **Fig. 18.6b** for the DD. Here we provide structural rationale for the role of the identified residues in the function of two-component system and compare our predictions with experimental results.

4.1.2. SDRs

RR. Five SDRs have been found on RR domain (**Table 18.1**). Three residues are located on the interface with DD, which is specifically recognized by RR. Two of these the SDRs are close to the conserved Asp1454, where the phosphate from DD is transferred (**Fig. 18.6a**).

Two SDRs located on the $\alpha 4$ helix of the RR do not appear to interact with DD or ATPase. However, in some RR domains the $\alpha 4$ helix has been shown to swing approximately 90° (25), roughly into the plane of interaction with HK (*see Section 4.1.4* below). The $\alpha 4$ helix was also shown to serve in the dimerization of the active (phosphorylated) RR (26), thus suggesting a role of the identified SDRs on the $\alpha 4$ helix in providing specificity of RR dimerization.

Table 18.1
SDRs and CERs in the RR domain

MSA position	22	73	112	115	117	122	129
1f51	ILE1415H	LYS1456H	TYR1484H	LEU1487H	MET1489H	SER1493H	LEU1499H
1kgs	LEU14A	MSE55A	LEU83A	VAL86A	TYR88A	GLY92A	ASP98A
Ala-scan	+	+	+	+	-	-	NA
SDR	0.006	0.0005	0.01	0.003	0.001	0.003	0.4
RR/DD	7E-06	9E-12	9E-13	0.01	9E-06	2E-06	8E-06
RR/HATPase	5E-09	4E-07	6E-17	0.03	3E-08	3E-06	5E-08

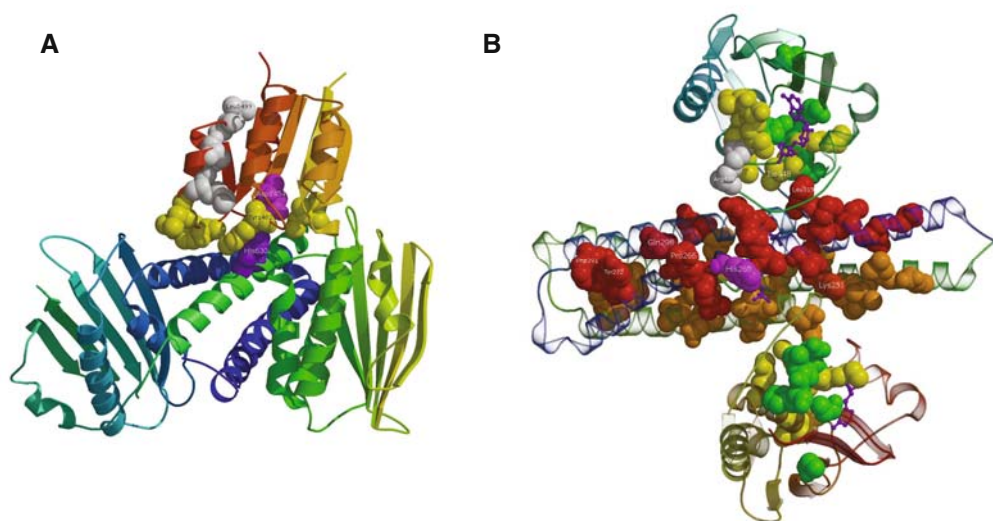


Fig. 18.6. Structural localization of putative SDRs and CERs in two-component system domains. (a) RR SpoOF (*red-brown ribbon*) bound to structural analog of the DD in SpoOB protein. The conserved His is shown in *purple*, the conserved Asp in RR in *magenta*. SDRs and CERs are shown in *yellow* or, when located on the α_4 helix, in *white* (PDB entry 1F51). (b) The non-catalytic conformation of HK homodimer. ADP is shown as a *purple wireframe*, the phosphate-accepting conserved His residue in *magenta spacefill*. SDRs and CERs on the ATPase are shown in *yellow*, or in *white* if located on the unresolved ATP-lid loop that was superimposed from PhoQ kinase (PDB entry 1ID0), or in *green* in the RR-specific CERs side patch. SDRs and CERs on the DD are shown in *red* on one homodimer and *orange* on another (PDB entry 2C2A). (see Color Plate)

ATPase. All three SDRs are located on the interface with the DD. Two SDRs are located close to the active site (i.e., conserved His260 in DD), strongly suggesting their role in specific ATPase-DD recognition.

DD Two SDRs on the DD (Table 18.3, Fig. 18.6) are part of the specific dimerization interface. Lys251 is known to be important for dimerization (27). Gln298 forms a contact with Pro265 (see Section 4.1.3 below), which has been shown to help exposing phosphate accepting His260 residue (27). Gln298 may also be involved in recognition of RR, suggesting its role as a key residue in controlling the access to His260 by ATPase (upon dimerization) by RR for signal transduction.

In summary, structural analysis demonstrates that the identified SDRs are consistent with the mechanism of activation and recognition in two-component proteins, pinpointing recognition to a small group of residues in the two-component system.

4.1.3. CERs

Here we take advantage of known pairwise interaction between the RR, DD and ATPase, derived from their genomic proximity and identify residues exhibiting coordinated evolution in interacting domains. These results are summarized on Figs. 18.5 and 18.6.

Table 18.2
SDRs and CERs in the ATPase domain

MSA position	14	18	25	28	29	96	97	108	117	119	126
2c2a	-	GLN372A	ASN379A	VAL382A	LYS383A	ARG430A	VAL431A	‡439A	ILE448A	LYS450A	HIS456A
SDR	0.01	8E-08	0.005	1	0.1	0.0005	0.2	0.5	0.03	1	0.01
HATPase/DD	0.007	2E-06	1E-06	9E-09	9E-07	9E-08	2E-07	7E-09	3E-16	9E-07	9E-07

Table 18.3
SDRs and CERs in the DD

MSA Pos	1	5	8	11	12	13	17	18	24
2c2a	GLU240A	LEU244A	ILE247A	MET250A	LYS251A	THR252A	ALA256A	ASN257A	ARG263A
Ijoy	MET223A	VAL227A	LEU230A	ASP233A	ARG234A	THR235A	ALA239A	GLY240A	ARG246A
SDR	0.1	0.4	0.02	0.3	0.02	0.2	0.4	0.4	0.06
DD-ATPase	0.002	3E-06	7E-09	5E-06	8E-06	1E-07	5E-06	1E-06	2E-06
DD-RR	1E-07	6E-08	9E-13	0.002	9E-12	1E-09	0.0002	2E-05	3E-07
MSA Pos	25	26	33	79	86	91	96	98	103
2c2a	THR264A	PRO265A	TYR272A	PHE291A	GLN298A	GLU303A	GLU308A	LEU310A	LEU315A
Ijoy	THR247A	PRO248A	ALA255A	LEU266A	ASP273A	ASN278A	GLN283A	ILE285A	-
SDR	0.04	0.1	0.2	0.1	0.02	0.1	0.2	0.05	1
DD-ATPase	0.001	3E-16	0.002	0.0004	0.001	4E-06	2E-06	1E-05	4E-06
DD-RR	1E-05	1E-09	2E-06	8E-06	8E-05	5E-05	0.02	2E-07	9E-05

RR-DD. CERs in the RR domain include all SDRs and two additional residues: Tyr1482 and Leu1499 (for the RR, residues are numbered according to the PDB entry 1F51). Tyr1482 forms a tight contact with the DD in the immediate neighborhood of conserved His. Leu1499 is located on top of the $\alpha 4$ helix and may participate in RR dimerization.

In contrast the DD contains lots of co-evolving residues. Nine residues are located on the surface of the domain facing and interacting with RR, seven of them in immediate neighborhood of conserved His260. Pro265 and Glu298 (an SDR) form a contact and may be recognized by the RR. Two RR-specific CERs, Tyr272 and Phe291, form a contact and are likely to interact exclusively with the RR. The other patch of residues is located closer to the N-terminus and consists of known dimerization residues (27); some of them may also be in contact with ATPase domain in the non-catalytic conformation.

DD-ATPase. When we examined residues in the DD that co-evolve with the ATPase domain, we found nearly the same set of CERs as those co-evolved with the RR. Some of these DD residues are in contact with their counterparts in the ATPase domain (e.g. Leu315); some are located close to the phosphorylated histidine and are likely to interact with both ATPase and RR domains residues during phosphate transfer events.

This suggests that mostly the same residues in the DD are responsible for recognition of the ATPase domain and recognition of the RR.

The ATPase domain residues that co-evolve with the DD contact the DD and/or surround the ATP (**Fig. 18.6b**, ATP shown in purple wireframe). They include all three SDRs. Interestingly, one of the residues (Ala439) was mapped on a disordered mobile loop (ATP-lid) in 2C2A X-ray structure, which is especially close to the DD. Two CERs, Ile448 in the ATPase and Leu315 in the DD, are the part of the labile hydrophobic interdomain interface that controls release of the ATPase for autophosphorylation when it receives a signal from the sensor domain. Mutations in these residues most significantly alter the degree of the autophosphorylation (27). It should be noted that, on crystal structure 2C2A, the ATPase domain is in the inactive conformation and the transfer of the phosphoryl group is not possible (*see Section 4.1.4* below).

RR-ATPase. Interestingly, the RR-ATPase CERs in RR domain are residues that contact DD or reside on the $\alpha 4$ helix, coinciding with RR-DD's CERs and RR's SDRs. However, the corresponding CERs in ATPase are different: although most of them also correspond to the ATPase-DD interface and ATPase-DD phosphate transfer, there is a distinct patch of residues located on the putative contact surface of the ATPase and RR (green residues on **Figs. 18.6b** and **18.7b**).

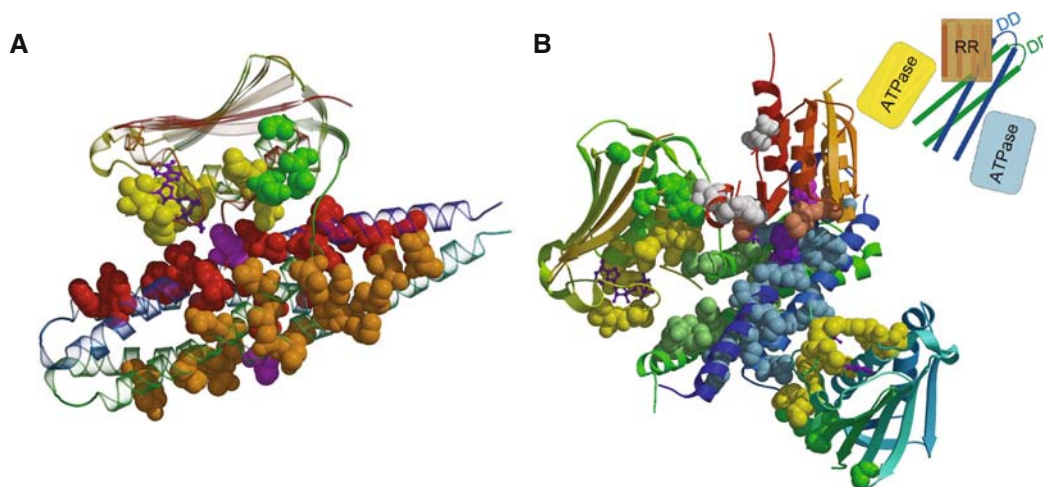


Fig. 18.7. Localization of putative SDRs and CERs on computationally obtained models (models provided by Marina et al (27)). (a) HK in the active conformation, the ATPase is docked on the DD so that transfer of the phosphoryl group is possible. SDRs and CERs on the ATPase domain are shown in *yellow* or *green* when located in the RR-specific CERs side patch. SDRs and CERs on the DD are shown in *red* on one homodimer and *orange* on another. (b) Spo0F computationally docked on HK and subsequently superimposed with RR from OmpR. RR (*brown-red ribbon*) (PDB entry *1KGS*) with its $\alpha 4$ helix swung $\sim 90^\circ$: the phosphorylated Asp in the RR is shown in *magenta*, SDR and CERs are shown in *light red* or, when located the $\alpha 4$ helix in *white*. DD (*dark blue and dark green ribbon*): SDRs and CERs are shown in *light blue* on one dimer and in *light green* on another. ATPase (*yellow-green ribbon on the left and light-blue on the right*): the colors are the same as in (a). (see Color Plate)

4.1.4. Modeled Structures

Unfortunately, the structure of HK in its active autocatalytic conformation has not been solved. The transfer of phosphoryl group is not possible on the available structure of HK depicted on **Fig. 18.6b**. The structure of the RR–DD complex has not been solved either. Although the structure of the RR-phosphotransferase complex Spo0F–Spo0B from *Bacillus subtilis* has been solved, the Spo0B phosphotransfer domain that structurally corresponds to DD is not homologous to the DD family, making mapping of predicted results unreliable and difficult. Likewise, the Spo0B catalytic domain is not homologous to any sequence in our ATPase dataset.

To solve these problems Marina et al. have modeled the HK in the active autocatalytic conformation and the RR (Spo0F) bound to HK (27). **Figure 18.7** presents predicted SDRs and CERs mapped onto the modeled structures that were kindly provided by Marina et al.

Our results agree well with the modeled structure. Predicted residues that surround ATP in ATPase domain make contacts with SDRs and CERs that surround conserved His on the DD (**Fig. 18.7a**). There is a distinct patch of ATPase-RR CERs on the ATPase that does not contact the DD. It appears to be close to the RR when it is bound to the DD (**Fig. 18.7b**). Moreover, it contacts the $\alpha 4$ helix and some of the predict residues on it when

$\alpha 4$ is swung $\sim 90^\circ$ as has been observed in OmpR homolog (25) (**Fig. 18.7b**). Specificity residues of the RR, except for the ones that contact the ATPase, are likely to interact with their counterparts on the DD. Interestingly, one of them (Lys1456) contacts the conserved His on the DD, although the precision of the computationally docked RR (Spo0F) makes it difficult to resolve this interaction.

4.1.4.1. Comparison with Experiment: Ala-Scan of RR

The predictions agree well with the available experimental data. According to the alanin scan of Spo0F (28) (*see Table 18.1*), 4 out of six predicted residues with available Ala-mutants indeed affect the recognition of the HK by the RR. Two residues located on the $\alpha 4$ helix were predicted, but did not affect phenotype in Ala mutagenesis (marked white on **Fig. 18.7a**). They do not contact the phosphotransfer domain (DD analog) in Spo0B-Spo0F, but are very likely to contact ATPase and/or participate in the RR protein homodimerization as discussed above.

This demonstrates that while finding specificity-determining residues in a broad class of two-component proteins is instructive, using more narrow groups of specific two-component proteins can provide predictions suited for these groups. While the same interfaces and patches are used for specific recognition through a broad class of two-component proteins, individual subclasses of the two-component system may use different residues in these patches to provide specificity.

4.2. Discussion of Two-Component Systems Putative SDRs and CERs

In conclusion, both methods – correlated mutations and SDRs, found two largely overlapping groups of residues – the ones that surround sites involved in the phosphotransfer and the ones located on the interfaces of the interacting domains. Somewhat unexpectedly, the method detected specificity residues on the $\alpha 4$ helix of the RR that have not been known to contact with HK. These residues may interact with the corresponding patch of CERs on ATPase domain or participate in activated RR dimerization.

The results point to the highly correlated evolution of the binding interfaces between all three domains. For instance, the surface residues of the RR that interact with the DD and residues of the ATPase that interact with the DD show a pattern of co-evolution, though no direct contact between these residues is observed in the available structures. We can see two possible explanations of co-evolution among apparently non-interacting domains. First is that some residues in the DD, especially the ones that are close to the phosphorylated His, contact both domains during different stages of the phosphate transfer reaction. The mutations in these residues must cause compensatory mutations in both the RR and the ATPase, thus making these domains

co-evolve. Likewise, a mutation in the His contact patch in RR or ATPase can cause compensatory mutation(s) in the DD and, in turn, in the third domain.

The second reason is that such correlation may be caused by “*negative design*” constraints. The concept of negative design has attracted lots of attention in the protein engineering community. Negative design assumes that not binding certain targets or not folding into certain conformations is as important for a protein as binding its targets and folding into its native conformation. In the context of the two-component (and broadly, signaling) proteins, negative design would be required to minimize a cross-talk between different signaling pathways. Thus, evolution of the two-component system can be driven by need to minimize affinity between proteins of different pathways, rather than a need to increase affinity within pathways. Negative design can lead to a constant accumulation of mutations that reduce cross-pathway affinity and compensatory mutations that keep affinity within a pathway at the required level. Since negative design does not target any particular interface or domain, it will lead to global co-evolution of protein interfaces within a pathway, even when such interfaces do not interact directly. Negative design can also cause specificity within a pathway to get distributed across several interfaces, consistently with our observation. This can explain, for example, why DD dimerization residues and DD residues that exclusively contact the ATPase are correlated with RR residues. This is also likely to be the reason for the strong overlap between SDRs and CERs.

Another interesting observation is that the DD is the fastest evolving domain among three (Table 18.4), which makes SDRs detection particularly difficult. Indeed, many DDs do not even share any detectable sequence similarity. This may be due to the simpler function, lack of catalytic activity, and, consequently, the smaller number of structural constraints imposed on the DD. An alternative explanation is that negative design targets homodimerization of HKs as the first and important step controlling the specificity of the two-component systems. Since many HKs are membrane-bound via their transmembrane and sensor domains,

Table 18.4
Average pairwise identities in three domain families of two-component system

	DD	RR	ATPase
Average identity	0.19	0.25	0.25
Average identity within orthologous groups	0.48	0.49	0.43

the concentration of different two-component HKs on the membrane can be very high, increasing the probability of non-cognate heterodimerization. This may lead to high selective pressure on the specialization of DDs and hence their rapid evolution.

4.3. Aminoacyl-tRNA Synthetases

The aminoacyl-tRNA synthetases are the enzymes involved in the core process of the cell – the translation of genetic information from messenger RNAs to proteins. Such encoding, i.e., implementation of the genetic code, relies on the correct assignment of nucleotide triplets to amino acids through correct “charging” of tRNAs with corresponding amino acids. The aminoacyl-tRNA synthetases attach an amino acid molecule to the corresponding tRNA in a highly specific way. Even small number of errors in this process would be harmful for a biological system, as it would result in numerous errors in protein synthesis.

The aminoacyl-tRNA synthetases (aaRSs) represent a diverse group of enzymes varying in structure and evolutionary history. They are roughly divided into two classes: I and II. Classes are then subdivided into smaller subclasses a, b, and so on, to reflect the sequence and structural similarity between different synthetases.

Here we focus on aaRSs belonging to the class Ia, namely the enzymes corresponding to leucine, isoleucine, methionine, and valine and aim to understand the mechanism that allows the enzyme to recognize the correct tRNA and amino acids for charging.

We collected catalytic domains of class Ia aaRSs from 20 bacterial genomes using tRNA-synt_I PFAM HMM profile. It should be noted that this profile does not include major part of the tRNA recognition arm. Then we manually grouped sequences according to their amino-acid specificity into four groups and applied the SDR method. We then paired each of the aaRSs with the corresponding tRNAs and applied CER analysis to the double-alignment of protein and RNA. Only one tRNA for each aaRS was used in CER analysis to avoid bias toward the amino acids encoded with more codons. In CER statistical evaluation, only the protein part of the double alignment was subjected of MC simulations, while the set of tRNAs remained unchanged.

Predicted SDRs and CERs are presented on (Fig. 18.8) and in Table 18.5, demonstrating a very good agreement between both methods.

Next, we used available structures of aaRS-tRNA complexes to validate the SDRs and CERs. Examination of the structures of class Ia aaRS shows that the side chain of a substrate amino acid fits into a hydrophobic pocket. Surprisingly, most of the residues of this pocket are the same in Val-, Ile-, and LeuRS, making it hard to understand how the specificity of amino acid recognition is achieved. Detected SDRs provide a mechanism for specific

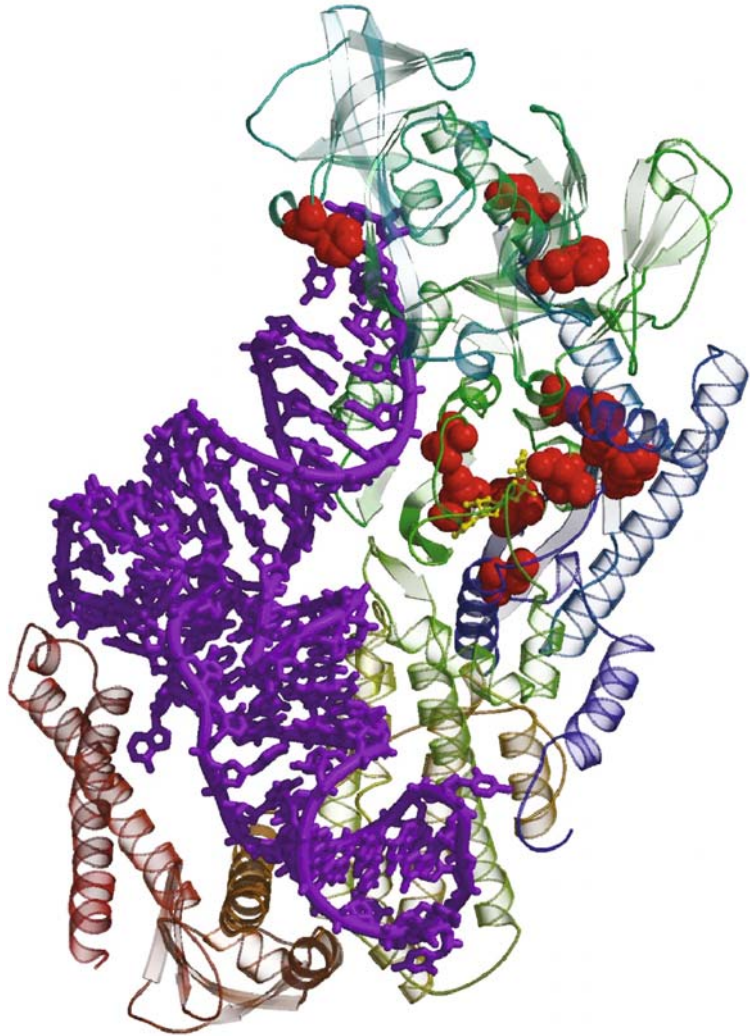


Fig. 18.8. Valine aminoacyl-tRNA synthetase (PDB entry 1GAX). The tRNA is shown as a *purple* wireframe structure, SDRs and CERs are *red* balls, and amino acid (valyl-adenylate analog) is in *yellow* wireframe. (see Color Plate)

recognition. Only two amino acids in this hydrophobic pocket: Asn44 and Pro41 are variable among aaRS and both have been detected as SDRs. They form immediate contact with the side chain of valine in ValRS (29)(PDB:1GAX). While several other SDRs (Val496, Leu485, Ser458) do not directly interact with the substrate amino acid, they form a layer around the conserved hydrophobic pocket and are likely to modulate its shape in aaRSs of different specificity, thus contributing to specific recognition. This structure also made it clear that one SDR is involved in the recognition of tRNA: Phe264 in ValRS (Trp227 in IleRS) forms

Table 18.5
SDRs and CERs in class Ia aaRSs

MSA position	38	40	43	58	83	149	408	543	547	693	701	740	751
<i>I_{gax}</i>	PHE39	PRO41	ASN44	ASN57	HIS82	PHE143	PHE264	ILE355	TRP359	SER458	THR466	LEU485	VAL496
SDR	5E-05	4E-05	0.002	1E-06	2E-06	5E-05	1E-05	7E-07	0.0003	0.02	3E-07	0.008	2E-10
aaRS-tRNA	5E-05	1E-07	4E-09	1E-06	1E-06	3E-06	1E-05	7E-07	3E-11	6E-08	2E-07	7E-08	2E-10

strong Van der Waals interaction with adenine-76 (29, 30). A76 was experimentally shown to be essential for correct recognition of tRNA by aaRS (31).

5. Summary and Conclusions

Related, homologous, proteins often perform the same general biochemical function. At the same time, different homologous variants of the protein can be utilized in the cell in similar ways but in different contexts. Often there are no physical barriers such as membranes to separate homologous proteins, preventing proteins from interacting with wrong, yet similar, targets. This necessitates highly specific recognition to be performed by the protein in order to bind its substrate and/or interacting partners.

In many cases, due to structural properties imposed by the general function common to the protein family, the specificity determinants are also constrained to certain locations on the 3D-structure and in the amino acid sequence of the protein.

Here we presented two techniques that allow the identification of amino acids involved in specific recognition. The first one relies on sets of orthologs from close species as proteins of the same specificity and contrasts them with paralogs that have different specificities. Another technique uses pairs of known interacting proteins to identify residues exhibiting coordinated evolution. Both methods rely solely on sequence information, allowing us to use available structures for validation. The techniques are very different in their assumptions and input information, so their predictions can be treated as independent pieces of evidence for the functional role of the identified residues.

These techniques allowed us to study the origin of specificity in two bacterial systems: two-component signaling and tRNA synthesis. Strikingly, the two very different techniques yield similar sets of amino acids. While the two-component system showed a very complicated pattern of specificity distributed across interfaces between three participating domains, the specificity of tRNA synthetases seem to be focused in a few specific locations on the protein–amino acid and protein–tRNA interfaces.

Our predictions generate a set of experimentally testable hypothesis about the role of specific amino acids in molecular recognition. Mutations or ligand binding at these positions can interfere with binding, thus disrupting the process. Thus, identified pockets of specificity can be used to direct drug design efforts. Moreover, if a small set of SDRs provides specificity, one can experimentally swap SDRs in two pathways in order to swap specificities and re-wire the signaling pathways.

References

1. Kopke Salinas R, Folkers GE, Bonvin AM, Das D, Boelens R, Kaptein R. Altered specificity in DNA binding by the lac repressor: a mutant lac headpiece that mimics the gal repressor. *ChemBiochem* 2005;6:1628–37.
2. de Prat Gay G, Duckworth HW, Fersht AR. Modification of the amino acid specificity of tyrosyl-tRNA synthetase by protein engineering. *FEBS Lett* 1993;318:167–71.
3. Livingstone CD, Barton GJ. Protein sequence alignments: a strategy for the hierarchical analysis of residue conservation. *Comput Appl Biosci* 1993;9:745–56.
4. Lichtarge O, Bourne HR, Cohen FE. An evolutionary trace method defines binding surfaces common to protein families. *J Mol Biol* 1996;257:342–58.
5. Hannenhalli SS, Russell RB. Analysis and prediction of functional sub-types from protein sequence alignments. *J Mol Biol* 2000;303:61–76.
6. Kalinina OV, Mironov AA, Gelfand MS, Rakhmaninova AB. Automated selection of positions determining functional specificity of proteins by comparative analysis of orthologous groups in protein families. *Protein Sci* 2004;13:443–56.
7. Mirny LA, Gelfand MS. Using orthologous and paralogous proteins to identify specificity-determining residues in bacterial transcription factors. *J Mol Biol* 2002;321:7–20.
8. Pei J, Cai W, Kinch LN, Grishin NV. Prediction of functional specificity determinants from protein sequences using log-likelihood ratios. *Bioinformatics* 2006;22:164–71.
9. Vernet T, Tessier DC, Khouri HE, Altschuh D. Correlation of co-ordinated amino acid changes at the two-domain interface of cysteine proteases with protein stability. *J Mol Biol* 1992;224:501–9.
10. Gobel U, Sander C, Schneider R, Valencia A. Correlated mutations and residue contacts in proteins. *Proteins* 1994;18:309–17.
11. Tress M, de Juan D, Grana O, Gomez MJ, Gomez-Puertas P, Gonzalez JM, Lopez G, Valencia A. Scoring docking models with evolutionary information. *Proteins* 2005;60:275–80.
12. Shindyalov IN, Kolchanov NA, Sander C. Can three-dimensional contacts in protein structures be predicted by analysis of correlated mutations? *Protein Eng* 1994;7:349–58.
13. Pollock DD, Taylor WR, Goldman N. Co-evolving protein residues: maximum likelihood identification and relationship to structure. *J Mol Biol* 1999;287:187–98.
14. Fariselli P, Casadio R. A neural network based predictor of residue contacts in proteins. *Protein Eng* 1999;12:15–21.
15. Yu GX, Park BH, Chandramohan P, Munavalli R, Geist A, Samatova NF. In silico discovery of enzyme-substrate specificity-determining residue clusters. *J Mol Biol* 2005;352:1105–17.
16. Fitch WM. Distinguishing homologous from analogous proteins. *Syst Zool* 1970;19:99–113.
17. Fitch WM. Homology a personal view on some of the problems. *Trends Genet* 2000;16:227–31.
18. Blattner FR, Plunkett G, 3rd, Bloch CA, Perna NT, Burland V, Riley M, Collado-Vides J, Glasner JD, Rode CK, Mayhew GF, Gregor J, Davis NW, Kirkpatrick HA, Goeden MA, Rose DJ, Mau B, Shao Y. The complete genome sequence of *Escherichia coli* K-12. *Science* 1997;277:1453–74.
19. Skerker JM, Prasol MS, Perchuk BS, Biondi EG, Laub MT. Two-component signal transduction pathways regulating growth and cell cycle progression in a bacterium: a system-level analysis. *PLoS Biol* 2005;3:e334.
20. Katoh K, Kuma K, Toh H, Miyata T. MAFFT version 5: improvement in accuracy of multiple sequence alignment. *Nucleic Acids Res* 2005;33:511–8.
21. Edgar RC. MUSCLE: multiple sequence alignment with high accuracy and high throughput. *Nucleic Acids Res* 2004;32:1792–7.
22. Altschul SF, Madden TL, Schaffer AA, Zhang J, Zhang Z, Miller W, Lipman DJ. Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Res* 1997;25:3389–402.
23. Metropolis N, Rosenbluth AW, Rosenbluth MN, Teller AH, Teller EJ. Equation-of-state calculations by fast computing machines. *J Chem Phys* 1953;21:1087–92.
24. Stock AM, Robinson VL, Goudreau PN. Two-component signal transduction. *Annu Rev Biochem* 2000;69:183–215.
25. Buckler DR, Zhou Y, Stock AM. Evidence of intradomain and interdomain flexibility in an *OmpR*/*PhoB* homolog from *Thermotoga maritima*. *Structure* 2002;10:153–64.

26. Birck C, Mourey L, Gouet P, Fabry B, Schumacher J, Rousseau P, Kahn D, Samama JP. Conformational changes induced by phosphorylation of the FixJ receiver domain. *Structure* 1999;7:1505–15.
27. Marina A, Waldburger CD, Hendrickson WA. Structure of the entire cytoplasmic portion of a sensor histidine-kinase protein. *Embo J* 2005;24:4247–59.
28. Tzeng YL, Hoch JA. Molecular recognition in signal transduction: the interaction surfaces of the Spo0F response regulator with its cognate phosphorelay proteins revealed by alanine scanning mutagenesis. *J Mol Biol* 1997; 272: 200–12.
29. Fukai S, Nureki O, Sekine S, Shimada A, Tao J, Vassilyev DG, Yokoyama S. Structural basis for double-sieve discrimination of L-valine from L-isoleucine and L-threonine by the complex of tRNA(Val) and valyl-tRNA synthetase. *Cell* 2000;103:793–803.
30. Silvan LF, Wang J, Steitz TA. Insights into editing from an ile-tRNA synthetase structure with tRNA^{ile} and mupirocin. *Science* 1999;285:1074–7.
31. Tamura K, Nameki N, Hasegawa T, Shimizu M, Himeno H. Role of the CCA terminal sequence of tRNA(Val) in aminoacylation with valyl-tRNA synthetase. *J Biol Chem* 1994;269:22173–7.

Chapter 19

Connecting Protein Interaction Data, Mutations, and Disease Using Bioinformatics

Jake Y. Chen, Eunseog Youn, and Sean D. Mooney

Abstract

Understanding how mutations lead to changes in protein function and/or protein interaction is critical to understanding the molecular causes of clinical phenotypes. In this method, we present a path toward integration of protein interaction data and mutation data and then demonstrate the identification of a subset of proteins and interactions that are important to a particular disease. We then build a statistical model of disease mutations in this disease-associated subset of proteins, and visualize these results. Using Alzheimer's disease (AD) as case implementation, we find that we are able to identify a subset of proteins involved in AD and discriminate disease-associated mutations from SNPs in these proteins with 83% accuracy. As the molecular causes of disease become more understood, models such as these will be useful for identifying candidate variants most likely to be causative.

Key words: Protein interaction, SNP, mutation, bioinformatics, data integration.

1. Introduction

Systems approaches are critical to understanding clinical phenotypes. One important challenge for further research is building functional knowledge on how genetic variation affects the function and expression of proteins (or some other functional unit) in a biological network. Until recently, a systems understanding of the effects of variation has been an elusive challenge, and it remains a significant opportunity for research. There are several resources that have begun to address this issue. Databases such as dbSNP (1), KEGG (2), UniProt/Swiss-Prot (3) are beginning to include information useful for understanding the prevalence of variation in proteins, functional domains, pathways, and systems. Meta-resources are also being

actively developed for understanding the effects of variation on systems. SNPs3D (<http://www.snps3d.org/>), for example, displays candidate pathways to links with functional predictions of nonsynonymous SNPs (4). The Pharmacogenetics Knowledgebase (PharmGKB, <http://www.pharmgkb.org/>) has well-curated pathways with links to variation submitted to the resource, and phenotype data whenever available (5).

The task of linking genetic variation to functional protein effects in a biological system is complicated by the variety of effects that genetic variation can impose on a gene or gene product. Nonsynonymous SNPs can affect protein stability and function, while synonymous SNPs can effect transcript alternative splicing; noncoding SNPs can also affect gene transcription (6). Although there are analytical tools for the prediction of functional nonsynonymous variation, other types of SNPs have been difficult to quantify. Our current efforts focus on how nonsynonymous mutations affect both protein function and the interplay of these proteins in the pathway contexts. Recently, researchers have begun to perform computational studies that address these important research questions. In particular, Ye et al. showed that many disease-associated mutations from Swiss-Prot are likely involved in disrupting protein interactions (7). They also found that Swiss-Prot mutations were distributed differently than SNPs using comparative protein structural models (7). PharmGKB has also been curating pathways and genotype data associated with pharmacogenetics (5).

A continuing open research question today is how to relate genetic mutation data to protein interaction networks, and therefore to the molecular causes of disease. Addressing this question raises several challenges. First, different types of global data sets must be integrated so that interaction data and mutation data can be arbitrarily queried, visualized, and analyzed together. Second, the disease context of the mutations and the proteins within the system must be revealed and understood. Finally, it would be useful to have a model for how mutations cause disease within the specific context of protein interaction sub-networks. However, there have not been immediate solutions to these challenges, although significant advancements have been made (4, 7). Further complicating this is that databases of mutations tend to be highly biased toward well-characterized proteins of interest, and therefore skewing the distributions of mutations compared to natural occurrence. Similarly, nonsynonymous SNPs may not be neutral and may alter the function of proteins, or disease-associated mutations may only be in linkage or linkage disequilibrium with the causative allele (depending on the genetic approach).

We present a method that explores solutions to the challenging questions of connecting genetic mutation data effects in interaction networks. Our initial case study uses a database of

human protein interactions and mapped annotated SNPs among interacting proteins, which are involved in the Alzheimer's Disease (AD) process. To this end we built a database consisting of all related protein interactions, participating protein's "AD relevant" score, and all annotated mutations and SNPs. We built a model for disease-causing mutations from a specific set of interacting proteins that are associated with the AD phenotype. Using a model trained on the most highly ranked AD proteins, we are able to discriminate phenotypically associated mutations from polymorphisms with nearly 83.7% accuracy. Interestingly, the model was more accurate when trained on the subset of proteins most likely to be associated with AD, while the full set of proteins resulted in poorer performance. When we mapped the SNP data onto interacting proteins, we also observed that while deleterious SNPs are broadly distributed among interaction hub proteins and peripheral (non-hub) proteins in the AD sub-network, most highly ranked AD proteins (subnetwork hubs) contain high ratios of deleterious SNPs, suggesting a link between the system properties of protein functions and the disease phenotypes.

We take the following steps toward integration and analysis of mutations in protein interaction data:

1. Collect a data set of experimentally determined protein interactions, and import the interactions into a relational database.
2. Using phenotypically annotated mutation data and SNPs derived from several sources, identify nonsynonymous genetic variation for the proteins identified in Step 1.
3. Identify proteins relevant to the specific phenotype of interest by identifying proteins highly connected with known disease-associated proteins.
4. Build a matrix of sequence- and structure-based features for each mutation in the subset of proteins identified in Step 3.
5. Train a supervised learning algorithm to discriminate disease-associated mutations from polymorphisms using the matrix from Step 4.
6. Visualize the model on a protein interaction network.

2. Materials

2.1. Protein Interaction Data

To integrate the protein interaction data we used the following steps:

1. We obtain the human protein interaction data from the Online Predicted Human Interaction Database (OPHID) (8). OPHID is a comprehensive and integrated repository of publicly known or predicted human protein interactions,

which are derived from curated literature, high-throughput experiments, or computational inference from homologous protein interactions in model organisms. Predicted human interactions are usually confirmed with additional evidence such as domain co-occurrence, co-expression, and GO semantic distances (8). We choose OPHID over other human protein interaction databases due to its high data coverage compared to the curated HPRD database (24) and its early availability that preceded some recent major development such as the HAPPI database (25).

2. We used the protein identifier mapping table provided at the ftp site of the UniProt Knowledgebase (3) to perform mapping between human protein's Swiss-Prot IDs and their UniProt IDs for all OPHID proteins.
3. The entire collection of processed OPHID data were subsequently loaded into an Oracle 10g relational database system using PERL program parsers and Oracle DBMS sqlldr data loading software facility.
4. For each interaction pair, we further assigned a heuristic interaction confidence score, based on a simple scoring method described in (1). The approach can be summarized as the following. First, we manually determine a confidence score between 0 and 1 (the higher, the better), which provides an estimate of data quality reliability. We assign heuristic scores of 0.9, 0.5, and 0.3 to protein interactions obtained from human, medium-quality non-human mammalian systems, and high-quality non-mammalian systems, respectively. Then, we perform a score combination using the following formula, where $p_{INT}(a, b)$ is the overall confidence score for the interaction between proteins a and b , and $p_i(a, b)$ is the confidence score for a specific interaction derived from resource i between proteins a and b :
$$p_{INT}(a, b) = 1 - \prod_i (1 - p_i(a, b))$$

For the case study described here, OPHID was downloaded in February 2006. This release of OPHID contains more than 47,213 human protein interactions among 10,577 proteins identified by Swiss-Prot IDs. After the mapping to UniProt, the database contains 46,556 unique protein interactions among 9,959 proteins identified by UniProt IDs.

2.2. Data Integration

We used Oracle 10g as the data integration platform and developed a coherent relational database to integrate data from protein interactions, SNPs, mutations and functional predictions, and the AD-specific disease protein list into relational database tables. The data integration process requires the use of PERL scripts to parse data generated from different sources, and SQL scripts to join different tables together to format the data into final report views.

We obtained mutation and SNP data from several sources. For bulk SNP data, dbSNP is the central repository (<http://www.ncbi.nlm.nih.gov/projects/SNP/>), and is distributed in parseable XML format. Non-synonymous SNPs are annotated with protein accessions and positions. For disease-associated mutation data, several options are available. First, Swiss-Prot (<http://www.expasy.org/>) entries contain mutations in the VARIANT records. Second, OMIM (<http://www.ncbi.nlm.nih.gov/entrez/query.fcgi?db=OMIM>) contains mutations associated with disease with natural language phenotype annotations. Other options include the semi-private HGMD (<http://www.hgmd.org/>) and locus-specific databases.

Here we describe our process to extract mutation and SNP data from Swiss-Prot files:

1. Mutation and SNP lists were determined by downloading each ID in UniProtKB/Swiss-Prot format using a BioPerl (<http://www.bioperl.org/>) script that utilizes the Bio::DB::SwissProt module.
2. The resulting files were then parsed using a home-grown Python script, by extracting the feature tags from each protein. Note that although we used an informal script to parse the Swiss-Prot entries, many users will likely use Swissknife (<http://swissknife.sourceforge.net/>), an object-oriented Perl library to handle these files.
3. When parsing the feature tags, we only considered VARIANT features and did not consider mutagens and VAR_SEQ features. dbSNP entries are annotated with a dbSNP rsid and mutations are annotated with a phenotype or left unannotated. Some SNPs contain both a phenotype annotation and an rsid. Swiss-Prot was chosen because it is generally well annotated and contains both mutation and SNPs within the same context.

3. Methods

3.1. Identification of Proteins Associated with a Specific Disease or Phenotype

The identification of disease-specific proteins was done in the following steps:

1. To obtain a list of disease-specific proteins, we performed a search of the OMIM database, retrieving each OMIM gene record in which the “description” field contains the term “Alzheimer”.
2. These initial proteins were used as “seeds” to expand into an interaction data set to proteins with high confidence interactions, described in detail in (1).

3. Using the database built in **Section 2.2**, all Swiss-Prot SNPs and mutations were identified in the expanded set of proteins.

For the AD-related case study, 65 OMIM genes or 70 proteins (identified by Swiss-Prot IDs or UniProt IDs) were retrieved from the initial search on OMIM. These 70 proteins were used as “seeds” to expand into an interaction data set (AD interaction sub-network) consisting of 655 expanded proteins as described in detail in (1). It should be noted that the full set of interacting proteins contains all possible interactions, not necessarily an interaction important, or even observed, in AD tissues. For each of 70 AD proteins, mutations from Swiss-Prot were extracted as described in **Section 2**. Among the 655 total AD expanded interacting proteins identified, 240 proteins have been found to contain 2,941 mutations, which were annotated by the Swiss-Prot database with information from the dbSNP with various annotated phenotypes. In **Fig. 19.1**, we show a histogram including the number of SNPs and disease mutations by protein. Annotated mutations ranged from 0 to a maximum of 210 in the androgen receptor (ANDR_HUMAN). Other proteins highly populated with mutations include TP53_HUMAN (193 mutations), FA8_HUMAN (185), CO4A5_HUMAN (141), and FA9_HUMAN (138).

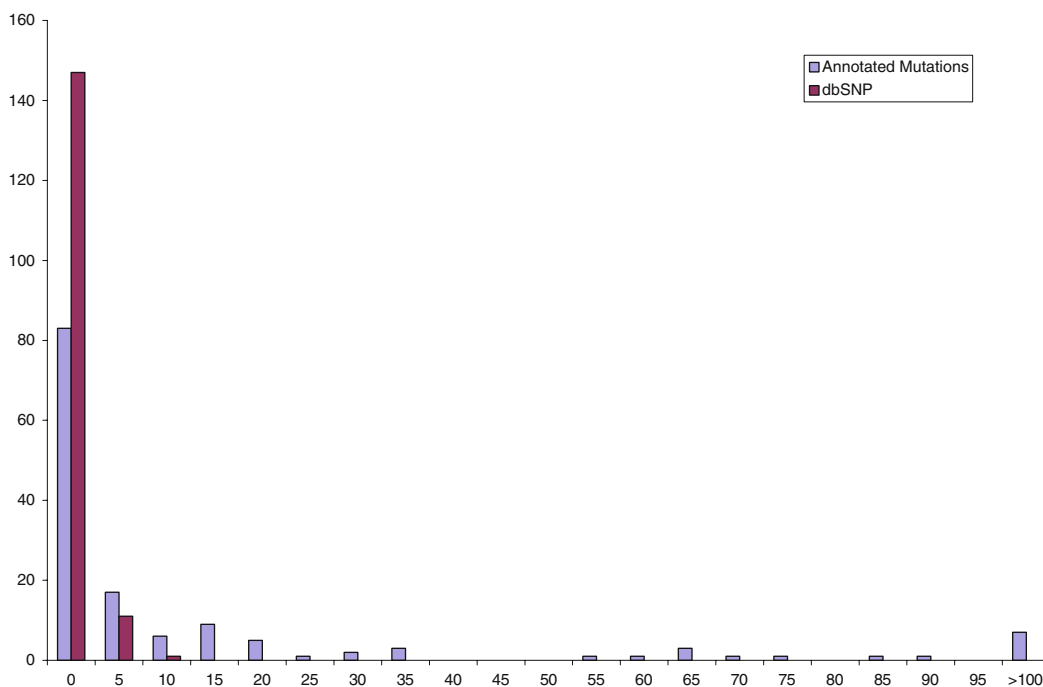


Fig. 19.1. Distribution of annotated mutations and SNPs by proteins shown to interact with an Alzheimer’s disease-related protein. Each of the 655 identified proteins in Swiss-Prot had their mutations and SNPs cataloged from the VARIANT features in the Swiss-Prot flat file. Not surprisingly, few proteins have more than 10 SNPs while many proteins have large numbers of mutations.

3.2. SNP Function Prediction

Initially, methods for SNP function prediction were based on conservation in multiple sequence alignments derived from similarity searches against nonredundant sequence databases and the mutations could be either ranked or scored based on parameterization. Parameterization was either performed using unbiased mutagenesis experiments in LacI, T4 Lysozyme and/or HIV protease, such as SIFT (9), or using structure/function rules, such as PolyPhen (10). SIFT is used as a feature here and SIFT-based predictions can be determined by running the method on their website (<http://blocks.fhcrc.org/sift/SIFT.html>). PolyPhen was not used here, but can be run from the website (<http://coot.embl.de/PolyPhen/>).

Later, more sophisticated approaches were utilized using decision trees (11) or support vector machines (12, 13). Generally, the primary predictive features for such methods are derived from protein sequence, protein sequence conservation, structure, structural rules; specifically, the most important features are generally the identities of the wild-type and mutant amino acids, sequence conservation, and the degree of residue burial in a protein structure (11). These prediction methods generally require the use of a statistical modeling package such as R or MATLAB.

3.3. Model to Predict Mutations Within the Context of Disease

To describe how we build a support vector machine (SVM) model for the mutations in a set of proteins, we will walk through the previously described AD case study. SVMs were chosen because they have been applied to mutation data previously with good performance (4, 14–16). The goal of such a model is to discriminate disease-causing mutations from polymorphisms. It is important to understand that the set of proteins in question can be derived from several sources. For example, they can be a database of proteins with annotated SNPs or mutations, a family of related proteins, or functionally connected proteins in a protein interaction network.

To build a model we employ the following steps:

1. Using the previously determined set of proteins associated with a specific phenotype, the SNPs and mutations associated with each protein are determined (*see Section 2.2*).
2. Each of the mutations and SNPs in the set based on protein interactions is annotated with features based on four different attributes. They include the *p*-value score from SIFT (9), sequence conservation score (based on information theory), and a window of sequence neighbors and blosum62 substitution score (17). Sequence conservation was evaluated using a position-specific scoring matrix (PSSM) from the output of PSI-BLAST (18). PSI-BLAST is performed using the BLASTPGP (<http://www.ncbi.nlm.nih.gov/BLAST/download.shtml>) package from NCBI and is queried against

the nonredundant protein database (NR) from GenBank (<ftp://ftp.ncbi.nih.gov/blast/db>). Three scores from the output were extracted; a PSSM, a weighted observed percentage, and information per position score for each residue in the sequence. For the sequence conservation of adjacent residues, a window size of 20 using the information per position score was considered. For N-terminal and C-terminal residues, we used an average conservation score of the sequence to fill in the N-terminal (C-terminal) residue scores. For the window of residue type, we considered window of size 21; 10 residues on the N-terminal side (left), 10 residues on the C-terminal side (right), and itself. Our description of the window of residue type has a similar protocol to those published for classifying protein phosphorylation sites (19). Other features are possible, including those based on sequence-based data mining tools and protein-structure-based approaches; *see Section 4* for more information.

3. Each mutation is labeled (-1: dbSNP, 1: Swiss-Prot) and positions annotated as being in both are removed, because their ambiguous annotation prevents them from being accurately classified as neutral or damaging. There are 45 features, which are (listed by feature index id followed by feature type): 1 SIFT, 2:4 PSSM, weighted, information per position, 5:24 information per position window, 25:44 sequence window, 45 BLOSUM62.
4. For SVM classification (20), we use a linear kernel and default regularization parameter (C). We employed an SVM for classification using the SVM^{light} (21) and its Matlab interface. Since each feature has a different scale, all examples were normalized to the $[0, 1]$ interval. The ratio between deleterious and neutral mutations is 13: 1. To overcome this imbalanced data training, we gave more weight to negative (neutral) samples. All evaluations were performed using leave-one-out cross-validation.

To continue the example of AD-related proteins, the 655 AD-related proteins in the protein interaction subnetwork were determined by looking at all interactions with the AD proteins which were extracted from OMIM. Second, we have the subset of the 70 most highly ranked proteins, based on the previously described analysis. SIFT was run on all 655 proteins, giving a total 2,893 annotations with an average SIFT p -value score of 0.118. When separated by dbSNP, the phenotypically annotated mutations had an average p -value of 0.095, while the SNPs in this set had a score of 0.290. Similarly, the subset of 70 important proteins (35 of which actually contained mutations) had both disease-annotated mutations and SNPs. The disease-annotated mutations had a SIFT p -value average of 0.110 and the SNPs had an average p -value of 0.327. Similar

trends have been observed before, because SNPs are less likely to affect protein function than the disease-associated mutations. We then encoded the 367 mutations annotated in the subset of proteins using the features described above.

As an example, we compare the performance differences between protein SVM predictions and protein SIFT predictions. Not surprisingly, the SVM method gives improved performance over the SIFT method alone, although SIFT remains a highly valuable feature (**Table 19.1**). Overall, the model has 83.7% accuracy and is more sensitive than SIFT. Interestingly, when the model is applied to an expanded set of interacting proteins less likely to be related to AD, the model's performance declines to a level similar to SIFT (accuracy of 70.94% versus 71.75%).

3.4. Feature Selection

We then used feature selection to find a subset of features for optimal classification. An important part of feature selection is to rank features according to their importance for class discrimination. Feature ranking is based on the weight associated with each feature in a classifier and can be determined by recursive feature elimination (RFE). RFE method computes the feature ranking as follows:

1. The SVM is trained and a weight associated with each feature is computed using all of the features initially.
2. The feature with the smallest magnitude of the weight is removed. This feature is the least important one. Leaving this feature out, we retrained the SVM and recomputed all the weights.
3. Steps 1–2 are iterated until all of the features are exhausted. In this way, features are recursively eliminated and ranked. The feature eliminated last is the most important feature. We implemented the RFE feature ranking method in Matlab. RFE with an SVM was described in Guyon's paper on micro-array data sets (22).

We wanted to evaluate which features were more important than others for class discrimination. The top ranked features based on the SVM-RFE method for the AD-related protein set include SIFT and information per position from PSI-BLAST for the central

Table 19.1
Performance comparison between SVM and SIFT score predictions on the set of 70 Alzheimer's disease-associated proteins

	Sensitivity (%)	Precision (%)	Specificity (%)	Accuracy (%)
SVM prediction	85.5	95.5	69.1	83.7
SIFT prediction	70.8	94.3	66.7	70.3

Table 19.2
Performance comparison using top k features by SVM prediction

Top k features	Sensitivity (%)	Precision (%)	Specificity (%)	Accuracy (%)
5	73.9	98.0	88.1	75.5
10	80.3	97.8	85.7	80.9
15	85.2	97.5	83.3	85.0
20	84.6	97.5	83.3	84.5
25	85.9	97.2	81.0	85.3
30	85.9	97.2	81.0	85.3
35	85.2	96.5	76.2	84.2
40	85.2	95.9	71.4	83.7
45	85.5	95.5	69.1	83.7

and adjacent residues. Based on this feature ranking, we computed performance using top ranked features (**Table 19.2**). The highest performance is achieved using the top 25 or 30 features.

3.5. Data Visualization

Due to the large-scale data integration nature of this study, we chose network data visualization software with the following two features, (1) ability to directly support relational database queries as input to the visualization environment; (2) ability to combine network features, scores, and data variables created in real time with maximal flexibility for the final visualization output. Therefore, we chose ProteoLens (<http://bio.informatics.iupui.edu/protolens>, also see reference 26) over the popular software tools such as Cytoscape (23). Using ProteoLens, we are writing SQL queries to generate network data and prepare annotated “data associations” either as “node associations” or “edge associations.” Examples of the “node associations” are protein target score identified by protein ID and the protein’s gene deleterious SNP count identified by proteins. Examples of the “edge associations” are protein interaction pairs identified by both protein A ID and protein B ID and protein interaction confidence score identified the same way. The node associations are mapped to the network node’s display property such as node size, shape, and color, whereas the edge associations are mapped to the network edge’s display property such as edge width, color, and type, all automatically by the ProteoLens software. The final visualization can be generated as either a PDF file or a PNG image file, specified by the user.

In **Fig. 19.2**, we show a visualization of the AD protein interaction subnetwork, in which protein interaction, SNP annotation, and disease protein curation information is integrated.

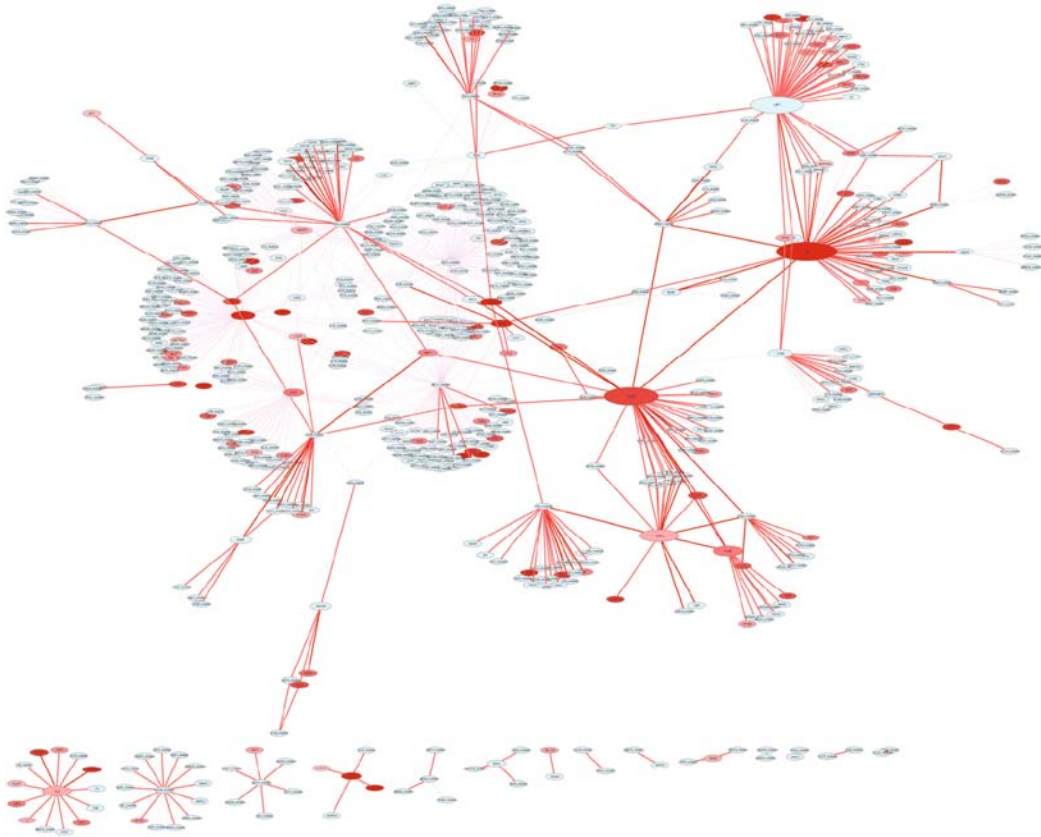


Fig. 19.2. **Visualization of the Alzheimer's disease protein interaction subnetwork.** Proteins are represented as nodes and protein interactions are represented as edges between nodes with a line width of 1 (faintly visible in some parts of the network). The size of the nodes is drawn in proportion to the protein's subnetwork significance score (approximately subnetwork protein centrality measure) described in (1). The color of the nodes, when not shown as the default color (missing SNP information from data integration effort), is on a sliding *gray* scale of which the color intensity is proportional to the ratio of deleterious mutations over all mutations recorded for the proteins annotated. The widths and color intensity of the edges is proportional to the level of confidence (ranging from 0 to 1) for the given protein interaction.

Note that most protein interactions are connected above statistical chance to form a large connected subnetwork, which underlies the context of the AD biological process. When we focus our attention only onto the subnetwork highly relevant proteins with matched SNP annotations (large nodes with non-default colors as shown in the figure), we do see a correlation between the protein's relevance to the AD process and the percentage of deleterious SNPs for those proteins. We can also observe other phenotypical effects among non-AD relevant proteins (seen as peripheral yet red-colored nodes in the figure), primarily because these proteins, although

insignificant in the AD process, play important roles in other non-AD disease processes (e.g., p53, etc, and btk, which are all peripheral in the AD subnetwork but are implicated in other diseases).

4. Notes



1. *Use of other features for the classification of SNPs and mutations.* It is important to note that the features described here are only a subset of the possible features that can be used to classify mutations. Features based on protein structure information or comparative modeling information have been used previously with improved results (11, 12). Features based on phylogenetic information have also been shown to be useful (12).
2. *Choice of a neutral set of mutations to compare with disease.* In this case we chose polymorphisms from Swiss-Prot as a model of a neutral set of mutations to compare against disease-associated mutations. A subset of non-synonymous SNPs are known to be functional, that is, they likely alter the protein product's function. A more rigorous approach might be to use SNPs experimentally determined to be neutral or to use saturation mutagenesis experiments for comparison (11).

Acknowledgments

Support for this work was provided by NIH grants K22LM009135 (PI: Mooney) and R01LM009722 (PI: Mooney).

References

1. Wheeler, D. L., Barrett, T., Benson, D. A., Bryant, S. H., Canese, K., Chetvernin, V., Church, D. M., DiCuccio, M., Edgar, R., Federhen, S., Geer, L. Y., Helmberg, W., Kapustin, Y., Kenton, D. L., Khovayko, O., Lipman, D. J., Madden, T. L., Maglott, D. R., Ostell, J., Pruitt, K. D., Schuler, G. D., Schriml, L. M., Sequeira, E., Sherry, S. T., Sirotkin, K., Souvorov, A., Starchenko, G., Suzek, T. O., Tatusov, R., Tatusova, T. A., Wagner, L., and Yaschenko, E. Database resources of the National Center for Biotechnology Information. *Nucleic Acids Res* 2006, 34:D173–80.
2. Kanehisa, M., Goto, S., Hattori, M., Aoki-Kinoshita, K. F., Itoh, M., Kawashima, S., Katayama, T., Araki, M., and Hirakawa, M. From genomics to chemical genomics: new developments in KEGG. *Nucleic Acids Res* 2006, 34:D354–7.
3. Wu, C. H., Apweiler, R., Bairoch, A., Natale, D. A., Barker, W. C., Boeckmann, B., Ferro, S., Gasteiger, E., Huang, H., Lopez, R., Magrane, M., Martin, M. J., Mazumder, R., O'Donovan, C., Redaschi, N., and Suzek, B. The Universal Protein Resource (UniProt): an expanding universe of protein information. *Nucleic Acids Res* 2006, 34:D187–91.

4. Yue, P., Melamud, E., and Moulton, J. SNPs3D: candidate gene and SNP selection for association studies. *BMC Bioinformatics* 2006, 7:166.
5. Klein, T. E., and Altman, R. B. PharmGKB: the pharmacogenetics and pharmacogenomics knowledge base. *Pharmacogenomics J* 2004, 4(1):1.
6. Mooney, S. Bioinformatics approaches and resources for single nucleotide polymorphism functional analysis. *Brief Bioinform* 2005, 6:44–56.
7. Ye, Y., Li, Z., and Godzik, A. Modeling and analyzing three-dimensional structures of human disease proteins. *Pac Symp on Bio-comput* 2006, 11:439–50.
8. Brown, K. R., and Jurisica, I. Online predicted human interaction database. *Bioinformatics* 2005, 21:2076–82.
9. Ng, P. C., and Henikoff, S. SIFT: Predicting amino acid changes that affect protein function. *Nucleic Acids Res* 2003, 31:3812–4.
10. Ramensky, V., Bork, P., and Sunyaev, S. Human non-synonymous SNPs: server and survey. *Nucleic Acids Res* 2002, 30:3894–900.
11. Saunders, C. T., and Baker, D. Evaluation of structural and evolutionary contributions to deleterious mutation prediction. *J Mol Biol* 2002, 322:891–901.
12. Karchin, R., Kelly, L., and Sali, A. Improving functional annotation of non-synonymous SNPs with information theory. *Pac Symp Bio-comput* 2005:397–408.
13. Krishnan, V. G., and Westhead, D. R. A comparative study of machine-learning methods to predict the effects of single nucleotide polymorphisms on protein function. *Bioinformatics* 2003, 19:2199–209.
14. Capriotti, E., Calabrese, R., and Casadio, R. Predicting the insurgence of human genetic diseases associated to single point protein mutations with support vector machines and evolutionary information. *Bioinformatics* 2006, 22(22):2729–34.
15. Karchin, R., Diekhans, M., Kelly, L., Thomas, D. J., Pieper, U., Eswar, N., Haussler, D., and Sali, A. LS-SNP: large-scale annotation of coding non-synonymous SNPs based on multiple information sources. *Bioinformatics* 2005, 21:2814–20.
16. Karchin, R., Monteiro, A. N., Tavtigian, S. V., Carvalho, M. A., and Sali, A. Functional impact of missense variants in BRCA1 predicted by supervised learning. *PLoS Comput Biol* 2007, 3:e26.
17. Henikoff, S., and Henikoff, J. G. Amino acid substitution matrices from protein blocks. *Proc Natl Acad Sci U S A* 1992, 89:10915–9.
18. Altschul, S. F., Madden, T. L., Schaffer, A. A., Zhang, J., Zhang, Z., Miller, W., and Lipman, D. J. Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Res* 1997, 25:3389–402.
19. Iakoucheva, L. M., Radivojac, P., Brown, C. J., O'Connor, T. R., Sikes, J. G., Obradovic, Z., and Dunker, A. K. The importance of intrinsic disorder for protein phosphorylation. *Nucleic Acids Res* 2004, 32:1037–49.
20. Vapnik, V. N. *The Nature of Statistical Learning Theory*, 2005, Springer Verlag, New York.
21. Joachims, T. *Learning to classify text using support vector machines: methods, theory, and algorithms*. 2002, Kluwer Academic Publishers, Dordrecht.
22. Guyon, I., Weston, J., Barnhill, S., and Vapnik, V. Supervised feature selection via dependence estimation. *Mach Learn* 2002, 46:389–422.
23. Shannon, P., Markiel, A., Ozier, O., Baliga, N. S., Wang, J. T., Ramage, D., Amin, N., Schwikowski, B., and Ideker, T. Cytoscape: a software environment for integrated models of biomolecular interaction networks. *Genome Res* 2003, 13:2498–504.
24. Mishra, G. R., Suresh, M., et al. Human protein reference database – 2006 update. *Nucleic Acids Res*, 2006, 34(Database issue):D411–4.
25. Chen, J. Y., Mamidipalli, S. R., and Huan, T. HAPPI: an Online Database of Comprehensive Human Annotated and Predicted Protein Interactions, *BMC Genomics* 2009, (In press).
26. Huan, T., Sivachenko, A. Y., Harrison, S. H., and Chen, J. Y. ProteoLens: a visual analytic tool for multi-scale database-driven biological network data mining. *BMC bioinformatics*, 2008, 9 Suppl: S5.

Chapter 20

Effects of Functional Bias on Supervised Learning of a Gene Network Model

Insuk Lee and Edward M. Marcotte

Abstract

Gene networks have proven to be an effective approach for modeling cellular systems, capable of capturing some of the extreme complexity of cells in a formal theoretical framework. Not surprisingly, this complexity, combined with our still-limited amount of experimental data measuring the genes and their interactions, makes the reconstruction of gene networks difficult. One powerful strategy has been to analyze functional genomics data using supervised learning of network relationships based upon reference examples from our current knowledge. However, this reliance on the set of reference examples for the supervised learning can introduce major pitfalls, with misleading reference sets resulting in suboptimal learning. There are three requirements for an effective reference set: comprehensiveness, reliability, and freedom from bias. Perhaps not too surprisingly, our current knowledge about gene function is highly biased toward several specific biological functions, such as protein synthesis. This functional bias in the reference set, especially combined with the corresponding functional bias in data sets, induces biased learning that can, in turn, lead to false positive biological discoveries, as we show here for the yeast *Saccharomyces cerevisiae*. This suggests that careful use of current knowledge and genomics data is required for successful gene network modeling using the supervised learning approach. We provide guidance for better use of these data in learning gene networks.

Key words: Gene network model, supervised learning, classification, functional coupling, functional bias, reference set, genomics data.

1. Introduction

A major goal for our system-level understanding of a cell or an organism is the identification of the functions of all genes/proteins and their organization into pathways. With the classical one-gene-one-study approach, this goal is certainly daunting, if not impossible. However, the massive generation of biological data by

high-throughput techniques developed over the past decade brings this ambitious goal much closer, and abundant functional genomics data provide opportunities for modeling global gene/protein networks, which may shed light on our understanding of cellular systems.

Supervised machine-learning approaches have recently become popular in global gene/protein network modeling for various organisms (1–5). Supervised learning is generally considered a “classification” task, in which we start with classes predefined by some criterion (usually given by expert opinion) and attempt to find additional cases of these from the data. For modeling gene networks, the typical approach is not the prediction of genes with a completely defined set of cellular functions – this strategy is difficult, not least, because the total set of gene functions is unknown and because many gene functions overlap. Instead, networks are often derived by examining two classes of gene pairs, functionally coupled or not. Note that the network models are intrinsically consistent with genes’ pleiotropic (multi-functional) natures. Connections (perhaps weighted) within such networks capture functional relationships among genes and can therefore be used to discover functions of uncharacterized genes, to define functional modules of genes, and to describe the organization of genes that contribute to the physiological state of the cell.

Learning by classification requires reference examples on which to train, and in this case they would be known, functionally coupled gene pairs. A set of reference examples is generally based on current knowledge and expert opinion. Reliable examples of gene functional coupling can be derived easily from various biological annotation sets based mostly on manual curation by expert biologists (**Table 20.1**). In order to allow effective supervised learning, a reference set must

Table 20.1
Annotation databases for gene functions

<p>GO (gene ontology) biological process http://www.geneontology.org/ontology/process.ontology GO is hierarchically organized, with the top-level (level 0) annotation being most general and the bottom level the most specific. Generally, the middle range of annotation provides a good compromise between specificity and comprehensiveness.</p>
<p>Kyoto Encyclopedia of Genes and Genomes (KEGG) pathway ftp://ftp.genome.jp/pub/kegg/pathways/sce/sce_gene_map.tab KEGG offers a three-level hierarchical annotation of biological pathways. The bottom-level terms are most useful as functional reference terms, but show a bias toward metabolic pathways.</p>
<p>CYGD (the comprehensive yeast genome database) functional category, hosted by MIPS ftp://ftpmips.gsf.de/yeast/catalogues/funcat/ CYGD is a reasonably comprehensive and detailed annotation set that is specific for yeast. The top level contains 11 broad functional categories that are useful for visualization and analysis of general functional trends.</p>

clearly be both comprehensive and reliable. However, reference sets for functional networks have another important requirement, which is freedom from functional bias. In fact, current biological annotations often display severe bias toward a few specific functions. In this chapter, we demonstrate how this reference set bias affects the investigation of functional linkages from diverse genomics data, and we present examples for the genes of the yeast *S. cerevisiae*.

2. Methods

2.1. Functional Bias in Current Functional Annotations

A number of different databases organize genes according to their pathways, such as the three listed in **Table 20.1**. Among these, gene ontology (GO) annotation has become popular for functional genomics studies due to its hierarchical organization and its separation of three aspects of gene function—biological process, which captures pathway relationships; cellular component, which describes sub-cellular localization of gene products; and molecular function, which focuses more on enzymatic and binding functionalities (6). GO has also consistently improved through community efforts (7). For example, by March 2005, 4,199 yeast genes (~72% of the total of 5,794 verified protein encoding genes) were annotated by at least one GO biological process annotation. Therefore, a functional annotation reference set based on GO biological processes is highly comprehensive, satisfying the first requirement for effective supervised learning.

Another requirement for an effective reference set is reliability. We can control the reliability of GO-derived reference sets both at the level of their generality and with regard to the evidence supporting them. First, we can control the generality of employed—general annotations such as metabolism (GO:0008152) are typically located near the top of the GO hierarchy and often provide poor resolution in the learning of specific cellular functions. By contrast, annotations near the bottom of the GO hierarchy are highly specific but annotate only one or a few genes, and thus they lack comprehensiveness. The middle layers of the gene ontology hierarchy generally provide a more optimal trade-off between comprehensiveness and reliability (*see Note 1*). GO also provides evidence codes (**Table 20.2**) as another way of controlling the reliability of the reference set. Annotations by traceable author statement (TAS), inferred from direct assay (IDA), inferred from mutant phenotype (IMP), inferred from genetic interaction (IGI), and inferred from physical interaction (IPI) are generally considered as highly reliable annotations. As of March, 2005, the GO biological process had 4,199 annotated yeast genes with a total 11,430 terms, of which 9,093 (~80%) are based on one of these five types of highly reliable evidence.

Table 20.2
Gene ontology evidence codes and their reliability

Code	Description	Reliability
TAS	Traceable Author Statement	High
IDA	Inferred from Direct Assay	High
IMP	Inferred from Mutant Phenotype	High
IGI	Inferred from Genetic Interaction	High
IPI	Inferred from Physical Interaction	High
ISS	Inferred from Sequence or Structural Similarity	Low
IEP	Inferred from Expression Pattern	Low
NAS	Non-traceable Author Statement	Low
IEA	Inferred from Electronic Annotation	Low

The reference set obtained by considering the above GO annotations, though highly reliable and comprehensive, may still have a systematic bias toward a few specific functions. This in turn may lead to biased learning of the given genomics data. We examined the distribution of GO biological process terms from the middle layers of the annotation hierarchy (between levels 6 and 10, *see Note 1*). Although we expected similar genome coverage among functional terms, we found a few dominant functional terms used to annotate yeast genes. From 1,067 selected GO biological process terms, we observed that a single functional term, protein biosynthesis (GO:0006412), accounts for more than 4% of total gene annotations, although its expected coverage is less than 0.1% ($100 / 1,067 < 0.1$) (**Fig. 20.1A**, filled bars).

In order to evaluate functional coupling between genes, we derived reference gene pairs that are functionally coupled (positives) and pairs that are not functionally coupled (negatives) from the given gene functional annotation set. The simplest way of deriving a set of positive examples is to pair genes that share at least one common functional description. A corresponding set of negative examples can be derived by pairing genes that do not share any functional description (*see Note 2*). As a result of this gene pairing, the functional bias of gene annotation is dramatically amplified in the reference sets of functionally coupled gene pairs (**Fig. 20.1A**, empty bars). This functional bias annotation is not specific to a particular annotation set. We observe a similar functional bias toward the ribosome, the core machinery of protein biosynthesis, in another commonly used gene function annotation set, the Kyoto Encyclopedia of Genes and Genomes (KEGG) (8), which focuses mainly on metabolic pathway information (**Fig. 20.1B**).

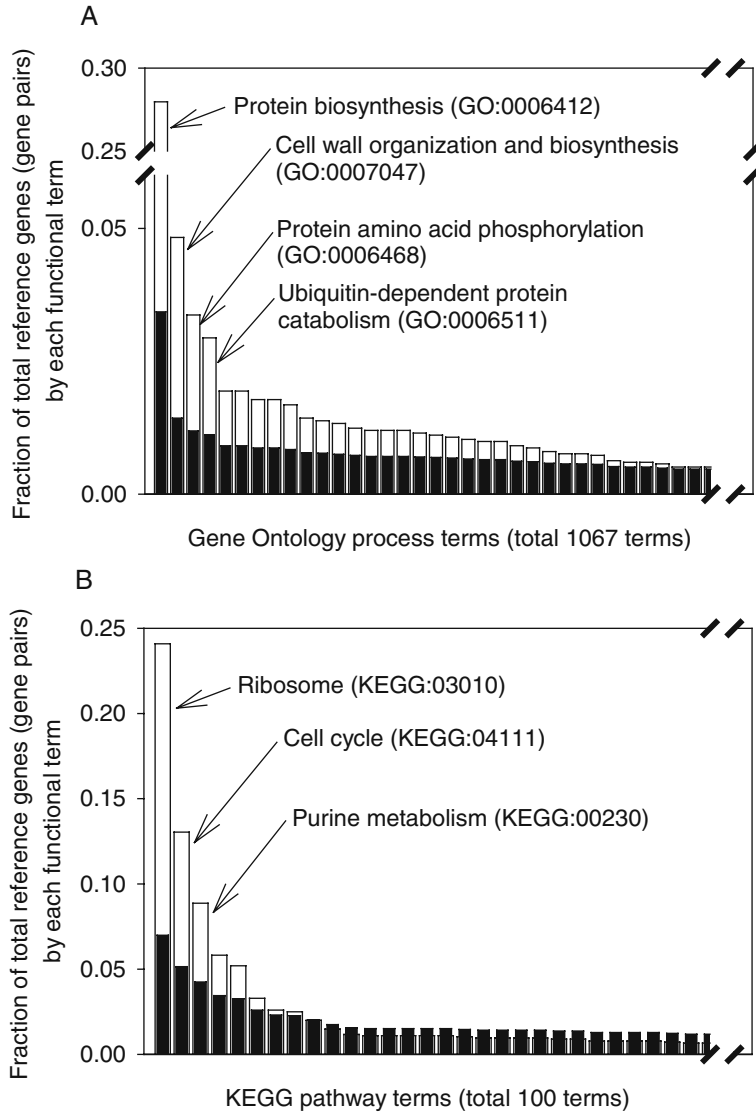


Fig. 20.1. The distributions of functional terms among annotated genes (*filled bars*) and gene pairs (*empty bars*) by (A) the gene ontology (GO) biological process annotations, and (B) the Kyoto Encyclopedia of Genes and Genomes (KEGG). The pathways illustrate functional bias in both major annotation databases. Only a few of the most dominant terms out of the 1,067 GO biological process terms between levels 6 and 10 and out of 100 KEGG pathway terms at the bottom level are labeled. In both GO biological process and KEGG, the most dominant functional term is related to protein biosynthesis (the ribosome being the major component represented), and this single term accounts for 3 and 7% of total gene annotations by GO and KEGG, respectively. This functional bias in gene annotations has become dramatically amplified by pairing genes for the same functional terms to provide references of gene pairs functionally coupled. As references of gene pairs, about 25% of total reference examples are based upon only a single most dominant functional term in both annotation sets.

There are two major systematic biases that introduce functional biases into our current knowledge bases. First, there are differences in the size of cellular functional modules. We will refer to this as size bias. For example, the protein biosynthesis functional module consists of a huge translational machine, the ribosome (composed of 150 ~ 200 proteins in yeast), as well as many other co-factors. Thus, this single specific functional module annotates many more genes than any other. The second bias we will refer to as a study bias. Biologists have historically studied genes using a one-gene-one-study approach, which naturally introduced a bias toward genes that are more important (and often biologically more essential) or those more readily studied (for example, genes with an obvious mutation phenotype are easier to study). Genes involved in protein biosynthesis have been subject to this study bias. The recent development of various high-throughput functional genomics analyses with reverse-genetics approaches solves the problem of study bias, but size bias is an intrinsic characteristic of cellular systems.

2.2. Effect of Reference Set Functional Bias on Supervised Learning

The inevitable functional bias in gene annotations potentially affects further discovery of gene functions and network organization. To demonstrate the effect of a single dominant functional term (protein biosynthesis, *see* Fig. 20.1A) of the reference set based on the GO biological process in learning functional gene coupling, we compare two different reference sets derived from the GO biological process annotation set: (1) a *biased reference set* that comprises all gene pairs sharing annotation, including the pairs sharing the function “protein biosynthesis”; and (2) an *unbiased reference set* based on the same pairs but excluding those sharing the protein biosynthesis term. The effects on learning for links are illustrated in Fig. 20.2 with examples from various types of yeast genome-wide functional genomics data.

We can infer which genes are functionally coupled by the co-expression patterns across different experimental conditions. The tendency toward co-expression can be measured by the Pearson correlation coefficient between any two genes’ expression profile vectors. For a set of gene pairs with a given range of co-expression tendencies, we calculate the log-likelihood score (*LLS*) using Bayesian statistics, as a measurement of the likelihood of functional coupling supported by the given data (*see* Note 3). In Fig. 20.2, log-likelihood scores are calculated with the 0.632 bootstrapping method (9) to minimize the over-fitting of models (*see* Note 4). For the functionally informative microarray data set, we observe a significant positive correlation between the tendency toward co-expression and the measured likelihood of functional coupling between pairs of genes.

For microarray data from yeast cell cycle time courses, log-likelihood scores are higher with the biased reference set than with the unbiased one (Fig. 20.2A). For the most significant data range

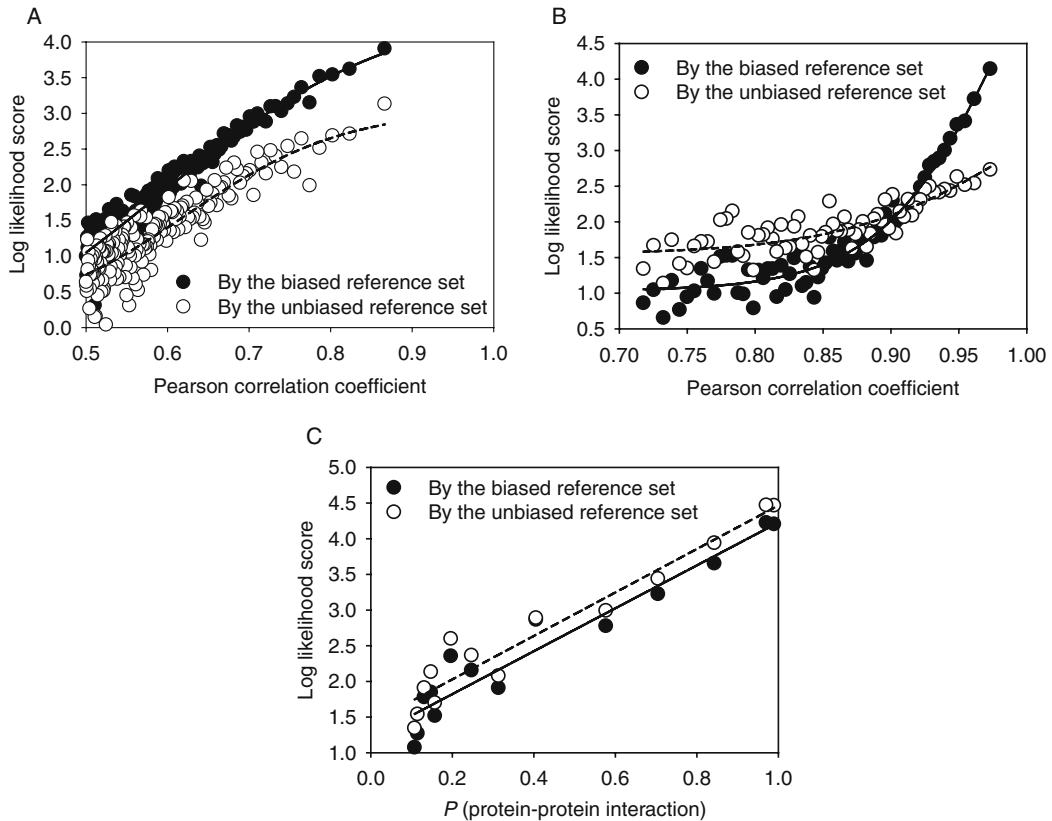


Fig. 20.2. Correlation between data-intrinsic scores that imply gene functional coupling (Pearson correlation coefficient measuring co-expression tendency or probability score of protein–protein interaction) and log-likelihood score (see **Note 3**) that measures the likelihood of gene functional coupling with the given supporting data. Three different data sets—**(A)** microarray data with cell cycle time courses, **(B)** microarray data with various heat-shock conditions, and **(C)** protein–protein interaction from affinity complex purification—are evaluated using two different reference sets derived from GO biological process annotation: (1) the biased set (*filled circle*), including gene pairs among the most dominant term “protein biosynthesis,” and (2) the unbiased set (*empty circle*), excluding reference gene pairs for the term “protein biosynthesis.” Each data point represents a bin of 1,000 gene pairs of the data set, which are sorted by data-intrinsic scores.

(the top 1,000 gene pairs), the log-likelihood score is about 4 (i.e., the likelihood is ~ 55 -fold higher than random) with the biased reference set, while it is about 3 (i.e., the likelihood is ~ 20 -fold higher than random) with the unbiased reference set. Thus, there is an increase of approximately threefold overall, deriving entirely from a single additional functional term. This observation becomes extreme as we examine the microarray data collected from heat-shock-treated cells (**Fig. 20.2B**). The positive correlation between the co-expression of genes across heat-shock conditions, and the likelihood of functional coupling, is very strong with the biased reference set, especially when the Pearson correlation coefficient is higher than 0.8. In the range of Pearson correlation coefficients from 0.8 to 1, the *LLS* increases from ~ 1 (i.e., the

likelihood is approximately threefold higher than random) to ~ 4.2 (likelihood ~ 66 -fold higher than random), achieving a ~ 22 -fold increase in the likelihood of functional coupling for the most co-expressed genes, apparently implying that this heat-shock microarray data carries strong information about gene functional couplings. However, masking the single dominant functional term, protein biosynthesis, is sufficient to remove most of the trend, showing only approximately threefold likelihood increase across the same range of Pearson correlation coefficients (from $LLS \sim 1.7$, 5.5-fold higher likelihood than random, to $LLS \sim 2.7$, 15-fold higher likelihood than random).

This over-optimism exhibited by the biased reference set largely disappears when we consider protein–protein interaction data. We compared the biased and the unbiased reference sets in evaluating a high-throughput protein–protein interaction data set derived from affinity purification of protein complexes followed by mass spectrometry analysis (10). Using machine-learning algorithms, the raw data set has been simplified to a set of 14,317 protein–protein physical interactions with associated probabilistic scores (10). In this data set, the biased reference actually provides very similar likelihood values to the unbiased reference (Fig. 20.2C). A similar trend is evident in a high-quality data set of 12,300 interactions derived from published protein physical and genetic interaction data (and excluding large-scale assay-derived interaction data) (11). This data set shows a very high overall quality and relatively little difference in performance between the unbiased reference set ($LLS = 3.85$) and the biased reference set ($LLS = 3.55$).

2.3. Effect of Genomics Data Set Functional Bias on Supervised Learning

What are the underlying characteristics of data sets sensitive to this reference set bias? Not surprisingly, in many data sets these appear to be functional biases that affect their performance in supervised learning. This trend is evident when measuring the functional bias as a function of interaction confidence score (the gene retrieval rate). The gene retrieval rates measured for genes of 11 different functional groups defined by the Munich Information Center for Protein Sequences (MIPS) (12) demonstrates a high bias toward genes involved in protein biosynthesis, which are among the most highly co-expressed gene pairs in yeast cell-cycle microarray experiments (Fig. 20.3A). This trend explains the overly optimistic evaluation of cell-cycle micro-array data sets by the biased reference set. It has been shown that proteins in stable complexes tend strongly to co-express (13). Therefore, co-expression of genes is an excellent feature for inferring interactions among proteins of stable complexes such as the ribosome, and, not surprisingly, the most strongly co-expressing genes are highly enriched for ribosomal protein pairs. This trend is exacerbated through the use of the biased reference set, which is over-represented for

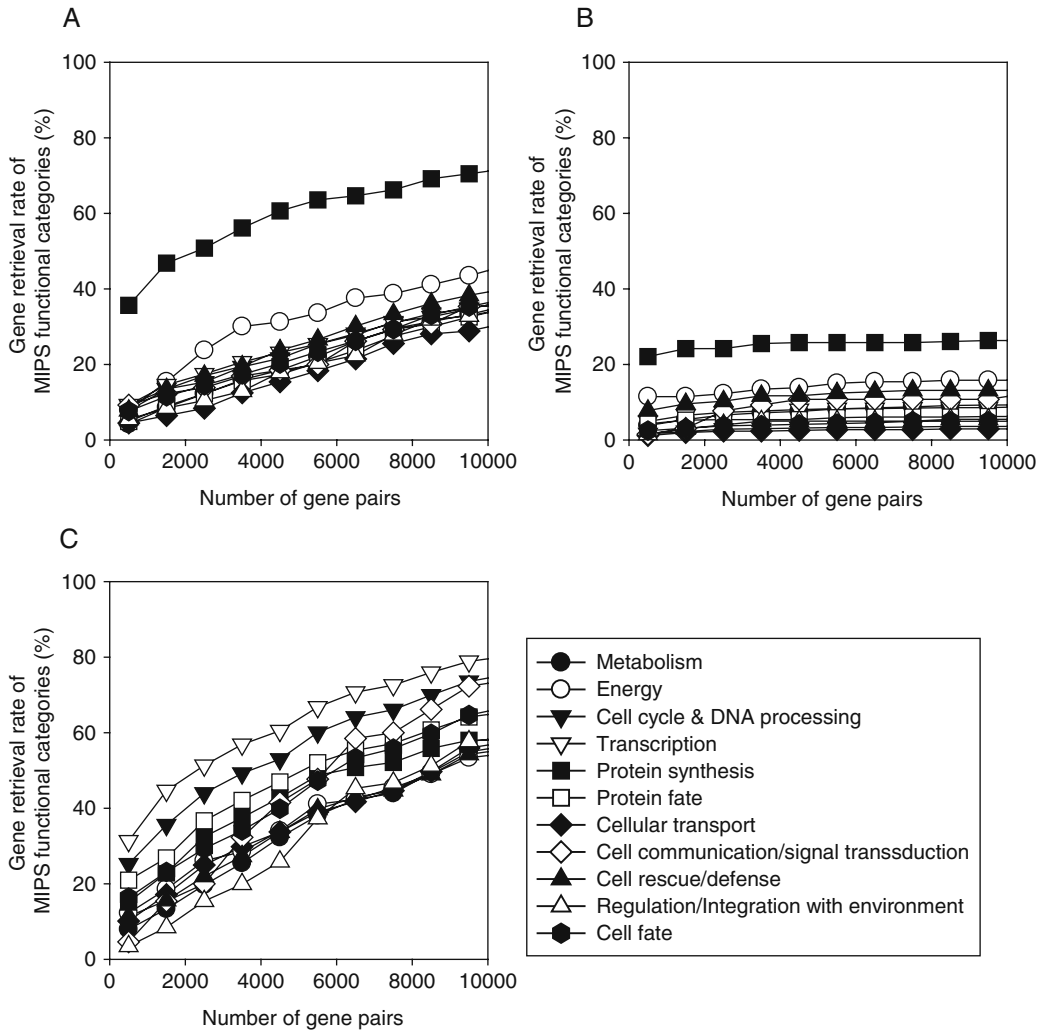


Fig. 20.3. Gene retrieval rate across 11 MIPS functional categories for the top 10,000 scored functional gene pairs in (A) co-expression during cell cycle time courses, (B) co-expression across various heat-shock conditions, and (C) protein-protein interactions from affinity complex purifications. The cumulative gene functional coverage for the given data set was measured based on MIPS' 11 top-level protein functional categories for every 1,000 gene pairs sorted by data-intrinsic scores.

ribosomal gene pairs. Thus, the generality of interactions discovered from these data may be suspect when the biased reference set is used.

The distribution of gene functions in the yeast heat-shock microarray data set is also interesting. While showing a similar enrichment of protein biosynthesis genes, this set also shows a flat gene retrieval rate (Fig. 20.3B)—i.e., the gene retrieval rate does not significantly increase with an increasing number of co-expressed gene pairs. This implies that co-expression during yeast heat shock is restricted to only a small percentage of cellular

systems. In this case, additional functional couplings are added to this small set of systems (which include the ribosome and other protein biosynthesis-related systems) without incorporating additional genes, leading to an increasingly dense network of functional linkages among this subset of genes. This is evident in the fact that functional couplings derived from co-expression on the heat-shock data cover only 9% of the yeast proteome and have a clustering coefficient (14) of 0.6 for the top 10,000 gene pairs, implying highly clustered interactions in a relatively few functional modules. By contrast, the same number of gene pairs derived from the cell-cycle data set covers about 36% of the proteome and has a lower clustering coefficient (0.28). The protein–protein interaction data sets show less functional bias (Fig. 20.3C), with large proteome coverage (58%) and a low clustering coefficient (0.14) for the number of gene pairs, implying that the information in these sets is distributed through many functional modules in the yeast cell.

2.4. Circumventing Functional Bias in Reference and Data Sets

How can we achieve reliable evaluation in the presence of persistent functional bias in the reference and data sets? Approaches for monitoring over-training, such as cross-validation and bootstrapping, do not solve this problem, as seen in the results of Fig. 20.2, which were carried out with 0.632 bootstrapping (see Note 4). One simple approach is to ignore the dominant terms for the purposes of training and testing. For an unbiased data set, this masking of a dominant functional term has minimal effects, as we show with the example of protein–protein interaction data (Fig. 20.2C and 3C). However, for biased data sets (Fig. 20.2A, B and 20.3A, B), we observe much lower likelihoods of functional coupling, implying that the optimistic likelihood scores were unrealistic and therefore risky to generalize to the rest of the data. Combined with cross-validation or bootstrapping, this dominant term masking is a simple but effective way to remove much of the negative effects of functional bias toward a few dominant functional annotations.

3. Notes



1. *Hierarchy in gene ontology.* The gene ontology is hierarchically organized, and references derived from different levels of the annotation hierarchy may result in quite different evaluations for identical data sets. This hierarchy is diamond-shaped, characterized by fewer descriptive terms at the top and bottom levels and by a gradual increase in terms as one moves toward

the middle layer of the annotation hierarchy. Generally speaking, top-level annotations provide extensive coverage but low information specificity (resolution), while low-level annotations describe fewer genes but with high specificity. Therefore, the trade-off between annotation coverage and specificity must be considered carefully in order to obtain an effective reference set for evaluating genomics data. Empirically, we find good performance using GO biological process terms between level 6 and 10 out of the total of 15 levels. The term “biological process” is considered to be level 0.

2. *Imbalance between positive and negative examples in a reference set of gene functional couplings.* Generating positive (negative) reference examples of functionally coupled gene pairs by pairing genes sharing (not sharing) any functional annotation results in a serious imbalance in the sizes of the two reference sets. We obtain a much larger negative reference set than positive (e.g., ~ 100 -fold larger negative reference set than positive based on the yeast GO biological process annotation of March 2005). This much higher frequency of negative examples in a reference set is problematic if one uses conventional data evaluation methods, which use a “true positive rate” (true positive / predicted as positive) such as a recall-precision curve (generally an overly pessimistic evaluation indicated by low precision for a given recall) or receiver operating characteristic (ROC) curve (generally an overly optimistic evaluation indicated by a high true positive rate for a given false positive rate) (15). With the severe size imbalance between the positive and negative reference sets, the measurement of the true positive rate is often discouraging in absolute terms; however, as a relative measure among different data sets, it works well. Gene functional couplings can be learned using these relative reliability scores, and various thresholds of scores will generate gene network models with varying accuracies and differing coverage.
3. *Evaluation of gene functional coupling by log likelihood scores.* We can evaluate the reliability of gene functional couplings supported by the given data using Bayesian statistics. A formal representation of Bayesian inference of the functional coupling between genes is the log likelihood score (*LLS*),

$$LLS = \ln \left(\frac{P(I|D)/P(\sim I|D)}{P(I)/P(\sim I)} \right),$$

where $P(I|D)$ and $P(\sim I|D)$ are the frequencies of gene functional coupling and its negation observed in the given genomics dataset (D), as measured by reference gene pairs. $P(I)$ and $P(\sim I)$ represent the prior expectations (the total frequencies of all positive and negative reference gene pairs,

respectively). A score of zero indicates coupled gene pairs in the data being tested are no more likely to be functionally coupled than random; higher scores indicate a more informative data set for identifying functional relationships.

4. *Evaluation with 0.632 bootstrapping.* To avoid over-fitting, we employed 0.632 bootstrapping (9) for all *LLS* evaluations. The 0.632 bootstrapping has been shown to provide a robust estimate of functional coupling accuracy. It is especially favored over cross-validation for very small datasets (9). Data evaluation with bootstrapping is therefore appropriate even for more poorly annotated genomes. Unlike cross-validation, which uses multiple tests and training sets by sampling data without replacement, 0.632 bootstrapping constructs the training set from data sampled with replacement and the test set from the non-sampled data. For the sampling, each instance has a probability of $1-1/n$ of not being sampled, resulting in $\sim 63.2\%$ of the data being in the training set and $\sim 36.8\%$ in the test set (16). The overall *LLS* is the weighted average of results for the two sets with 10 repetitions, equal to $0.632 * LLS_{\text{test}} + (1-0.632) * LLS_{\text{train}}$.

Acknowledgments

This work was supported by grants from the N.S.F. (IIS-0325116, EIA-0219061, 0241180), N.I.H. (GM06779-01), Welch (F-1515), and a Packard Fellowship (E.M.M.). We thank Cynthia V. Marcotte and Ray Hardesty for help with editing.

References

1. Jansen, R., Yu, H., et al. A Bayesian networks approach for predicting protein-protein interactions from genomic data. *Science* 2003; 302:449–53.
2. Lee, I., Date, S. V., et al. A probabilistic functional network of yeast genes. *Science* 2004; 306:1555–8.
3. Myers, C. L., Robson, D., et al. Discovery of biological networks from diverse functional genomic data. *Genome Biol* 2005; 6:R114.
4. Rhodes, D. R., Tomlins, S. A., et al. Probabilistic model of the human protein-protein interaction network. *Nat Biotechnol* 2005; 23:951–9.
5. Zhong, W., and Sternberg, P. W. Genome-wide prediction of *C. elegans* genetic interactions. *Science* 2006; 311:1481–4.
6. Ashburner, M., Ball, C. A., et al. Gene ontology: tool for the unification of biology. The Gene Ontology Consortium. *Nat Genet* 2000; 25:25–9.
7. Cherry, J. M., Adler, C., et al. SGD: *Saccharomyces* genome database. *Nucleic Acids Res* 1998; 26:73–9.
8. Kanehisa, M., and Goto, S. KEGG: Kyoto encyclopedia of genes and genomes. *Nucleic Acids Res* 2000; 28:27–30.
9. Efron, B., and Tibshirani, R. An introduction to the bootstrap. New York: Chapman & Hall, 1993.
10. Krogan, N. J., Cagney, G., et al. Global landscape of protein complexes in the yeast *Saccharomyces cerevisiae*. *Nature* 2006; 440:637–43.

11. Reguly, T., Breitkreutz, A., et al. Comprehensive curation and analysis of global interaction networks in *Saccharomyces cerevisiae*. *J Biol* 2006; 5:11.
12. Mewes, H. W., Amid, C., et al. MIPS: analysis and annotation of proteins from whole genomes. *Nucleic Acids Res* 2004; 32:D41–4.
13. Jansen, R., Greenbaum, D., et al. Relating whole-genome expression data with protein-protein interactions. *Genome Res* 2002; 12:37–46.
14. Watts, D. J., and Strogatz, S. H. Collective dynamics of 'small-world' networks. *Nature* 1998; 393:440–2.
15. Jansen, R., and Gerstein, M. Analyzing protein function on a genomic scale: the importance of gold-standard positives and negatives for network prediction. *Curr Opin Microbiol* 2004; 7:535–45.
16. Witten, I. H., and Frank, E. *Data Mining: Practical Machine Learning Tools and Techniques*. San Francisco, CA: Morgan Kaufmann, 2005.

Chapter 21

Comparing Algorithms for Clustering of Expression Data: How to Assess Gene Clusters

Golan Yona, William Dirks, and Shafquat Rahman

Abstract

Clustering is a popular technique commonly used to search for groups of similarly expressed genes using mRNA expression data. There are many different clustering algorithms and the application of each one will usually produce different results. Without additional evaluation, it is difficult to determine which solutions are better.

In this chapter we discuss methods to assess algorithms for clustering of gene expression data. In particular, we present a new method that uses two elements: an internal index of validity based on the MDL principle and an external index of validity that measures the consistency with experimental data. Each one is used to suggest an effective set of models, but it is only the combination of both that is capable of pinpointing the best model overall. Our method can be used to compare different clustering algorithms and pick the one that maximizes the correlation with functional links in gene networks while minimizing the error rate. We test our methods on several popular clustering algorithms as well as on clustering algorithms that are specially tailored to deal with noisy data. Finally, we propose methods for assessing the significance of individual clusters and study the correspondence between gene clusters and biochemical pathways.

Key words: Microarrays, mRNA expression, clustering, evaluation.

1. Introduction

Since the introduction of microarray technology, expression data have played an essential role in functional analysis of genes. Genome-wide experiments produce new data sets that provide a snapshot into the molecular machinery of the cell and pose new challenges in genomic research. For example, through differential analysis expression data can help in functional characterization of

genes and detection of “critical” genes that are involved in a specific process or related to a specific disease (1–6). Furthermore, expression data can hint into the underlying functional links in cellular gene networks, as it is generally accepted that genes that are similarly expressed under different experimental setups are functionally related. Co-expression can be explained by similar regulatory elements but there are other possible reasons. For example, genes might be expressed similarly because they are part of the same complex as interacting partners, or they can participate in the same cellular pathway.

However, individual (differential) or pairwise analysis of genes (co-expression) does not give the complete picture. Genes work in concert with each other, correlated and coordinated and groups of co-expressed genes often correspond to fundamental biological processes. Hence, there is a strong interest in algorithms that can detect coordinated groups of genes. Moreover, expression data are noisy and in many cases unreliable; there are many factors that may affect the experiment and the measurements, thus obscuring signals that might indicate relations between genes. Therefore, to enhance the signal and expose subtle functional links between genes, most studies of mRNA expression data employ clustering algorithms, hoping that the coordinated expression profiles of multiple functionally related genes would be more easily detectable than a single pair of similarly expressed genes. These groups of similarly expressed genes are assumed to have related biological functions and possibly correspond to cellular processes and pathways.

The emergence of expression data triggered many studies that focused on clustering. However, with the many clustering algorithms that are available at the disposal of the practitioner in data mining, it is not clear which one should be used to cluster this kind of data and which one would produce the most meaningful results.

In this chapter we discuss methods for assessing clustering algorithms for microarray data. We review existing methods and then introduce two criteria to evaluate the quality of the clustering results. We demonstrate our combined evaluation methodology by testing several popular clustering algorithms that were previously applied to the analysis of expression data. We study their effectiveness in detecting functionally related genes and the correlation with experimentally verified functional relationships extracted from pathway data, protein–protein interaction data, sequence data, structure data, and promoter data. We also explore unsupervised learning algorithms that are especially designed to deal with noisy data (such as microarray data). Finally, we describe a method for assessing the significance and correlation of individual clusters with cell-wide processes and pathways. We start with a brief overview of cluster analysis and then turn to describe our evaluation methodology in detail.

2. Methods I – Cluster Analysis

Cluster analysis is an unsupervised learning technique commonly used to organize samples into groups (clusters) so that samples within a cluster are more similar to each other than to samples in other clusters. There are many papers in the literature describing different approaches to clustering, some of which date back to the 1960's. In recent years we have seen a surge of papers describing clustering algorithms and their application to expression data, in search of groups of similarly expressed genes. In fact, more than 50 papers were published between 1998 and 2004, describing different strategies (**Note 1**). It is beyond the scope of this chapter to review all the different clustering algorithms. (For a review on general clustering algorithms *see* (7, 8). For a review on application of clustering algorithms to mRNA expression data *see* (9, 10)).

Here we focus only on several common techniques that are based on different approaches to clustering. Many of the existing clustering algorithms are variations of these general approaches, which we describe in more detail next. The main element of this paper is the evaluation methodology that we describe in **Section 3**. We do not attempt to test all existing clustering algorithms; however, our method can be applied to test other clustering algorithms and will be made available as a webserver at biozon.org.

In the following sections, the discussion assumes a hard clustering model (where each point belongs to exactly one cluster), but it can be easily generalized to the fuzzy clustering model (each point belongs to all clusters with different probabilities).

2.1. *Partitional Clustering Algorithms: K-Means*

Clustering algorithms can be generally divided into partitional clustering algorithms and hierarchical clustering algorithms. Partitional clustering algorithms usually select a clustering criterion or an objective function and set the desired number of clusters k . Then the objective function is optimized by searching for the best configuration (the best partition of sample points into clusters).

The popular k -means algorithm belongs to that category (using the total squared error objective function (**Note 2**)). An exhaustive search for the partition that minimizes the total squared error is impractical and the algorithm employs an iterative hill-climbing procedure to optimize the criterion function, as follows:

- Select an initial partition into k clusters.
- Loop:
 1. Compute the center of each cluster as the average of all samples assigned to it.
 2. Assign each sample to the closest cluster (the one whose center is closest). This will result in a new partition.
 3. If cluster memberships change, go to Loop; otherwise stop.

For squared error function, the optimal set of representatives is the cluster centers, and this scheme ensures that the global distortion decreases or remains unchanged with every iteration. However, a major problem with this algorithm and other similar variations is that they are very likely to become trapped in a local minimum. Also, their output may vary a great deal, depending on the starting point. The objective function may also significantly affect the output and algorithms that use this scheme with a different function can produce completely different results. Obviously, there is no single best objective function for obtaining a partition. Each criterion imposes certain structure on the data, but the true clusters are not necessarily recovered. This is especially true when the real number of clusters k is unknown, and one has to choose it arbitrarily.

2.2. Hierarchical Clustering Algorithms

Hierarchical clustering schemes can minimize the sensitivity to different initial parameters. In general, hierarchical clustering algorithms operate by obtaining a sequence of nested groupings. However, there are many clustering algorithms that apply hierarchical schemes, which do not necessarily result in nested groupings, such as the LBG algorithm (11) or deterministic annealing (12).

Perhaps the most popular hierarchical clustering algorithm is pairwise clustering. Pairwise clustering algorithms are of great interest when the data are given as a proximity matrix (pairwise similarities/dissimilarities). However, they are not confined to proximity data, and obviously can be applied to vector data as well (simply by first calculating the distances between the vectors). Pairwise clustering algorithms are often called graph-based clustering algorithms. In terms of graph theory the sample set is represented as a weighted undirected graph $G(\mathbf{V}, \mathbf{E})$, where \mathbf{V} is the set of vertices (one for each sample point) and \mathbf{E} is the set of edges (one between each pair of nodes). The weight of an edge $w(i, j)$ is a function of the similarity/dissimilarity between the nodes i and j . As described below, the steps of a pairwise clustering algorithm are equivalent to operations on the graph (e.g., deleting/adding edges) and the resulting clusters are equivalent to sets of nodes in the graph.

Almost all pairwise clustering algorithms share the same basic hierarchical scheme, which partitions the data into nested groupings using an agglomerative approach (algorithms that use divisive procedures are more computationally intensive and are less common). Given n samples with pairwise dissimilarities $w(i, j)$ ($1 \leq i, j \leq n$), the basic agglomerative clustering is as follows:

Denote by k the number of clusters, and let k_0 be the desired number of clusters.

- Let $k = n$ and initiate the clusters $C_i = \{i\}$ $i = 1, \dots, n$
- Loop:
 - If $k \leq k_0$, stop.
 - Find the nearest (most similar) pair of distinct clusters, say C_i and C_j .

- Merge C_i and C_j and decrement k by one.
- Go to loop.

Letting $k_0 = 1$ terminates the process when all samples are classified to one cluster. But usually a threshold is set and the process terminates at the level where the distance between the nearest clusters exceeds the threshold or when the desired number of clusters was reached. At any level the distance between the nearest clusters can provide the dissimilarity value for this level.

There are different ways to measure the distance between clusters. Each method can result in an algorithm with different properties. For example, the single linkage algorithm (also called the nearest neighbor algorithm) uses the minimal distance between members of the clusters

$$d_{min}(C_i, C_j) = \min_{i \in C_i, j \in C_j} \{w(i, j)\}$$

The complete linkage algorithm uses the maximal distance

$$d_{max}(C_i, C_j) = \max_{i \in C_i, j \in C_j} \{w(i, j)\}$$

And the average linkage uses the average distance between clusters

$$d_{ave}(C_i, C_j) = \frac{1}{|C_i||C_j|} \sum_{i \in C_i, j \in C_j} w(i, j)$$

2.3. Spectral Clustering Algorithms

Spectral algorithms consider clustering as a graph partitioning problem over the undirected graph $G(\mathbf{V}, \mathbf{E})$. A clustering $\mathbf{C} = \{C_1, C_2, \dots, C_k\}$ is a partitioning of \mathbf{V} into nonempty mutually disjoint subsets C_1, C_2, \dots, C_k where $k \leq n$.

There are several variations of spectral clustering algorithms. For example, the minimal cut clustering algorithm (13) uses the notion of cuts in graphs and seeks partitions into subgraphs such that the maximum cut across subgroups is minimized. A cut (\mathbf{A}, \mathbf{B}) in graph $G(\mathbf{V}, \mathbf{E})$ is a partition of \mathbf{V} into two disjoint sets of vertices \mathbf{A} and \mathbf{B} such that $\mathbf{V} = \mathbf{A} \cup \mathbf{B}$. The capacity of a cut is the sum of the weights of all edges that cross that cut, i.e.,

$$cut(\mathbf{A}, \mathbf{B}) = \sum_{i \in \mathbf{A}, j \in \mathbf{B}} w(i, j)$$

where $w(i, j)$ here denotes the similarity of i and j . The capacity of the cut can be used as a measure of similarity between these two sets. Note that there may be more than one possible cut between two given vertices. The minimal cut is the one which has the minimal capacity, and the bi-partitioning strategy that is adopted by the minimal cut algorithm is to split the graph along minimal cuts.

A variant of the minimal cut algorithm is the minimal normalized cut algorithm. This algorithm addresses one of the problems with the minimal cut algorithm, namely the minimal cut measure,

which is prone to create partitions with small sets of isolated nodes as subgraphs. To avoid this bias, the normalized cut algorithm (14) normalizes the capacity of the cut (\mathbf{A}, \mathbf{B}) by the total association of the set \mathbf{A} (which is the sum of weights incident on \mathbf{A}) and the total association of \mathbf{B} . This disassociation measure is called the normalized cut:

$$Ncut(\mathbf{A}, \mathbf{B}) = \frac{cut(\mathbf{A}, \mathbf{B})}{assoc(\mathbf{A}, \mathbf{V})} + \frac{cut(\mathbf{A}, \mathbf{B})}{assoc(\mathbf{B}, \mathbf{V})}$$

where

$$assoc(\mathbf{A}, \mathbf{V}) = \sum_{i \in \mathbf{A}} \sum_{k \in \mathbf{V}} w(i, k)$$

The algorithm looks for partitions with small $Ncut$ value and the graph is partitioned recursively so as to minimize the normalized cut criterion at each level until $Ncut$ exceeds a predefined threshold.

2.4. Algorithms for Clustering of Noisy Data

Expression data are very noisy and in some cases unreliable, which might greatly affect clustering algorithms that are based purely on the pairwise distances between expression profiles. In this section we describe two clustering algorithms that are designed to work effectively with noisy data. The first algorithm (strict clustering) attempts to improve the clustering by maintaining a certain level of confidence about all pairwise relationships in a cluster. The second (iterative clustering) diverges from the original representation of genes as expression profiles and gains its power by employing a dynamic representation that samples the complete space of expression profiles to determine the whereabouts of each individual expression profile.

2.4.1. Strict Clustering

In strict clustering, clusters are constrained to be strongly connected. A threshold is chosen first and for every cluster we require that the average similarity of a node to every other node in that cluster is more significant than the threshold chosen. This procedure generates clusters of very high quality in the sense that we have high confidence in the cluster membership of each individual gene. The algorithm can be viewed as a heuristic approximation to the max-clique problem, where the resulting clusters are in essence pseudo-cliques (pseudo, as the average similarity of each member with all other members is above the threshold, as opposed to all the individual pairwise similarities).

Strict clustering is implemented by first considering the node with the maximal number of neighbors and grouping it and all its neighbors into one candidate cluster. The node with the worst average within-cluster similarity is then considered for elimination. If its average similarity is below the significance threshold, then the node is thrown out. The process is continued until all nodes satisfy the above condition. Other nodes are considered for inclusion if

their average similarity score and the average cluster similarity score exceed the threshold, adding the maximally similar node at a time. This procedure is repeated with the remaining nodes until all nodes are exhausted. The clustering procedure ends when every node is in a cluster (possibly a singleton).

2.4.2. Iterative Clustering

The iterative clustering method we tested is an hierarchical algorithm based on iterative modification of data representation (15). The algorithm begins with the proximity matrix of the pairwise similarities. After preprocessing (see **Section 4.4**), the clustering procedure precedes in two steps. In the first step each row is normalized so its Euclidean norm is one (**Note 3**). Next, a new estimate of the distances between genes is calculated from the proximity matrix. Specifically, the distance between genes i and j is estimated by the distance between rows i and j of our proximity matrix (**Note 4**). This procedure is repeated until the matrix converges to a binary matrix containing only zeros and some positive constants. In most cases this final matrix can be permuted to a block diagonal matrix with two blocks. If this is the case, the two blocks are considered as a split of the data into two clusters. If the final matrix converges to more than two blocks, one of the blocks is chosen arbitrarily to be one of the clusters and the remaining genes are classified to the other cluster. This process is repeated with each one of the two clusters generated. A cross-validation protocol is employed (15) to assess the stability and the prominence of the split and splitting is stopped when the sets disagree (see **Section 4.5**).

3. Methods II – Assessment of Clusters

Clustering algorithms can be very effective in discovering patterns in unlabeled data. However, when applied simple-mindedly, they can lead to false conclusions. Clearly, the application of any clustering algorithm will result in some partitioning of the data into groups, and the choice of the clustering algorithm may greatly affect the outcome. Moreover, clustering algorithms are likely to become trapped in a local minimum and their output may vary a great deal, depending on the starting point. Most importantly, determining the “correct” number of clusters (classes) within the data set is considered one of the most difficult problems in unsupervised learning, and one usually presets this number arbitrarily, or selects it based on hierarchical cluster analysis. However, in the absence of a natural criterion by which to stop the process of splitting or merging data points, usually an external user-specified parameter is used (e.g., when the distance between clusters exceeds

a certain threshold). Consequently, the clustering (the “recovered structure”) does not necessarily fit the “real” structure of the data, and its validity is questionable.

Naturally, with such a large variety of clustering algorithms to choose from and the explicit dependency on the parameters’ values, one could ask which method is the best and whether the results produced with one algorithm are more meaningful than the other. This problem is not unique to microarray data analysis, but rather is common to any learning task that is involved with clustering and the issue has occupied the machine-learning community for many years now, and many different strategies have been proposed.

In general, there is no natural definition according to which the quality and validity of the clustering can be assessed, and external or relative parameters are used to adjust the number of clusters and assess the cluster structure. There is no single approach that is generally perceived as universally acceptable and many indices of validity have been proposed in the literature (*see* (7) for a discussion). In general, there are four main types of validation: (i) External validation compares the recovered structure to an a priori structure and tries to quantify the match between the two. (ii) An internal validation test seeks to determine if the structure is intrinsically appropriate for the data. This may be done, for example, by measuring the compactness and isolation of clusters. A related approach is the minimum description length (MDL) criterion (16) where the total clustering cost takes into account the distortion as well as the model’s complexity (which is a function of the number of parameters needed to describe the model). (iii) A relative test compares two different structures (e.g., clustering structures for $k = 4$ clusters and for $k = 5$ clusters) and measures their relative merit. (iv) A cross-validation test checks the parameters of the clusters against independent validation data. Statistical tests are applied to verify that the “match” between the parameters of the two sets is statistically significant. Cross-validation can prevent overfitting of the model to the training data and the statistical tests can also provide a useful estimate of the generalization error.

Several studies proposed and applied external indices to compare and evaluate clusterings of gene expression data. For example, Bolshakova et al. (17) propose an external index that assesses cluster validity based on similarity scores extracted from the Gene Ontology (GO) (18). Another approach is described in (19). This approach also incorporates GO annotations when clustering genes using expression data. To estimate the quality of the clusters the authors applied the Davies & Bouldin index, which is defined as the ratio of within-cluster scatter to between-cluster separation, summed over all clusters. The authors in (20) used statistical natural language processing techniques to utilize information about gene function in the published literature and assess the biological significance of clusters. Their method, neighbor divergence,

assesses whether the genes within a group are functionally similar by computing the similarity between the word vector representations of the corresponding paper abstracts. Another knowledge-driven method is described in (21). The authors represent each gene as a vector of biological attributes (e.g., membership in functional classes derived from GO annotations) and project the vectors onto a one-dimensional space such that the cluster separation is maximized (measured by the ratio of between-clusters and within-clusters variance). The clustering is then scored by computing the p -value of the observed variance in the projected vectors for each cluster.

A few other studies used internal indices to assess gene clusters. For example, a software package that tests clustering solutions using multiple indices that measure cluster separation is described in (22). The authors in (23) propose cluster validity measures based on subspace projection methods. The original gene expression data are projected into lower dimensional subspaces using the Johnson-Lindenstrauss method, which approximately preserves distances between examples, and the stability of the clusters is estimated by comparing them with the clusters discovered in randomly projected lower dimensional subspaces. Another internal method is explored in (24), where the statistical significance of each cluster is computed by comparing its density to the local background. The validity of a cluster is based on the probability of it having essentially the same distribution as the neighboring points. A low probability indicates a higher-quality cluster. Yet another method is introduced in (25), where the authors assess the clustering using measures of cluster reproducibility.

Other studies used cross-validation techniques and related resampling methods. For example, the method for validating clusters in (26) is based on cross-validation. The paper applies a clustering algorithm to the data from all but one experimental condition. The left-out condition is used to assess the predictive power of clusters by measuring the variation in the left-out condition and comparing it to the variation observed in random clusters (with lower values anticipated for “meaningful clusters”). In (27) the stability of individual clusters was assessed with scores that were computed using subsampling techniques. Another related method is described in (28), using re-sampling methods based on bagging to assess clustering procedures. Bagging is an ensemble method in which multiple bootstrap learning sets are generated by sampling with repetitions. A model is generated (learned) for each set and the final prediction is made by combining the predictions of the individual models. The authors apply a clustering algorithm to each set and the solutions are combined by voting. The cluster votes are used to assess the confidence of cluster assignments. Finally, in (29) the authors propose a statistical resampling method to assess the reliability of gene clusters identified with hierarchical clustering algorithms. A tree is generated for

each resampled data set and a confidence value is computed for each node in the original clustering tree. Clusters occurring in majority of the resampled trees were deemed more reliable.

While these methods can be effective in detecting “good” structures, each one is sought with its own problems. For example, to determine if a specific external index for the clustering structure is “unusual”, or surprising, one must obtain (by theory or simulation) the baseline distribution of the index for random data. A clustering is considered valid if the index is unusually high, as measured with respect to the corresponding baseline distribution. However, establishing the baseline distribution requires purely random data that match the characteristics of the real data, or extensive statistical sampling. In addition, external tests can be applied only when a prior structure is known. Internal indices also require estimating the background distribution. In addition, in order to determine if the clustering structure is surprising, it should be compared to the best clustering of random data. Practically, these requirements complicate the application of internal validation tests. Relative indices are usually a derivative of internal indices and they suffer from the same problems. The MDL criterion requires a parametric model, which is not always explicitly defined (for example, with graph-based clustering). Its dependency on the external parameters can result in over-sensitivity to perturbations in the data or in the parameters. As with the MDL criterion, cross-validation also requires a certain parametric model, which limits its applicability. It also requires relatively large data set for training and validation. Most importantly, no index is guaranteed to work under all scenarios, and even if a specific individual index performs well on some data sets it might not perform well on others.

3.1. Our Approach – The Double Index Assessment

Our goal is to determine whether a structure is meaningful and cannot reasonably be assumed either to have occurred by chance or to be an artifact of the clustering algorithm. As pointed out in the previous section, no single index provides a global solution and therefore we suggest a combined approach that uses two criteria to assess the quality of different clustering models. The first is based on internal index of validity, specifically the complexity of the model and its ability to explain new instances. The second is based on external index of validity, specifically the correlation with the experimental data. We corroborate the evidence as provided by the two measures to select the best clustering.

3.2. Internal Index of Validity – The Bayesian Approach

We consider each possible clustering (partition) of the data as a unique model, denoted by h . Posed in probabilistic terms, the problem of selecting the best model (best clustering) can be phrased as a problem of maximizing the posterior probability of the model given the data, i.e.,

$$P(h/D) = \frac{P(h)P(D/h)}{P(D)}$$

The optimal hypothesis h^* is the one that maximizes $P(h/D)$ (minimizes $-P(h/D)$) or equivalently

$$h^* = \arg \min_b [-P(h)P(D/h)] = \arg \min_b [-\log_2 P(h) - \log_2 P(D/h)]$$

Note that this Bayesian approach resembles the minimum description length principle. The description length of a given model (hypothesis) and data is defined as the description of the model plus the description of the data given the model. This heuristic calls for selecting models that minimize the overall description length, thus compromising between complex and overly specific models and simple models that are too general and fail to provide reliable and accurate description of the data. By the Shannon theorem $-\log_2 P(x)$ is related to the shortest description of a string x and therefore the first term can be considered as the model description length, while the second term $\log_2 P(D/h)$ is the likelihood of the data given the model and can be considered as the description of the data given the model (this is a measure of the uncertainty that is left in the data, given the model).

The second term is simply the minus log-likelihood of the data given the model

$$\log_2 P(D/h) = \log_2 \prod_{i=1}^n p(\mathbf{x}_i/h) = \sum_{i=1}^n \log_2 p(\mathbf{x}_i/h)$$

In our hard clustering model, each sample is classified to a single cluster C_i , whose prototype is estimated as the average of all samples that are classified to that cluster. The prototype is assumed to be corrupted by some normally distributed noise, with mean zero and standard deviation that can be estimated from the data. Thus, each cluster is parametrized by a d -dimensional vector of means and standard deviations (each one corresponds to a different experiment, e.g., a different time step). The probability of an individual sample to belong to cluster C_i is given by the product of the d normal distributions characterized by the individual means and standard deviations at each time step

$$\begin{aligned} p(\mathbf{x}/C_i) &= p((x_1, x_2, \dots, x_d)/C_i) \\ &= p(x_1/\mu_{i1}, \sigma_{i1})p(x_2/\mu_{i2}, \sigma_{i2}) \dots p(x_d/\mu_{id}, \sigma_{id}) \end{aligned}$$

where each $p(x_j/\mu_{ij}, \sigma_{ij})$ is a one-dimensional normal distribution (j being the experiment index). This equation can be easily generalized to the soft clustering case, where each gene has a nonzero probability to belong to each one of the clusters.

The first term $-\log_2 P(h)$ is the approximated model description length. It has been suggested (*see* Bayesian Information Criterion (30)) that the model complexity can be approximated using

0.5 log n bits per parameter. However, this ignores the uncertainty we have about the exact value of each parameter (that can vary between parameters). Here we estimate the term $-\log_2 P(b)$ by calculating how many bits are needed to specify (localize) the coordinates of the prototypes (prototype means), taking into account this uncertainty. We make a distinction between singletons and non-singleton clusters. For singleton clusters we use 0.5 log n bits per parameter (one per dimension) while for non-singleton clusters the resolution depends on the uncertainty in the value of each parameter. Estimating the uncertainty is done as follows: Since the standard deviations within clusters are not explicitly used in the clustering algorithm, we can assume that each cluster in our model is characterized by only d parameters (the prototype means), and the standard deviations can help in determining the exact whereabouts of each one. Since each parameter (the mean) is estimated as a sum of i.i.d. random variables drawn from an arbitrary probability distribution with mean, μ and variance σ^2 (estimated from the data) as in $\theta = \frac{1}{n} \sum_{i=1}^n X_i$, then by the central limit theorem we know that as $n \rightarrow \infty$ then $\theta \rightarrow \mu$ with standard deviation $\sigma_\theta = \frac{\sigma}{\sqrt{n}}$. This standard deviation (in bits) is a measure for the uncertainty in the value of the estimator. The overall uncertainty is a sum over the uncertainties of all parameters for all clusters.

3.3. External Index of Validity – The Relation Graph

While the internal validity index described above can help in discerning better models, it does not necessarily produce clusterings that best describe the experimental data. Moreover, as this principle is a heuristic, it can only guide the designer by pointing to more effective models; however, usually the designer is left with several models, all seem to describe the data equally well (*see Section 5*).

To reduce this set to a single model, we re-evaluate the clustering results and select the most significant one given the experimental data using the relation graph described in **Section 4.2**. Specifically, we count the number of true edges within each cluster and estimate the probability to obtain this partition of edges amongst clusters by chance. In other words, the significance of the clustering is determined by how much this partition deviates from the partition one would expect to get by assigning edges to clusters and between clusters at random.

Formally, we are given a cluster set $C = C_1, C_2, \dots, C_k$ with n_1, n_2, \dots, n_k genes and e_1, e_2, \dots, e_k edges, which partitions the total set of true edges such that t_i edges fall in cluster i ($1 \leq i \leq k$). In addition, we have e_{cross} edges across clusters, of which t_{cross} are true. The probability to obtain this partition by chance is

$$P(t_1, t_2, \dots, t_k, t_{cross}/C) = \binom{T}{t_1 t_2 \dots t_k t_{cross}} p_1^{t_1} p_2^{t_2} \dots p_k^{t_k} p_{cross}^{t_{cross}}$$

where $E = \sum_i e_i + e_{cross}$ is the total number of edges, $T = \sum_i t_i + t_{cross}$ is the total number of true edges, and $p_i = e_i/E$ (we refer to set $\{p_i\}$ as the random type, and to set $\{t_i\}$ as the sample type).

This model of the edge distribution may seem odd at first since one is used to thinking about partitioning the set of genes and not the set of edges. One can imagine choosing an edge at random is the same action as choosing two genes at random. This must result in either the two genes landing in the same cluster or that these genes lie in different clusters. The multinomial distribution is a model of this edge picking process.

However, this probability in itself is not enough to ascertain the significance of the overall partition of genes into clusters. To assess the significance one needs to know how probable it is to get partitions that are less likely by chance than the one in question (in other words, one needs to know the weight of the tail of the distribution). Alternatively, one can use a concentration of measure result such as standard deviation, when known, to determine significance. Given a significance measure, one can use it as an external validation index and pick the clustering with the minimal total random probability (maximum significance).

A common significance measure for the multinomial distribution is the following statistic

$$Q = \sum_i \frac{(t_i - Tp_i)^2}{Tp_i} + \frac{(t_{cross} - Tp_{cross})^2}{Tp_{cross}}$$

This statistic has an approximate chi-square distribution with k degrees of freedom. However, as many clusters are fairly small, the underlying statistical assumptions are violated and the chi-square approximation does not hold. Instead, we use an algorithm that was recently introduced in (31) for efficient exact computations of p -values for the multinomial distribution using the likelihood ratio statistics. Given two probability distributions P_1 , P_2 and a type $T = \{t_i\}$, the likelihood ratio of the type is defined as

$$-2 * \log \frac{P(T/P_1)}{P(T/P_2)}$$

In our case P_2 is defined based on the empirical sample type, and P_1 is the random type. High positive ratio indicates that T is more similar to the observed type than to the random type, and when considering the two hypotheses it is less likely to emerge from the random type. We implemented this algorithm to compute the probability that a type sampled according to the random type will obtain a higher likelihood ratio than that of the observed type.

4. Methods III – Experiments

4.1. Data

The main entities of our study are genes, their expression profiles and sets of gene relationships that are believed to be the underlying reason for co-expression. Our model system is the yeast genome that was selected because of the myriad of information which is available about genes, interactions and cellular processes. In our experiments we used the well-known yeast time-series data set (*I*); however, our tests can be applied in the same way to clusterings that are obtained from other data sets. The time-series data set of (*I*) measures the expression of genes at different time points of the cell cycle. The data set contains measurements for four cell cycles, and in our analysis each ORF is represented by concatenating the measurements from all four cell-cycles together, for a total of 73 measurements.

4.2. The Reference Set (For External Validation Index)

When evaluating the clustering results using the external index of validity (*see* Section 3.3), we need a data set of experimentally verified relationships. Four possible gene relationships are considered in this work: protein–protein interactions, pathway membership, promoter co-regulation, and sequence homology. Each of the four relationships can be thought of in terms of a graph. In this graph each node represents a gene and an edge exists between the nodes if the genes are functionally linked. These subgraphs are compiled together into a single graph (called the relation graph) in which two nodes are connected if a known relationship exists between the two corresponding genes. Table 21.1 lists the number of genes and edges in each data set. For more details *see* (32).

Table 21.1
The relation graph. Number of genes and edges (true relationships) in each data set. The sum of the number of edges in the four categories does not equal the total number of edges because two genes may have multiple types of edges between them, but this relationship is counted only once in the total set

Relation type	#genes	#edges
Interaction	3592	5339
Sequence	3092	19074
Promoter	213	2439
Pathway	642	15789
Total	5079	41902

4.3. Similarity Measures

To cluster genes based on expression data, one has to choose first a similarity or distance function for expression profiles. The choice of the similarity function can greatly affect the clustering results and therefore it has to be chosen carefully.

The two most popular measures used in studies of mRNA expression data are the Euclidean metric and the Pearson correlation. For two expression vectors \mathbf{V} and \mathbf{U} of dimension d , we denote by $Dist(\mathbf{V}, \mathbf{U})$ the normalized Euclidean metric

$$Dist(\mathbf{V}, \mathbf{U}) = \sqrt{\frac{1}{d} \sum_{i=1}^d (V_i - U_i)^2}$$

and by $Corr(\mathbf{V}, \mathbf{U})$ we denote the Pearson correlation of the two vectors

$$Corr(\mathbf{V}, \mathbf{U}) = \frac{1}{d} \sum_{i=1}^d \frac{(V_i - \langle V \rangle)(U_i - \langle U \rangle)}{\sigma_V \sigma_U}$$

Due to missing data the dimension of each vector (expression profile) can vary. Instead of using extrapolation methods, we decided to use only the available data; to avoid errors one might introduce by extrapolation. When comparing a pair of vectors, the dimension d is defined as the number of features both vectors have in common.

In a previous study (32) we tested different similarity measures between expression profiles (including the Euclidean metric and Pearson correlation) and compared their effectiveness in terms of their ability to detect functional links between genes, such as protein–protein interactions, pathway membership, promoter co-regulation, and sequence homology. Of all measures that we tested, the z-score-based measure which combines the Euclidean metric and the Pearson correlation was one of the most effective ones (referred to as EucPear). Formally, the two distance measures are converted to z-scores based on background distributions that are generated by permuting one of the vectors and re-computing the distance between the permuted vectors. This method provides reliable measure of significance as it adjusts to the “compositions” of the vectors compared. The z-scores are then summed to determine the final similarity score. Since higher correlation scores are assigned positive z-scores and smaller Euclidean distances are assigned negative z-scores, the final score is defined as

$$EucPear(\mathbf{V}, \mathbf{U}) = Z[Corr(\mathbf{V}, \mathbf{U})] - Z[Dist(\mathbf{V}, \mathbf{U})]$$

with higher scores indicating stronger similarity.

It should be noted that the most effective measure we analyzed in (32) is the new mass–distance measure. However, for the sake of simplicity we used the more standard measures in this study.

4.4. Pre-Processing Similarity Data

When applying each of the clustering algorithms to the expression data sets, some preprocessing is necessary. Expression data are very noisy and incomplete. In our analysis genes for which expression data were not available were removed altogether (a total of 593 genes in the time-series data of (1)). Even after excluding these, it might not be possible to compute the similarity score between all remaining genes, for example, when the expression profiles associated with genes i and j are over non-overlapping subsets of experiments. To recover missing values we considered intermediate genes. Specifically, a missing value at position i, j is computed as the maximum similarity over all possible intermediate genes

$$w(i, j) = \max_k \frac{w(i, k) + w(k, j)}{2}$$

When applied to the time-series data set, the procedure recovered 7262 positive similarity scores. With spectral clustering, after recovering the missing scores all negative scores were set to zero, since spectral algorithms assume a non-negative proximity matrices. By setting negative scores to zero some information is being disregarded; however, negative similarity scores indicate that the two genes have very different expression profiles and therefore can be simply considered as non-similar (i.e., zero similarity score).

4.5. Algorithms

In total we tested six clustering algorithms, all were implemented in-house. We tested three conventional clustering methods: the k -means algorithm and two hierarchical clustering algorithms (single-linkage and average-linkage). We also tested the normalized-cut spectral clustering algorithm and two noise-tolerant algorithms: strict clustering and iterative clustering. All experiments were done using the EucPear proximity matrix, excluding the k -means algorithm that used the Euclidean metric (**Note 5**).

The data set contains many genes with expression profiles that are unique and are not similar to any other gene, and therefore the clustering solutions contain many singletons. To reduce the impact of these singletons on the evaluation, we first ran strict clustering and collect all genes that are members of non-singleton clusters (a total of 3775 genes). We then ran all other clustering algorithms only on this subset of 3775 genes.

5. Results

There are two types of comparisons when assessing clustering algorithms. One could compare the performance of the same algorithm with different sets of parameters (parameter optimization). For example, the cross-validation threshold in the iterative

clustering, or the Ncut parameter in spectral clustering, or the number of clusters with k -means and hierarchical clustering. The second type of comparison would take the best clustering obtained with each method and compare them to each other, to find the best clustering overall (best clustering). In discussing the results we focus mostly on the first type.

5.1. Parameter Optimization

5.1.1. Strict and Iterative Clustering

With strict clustering there is one parameter – the threshold (for the average similarity) below which a sample point is removed from a cluster. The threshold was chosen based on an analysis of the distribution of EucPear distances (*see* (33)) and set to 7.5. The iterative clustering algorithm was run at different cross-validations levels, starting from 0.98 (the percentage of points on which the two validation sets agree). The splitting was terminated when the validation score fell below 0.6 or when the size of the cluster fell below 4, and all splits were stored for further evaluation.

In Fig. 21.1a we plot the description length of the different clustering solutions we produced with the iterative clustering algorithm, using thresholds in the range of 0.6 and 0.98. As the graph indicates, the description length decreases as the threshold decreases (note the major decrease around 0.83) until it reaches a plateau around 0.75. Without any further information one might suggest 0.75 as the optimal threshold, were the plateau starts. However, there is no clear reason to prefer 0.75 over 0.6, for

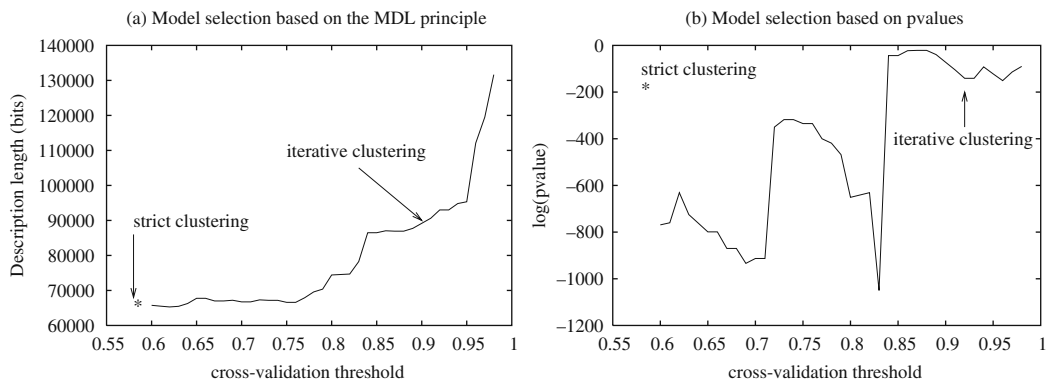


Fig. 21.1. Strict and iterative clustering: (a) Model selection based on the posterior probability/MDL principle. We tested all clustering solutions that we produced with the strict clustering algorithm and the iterative algorithm (using a variable threshold). For each solution we computed the overall description length (the sum of the description length of the model and the data given the model). This is proportional to the posterior probability of the model (*see* text). As the graph indicates, all models with thresholds ranging from 0.6 to 0.75 can be considered good models, as well as the model generated with the strict clustering algorithm (denoted by “*”). (b) Model selection based on p -value. For each model we compute the significance (p -value) of the sample type (the observed distribution of true edges), given the clustering model (*see* text). Both clustering techniques perform well and the resulting clusters are strongly correlated with the experimental data sets (strict clustering is denoted by a “*”). Of all models, those that correspond to thresholds of 0.83 (with 66 clusters) and 0.69 (213 clusters) are the most significant ones. However, corroborating the evidence from the MDL computations, we conclude that the best model is obtained at threshold of 0.69.

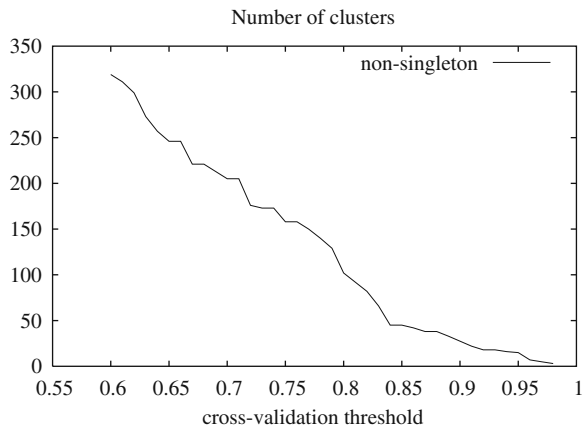


Fig. 21.2. Number of clusters as a function of the cross-validation threshold. Note that there are no singletons in this data set.

example, based on the MDL principle alone, and each one of these clusterings is a viable model. However, the clustering structure changes quite drastically, ranging from 158 clusters (at threshold 0.75) to 319 clusters (at threshold 0.6), as is depicted in **Fig. 21.2**.

The second criterion helps to reduce this set of models to a single best one. We compute the p -value for each one of the clustering solutions and compare them based on their significance (their p -value). The results are given in **Fig. 21.1b**. Note that the most significant clusterings are those produced with thresholds 0.83 and 0.69. The first corresponds to a major decrease in the MDL (*see Fig. 21.1a*); however, it is the second one that obtains the better score when considering both the MDL principle and the p -value.

5.1.2. Spectral Clustering

With the normalized-cut spectral clustering algorithm, the original graph was partitioned recursively until all clusters contained at most five nodes. One can stop the splitting at any point before reaching this threshold on the cluster size. Since the Ncut value increases as the splitting progresses, is it possible to set a threshold on the Ncut value and halt when the Ncut value exceeds this threshold. We produced partitions for Ncut threshold values ranging between 0.5 and 1.4 (in steps of 0.05).

We scored all solutions using both the MDL measure and the p -value measure. Our evaluation suggests that the model obtained with Ncut threshold of 0.9 is a much better model than the other ones (**Fig. 21.3**). This model has 93 clusters. It should be noted that even slight modifications in the value of the threshold change the number of clusters drastically (*see Fig. 21.4*), and to find the best model the experiment has to be repeated with finer

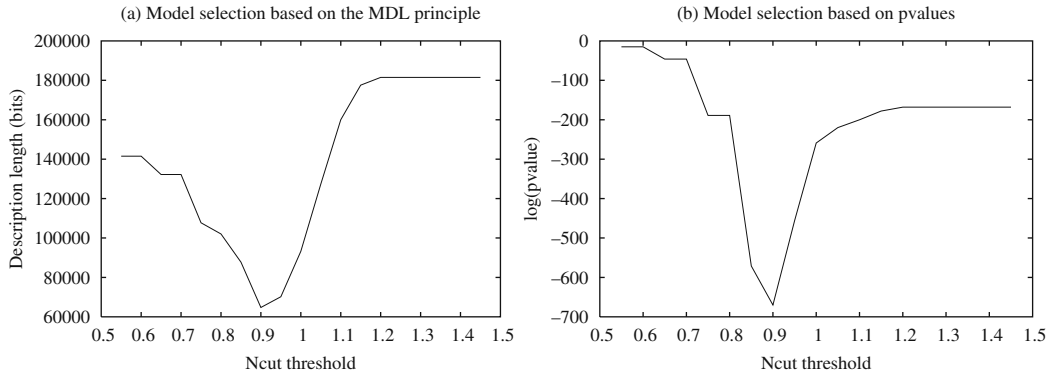


Fig. 21.3. Spectral clustering: Model selection based on (a) the MDL principle and (b) the p -value measures.

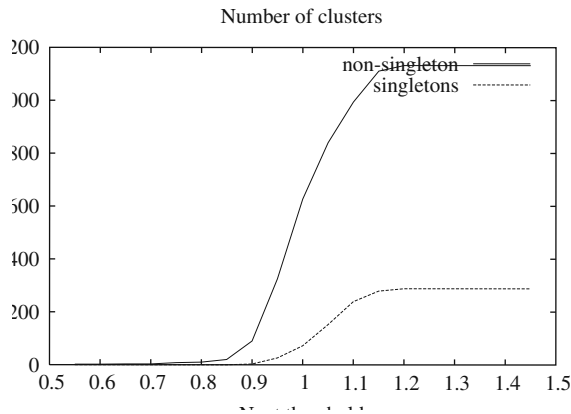


Fig. 21.4. Number of clusters as a function of the Ncut threshold.

increments in the parameter value. We did not repeat the experiment, but the graphs clearly indicate that the two measures we propose can pinpoint the best model.

5.1.3. Hierarchical Clustering

With hierarchical clustering the algorithm outputs a tree of nested clusters. Each level of the tree corresponds to a different grouping and it is necessary to assess the clustering at each level. For the single linkage algorithm we examined all clusterings between levels 1000 and 3750, in steps of 50 (to save computation time, the ranges were determined dynamically as results were accumulating; clusterings outside these ranges were far from optimal). The main problem with the single-linkage algorithm is the large number of singletons (see Fig. 21.5a). Of the 1,000 clusters at level 1,000 only 52 are non-singletons and at 3,750 clusters, only 14 are non-singletons. The MDL index is not very effective in this case, as it is dominated by the large number of singletons. However, the

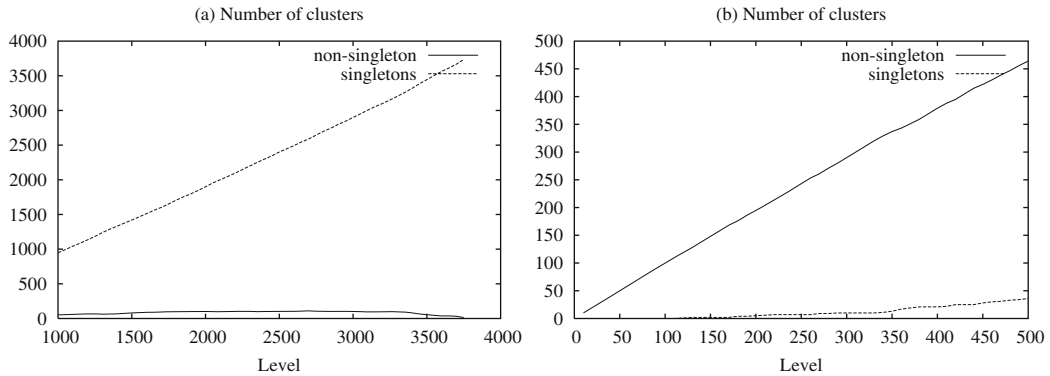


Fig. 21.5. Number of clusters at different levels of hierarchical clustering. (a) Single-linkage (b) Average-linkage.

p -value measure suggests that the clustering at 3,300 is relatively better than other models (Fig. 21.6b). However, it is still dominated by singletons and only 96 clusters are non-singletons.

For the average-linkage algorithm we assessed the clusterings with the number of clusters (level) ranging from 10 to 500, in steps of 10. Note that there are much fewer singletons (Fig. 21.5b) and the evaluation suggests that the best model is obtained with about 220 clusters (Fig. 21.7). We did not attempt to find the best possible clustering, just to demonstrate the usefulness of our tool. However, the analysis can be easily repeated at a finer granularity to determine the best clustering.

5.1.4. *k*-Means

With the *k*-means clustering algorithm one must decide on a number of clusters a priori. We did the clusterings for different values of k ranging between 10 and 500, in steps of 10. We used the original raw expression data set and distances were computed using the Euclidean metric. As expected, the number of singletons increases as k increases (Fig. 21.9), but not as much as with single-linkage clustering.

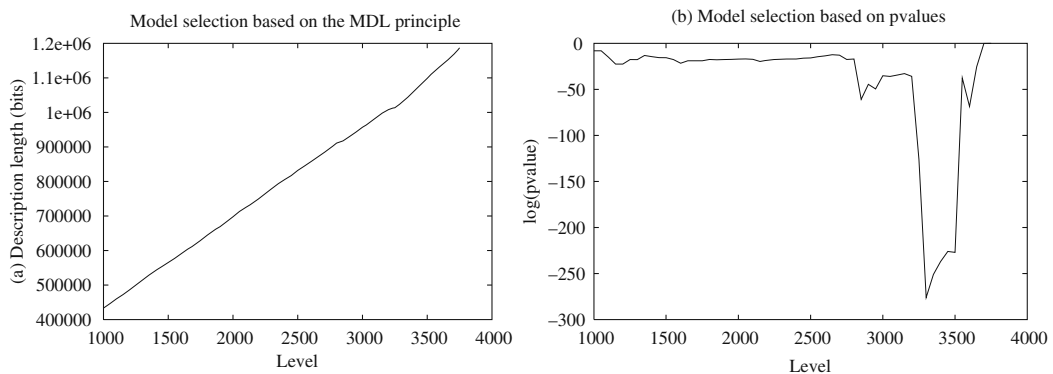


Fig. 21.6. Single-linkage clustering: Model selection based on (a) the MDL principle and (b) the p -value measures.

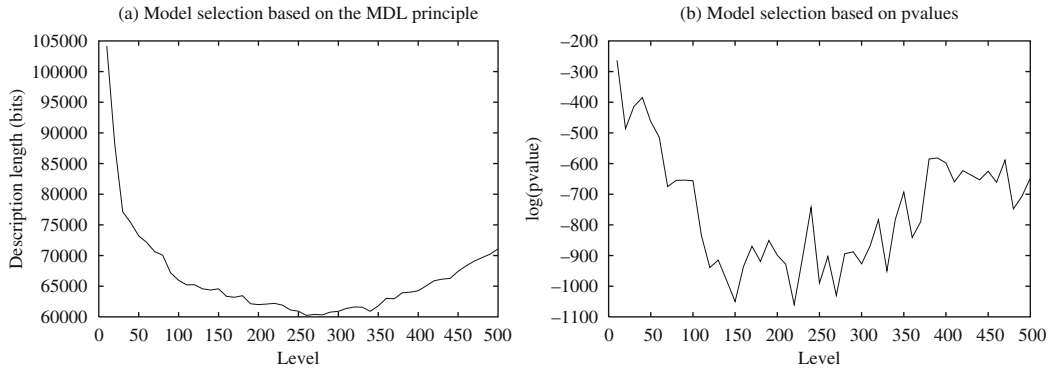


Fig. 21.7. Average-linkage clustering: Model selection based on (a) the MDL principle and (b) the p -value measures.

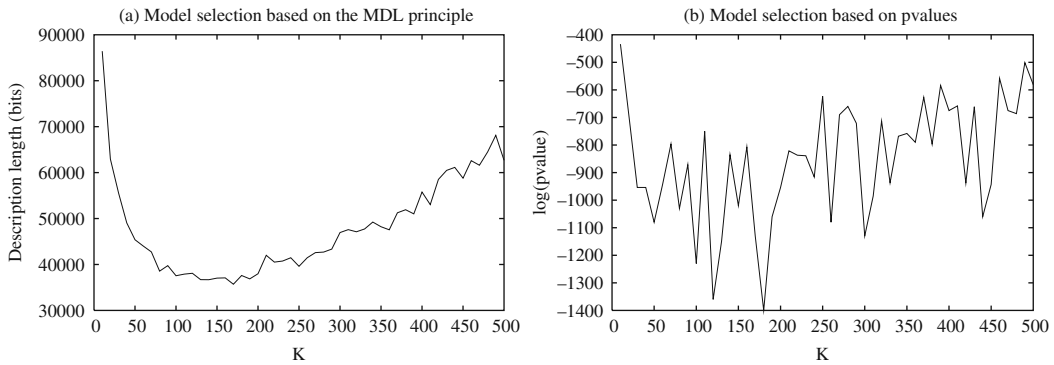


Fig. 21.8. K-means clustering: Model selection based on (a) the MDL principle and (b) the p -value measures.

Note that the p -value graph fluctuates substantially (Fig. 21.9b), more than with other algorithms. This is because the k -means algorithm starts from a different random configuration for every k , while other algorithms gradually refine the clustering at a given

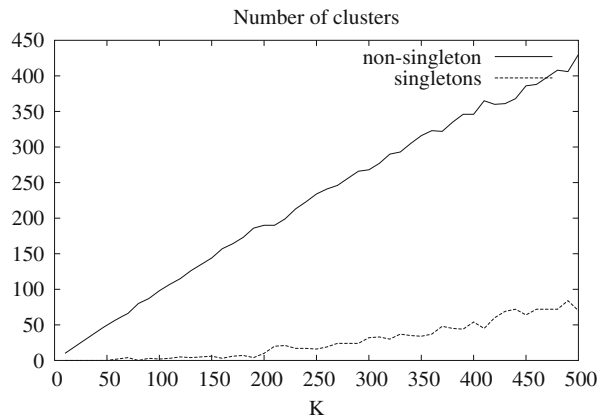


Fig. 21.9. Number of singleton and non-singleton clusters as a function of K .

level to obtain the clustering at the next level. Hence, clusterings obtained for close values of k can be quite different. Running the algorithm repeatedly with different starting points and picking the best model, or employing a hierarchical scheme as in (11), can improve the stability of the algorithm.

Despite the noisy results the graphs clearly indicate some trends. The MDL graph suggests a range of models (between 100 and 200), but the p -value graph indicates that the clustering generated with $k = 180$ is more significant than others (Fig. 21.8). Another good clustering is obtained at $k = 120$.

5.2. Best Clustering

Determining the best clustering requires extensive tests of many clustering algorithms with multiple configurations. The tests we ran in the previous section were by no means exhaustive. Our goal was to demonstrate that we can compare different configurations and clustering algorithms and pick the better models. Actually finding the best clustering would require running these algorithms with many more configurations before we can reach clear conclusions. Furthermore, with iterative clustering, spectral clustering and hierarchical clustering the optimal clustering is likely to be composed of clusters at varying thresholds. However, this means an exponentially large set of possible clusterings to test, and to make this problem tractable we reduced the set to those with a uniform threshold. Nevertheless, with the results we got we can already reach some conclusions regarding the effectiveness of the different algorithms and the quality of their results, when applied to the yeast cell cycle expression data set.

Using the two indices discussed above we compare the best clustering we obtained with each algorithm. According to the MDL index the k -means algorithm seems to perform very well (Fig. 21.10a). However, this is misleading, as the k -means algorithm

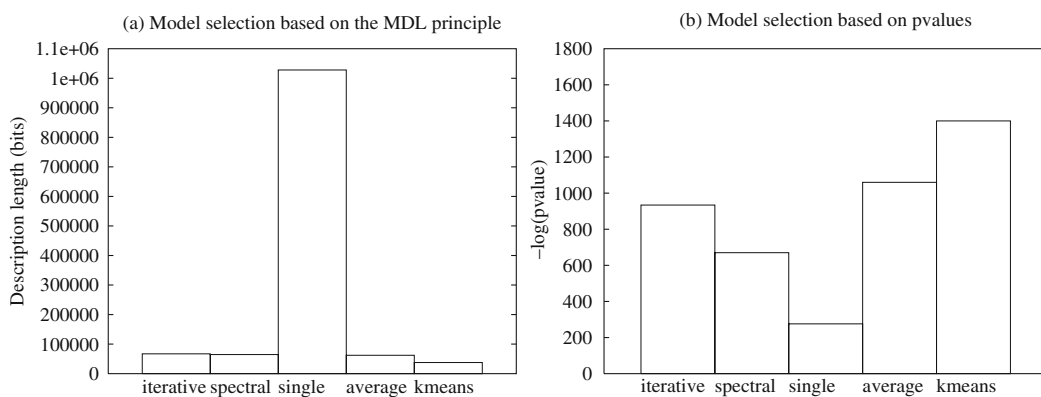


Fig. 21.10. Comparison of clusterings based on (a) the MDL principle and (b) the p -value measures. Lower description lengths and higher $-\log(p\text{-value})$ indicate better models.

uses the Euclidean metric and explicitly minimizes the squared error, which is directly related to the likelihood component of the MDL measure (*see* **Section 3.2**). The spectral algorithm and the average-linkage algorithm produce models that are comparable in terms of the MDL, and the iterative algorithm performs as well. The single-linkage algorithm performs poorly compared to all other algorithms.

The p -value index suggests that the best clustering is obtained with the k -means algorithm (**Fig. 21.10b**), followed by the average-linkage algorithm. The iterative algorithm comes third followed by the somewhat disappointing results of the spectral clustering algorithm. The single-linkage algorithm is far behind, its so-so p -value affected by the many singletons.

5.3. Assessment of Individual Clusters

Clearly, with any clustering solution not all clusters are equally significant. One can compare the clusters across different algorithms and focus on the clusters that are more stable (detected with significant overlap with multiple algorithms). However, this requires running multiple clustering algorithms on the same data set. A more practical approach is described next.

In this section we focus on the solution obtained with the iterative clustering algorithm. Our best clustering model with this algorithm is the one produced at a cross-validation threshold of 0.69 (*see* **Section 2.4**). This clustering consists of 213 clusters with the number of genes per cluster ranging from 5 to 141. We were interested in those clusters that are most strongly correlated with the experimental data.

We used a p -value measure similar to the p -value we described in **Section 3.3**. One can compute the p -value of individual clusters and pick the most significant ones. Given a cluster i with n_i genes and e_i edges, of which t_i edges are experimentally verified edges, and $e_{i,cross}$ cross edges of which $t_{i,cross}$ are experimentally verified, we define the probability of the type

$$P(i) = \binom{T}{t_i t_{i,cross}} p_i^{t_i} p_{i,cross}^{t_{i,cross}} (1 - p_i - p_{i,cross})^{T - t_i - t_{i,cross}}$$

and its likelihood ratio. We then compute the probability that a type sampled according to the random type will have higher likelihood ratio.

In **Fig. 21.11** we plot the significance of individual clusters. We examined the annotations for genes in the clusters with the top p -values (**Table 21.2**). While one cannot assign a singular function to each cluster, we observe the themes of these clusters. For example, the second most significant cluster contains mostly dehydrogenases, while the third most significant cluster contains genes that are involved in DNA repair, mitosis, and meiosis. The fourth contains many ribosomal genes as well as translation initiation

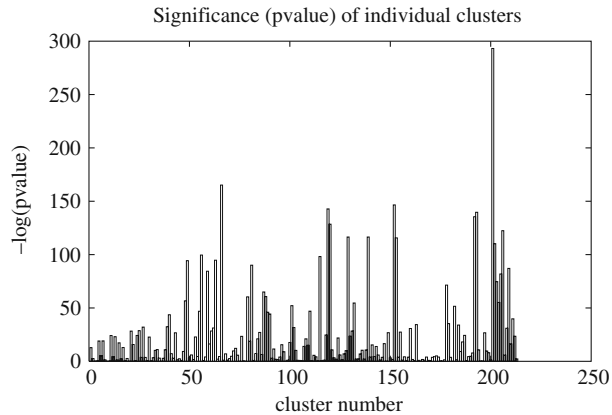


Fig. 21.11. Significance of individual clusters. Significance is measured in terms of the p -value (see text). Note that cluster number does not indicate cluster size. Using the Bonferroni correction we set the significance threshold at $< 1e^{-3}$. Of the 213 clusters 143 have significance value $< 1e^{-3}$.

factors and tRNA synthetases, and the fifth contains mitochondrial genes. The collective expression profiles for the five most significant clusters are shown in **Fig. 21.12**.

5.4. Association of Clusters with Cellular Pathways

Our results in the previous section indicate that almost all clusters are indeed strongly correlated with the experimental data and are the consequence of true functional links between genes. As we have shown in (32, 33), many of the pairwise similarities that are detected based on expression data are between proteins that participate in the same pathway.

It is interesting to analyze the mapping between pathways and clusters. Specifically, we were interested in finding if specific pathways can be mapped to a single or a few clusters. If a non-random mapping exists, then the corresponding clusters can help in refining the pathway structure as they might suggest activation and inhibition between the component proteins (for example, if the consensus profiles of clusters are correlated when a time shift is introduced).

To assess the significance of a mapping between a pathway and clusters, we analyze the distribution of the pathway proteins among the clusters and compare the distribution to the distribution one would obtain if the genes were to be placed into the clusters at random. This random process has a multinomial distribution where the probability p_i to fall into a cluster C_i is estimated by the number of all pathway proteins that are classified to C_i divided by the total number of pathway proteins. Given a pathway with n_i proteins in cluster i ($1 \leq i \leq k$), the probability of the mapping is

$$P(n_1, n_2, \dots, n_k) = \binom{N}{n_1 n_2 \dots n_k} p_1^{n_1} p_2^{n_2} \dots p_k^{n_k}$$

where $N = \sum_i n_i$ is the total number of proteins in the pathway.

Table 21.2
Most significant clusters. A summary of the significance, size, and type of genes in each cluster

Cluster number	<i>P</i> -value	Number of genes	Cluster summary
201	6.51e – 294	44	Large diverse cluster, cytochrome c oxidase subunits ATP synthase subunits, mitochondrial proteins
66	6.72e – 166	14	Dehydrogenases
152	2.42e – 147	88	DNA replication, DNA repair, mitosis, meiosis
119	1.50e – 143	141	Translation-related proteins, ribosomal proteins tRNA synthetases, translation initiation factors
193	2.20e – 140	113	Mitochondrial proteins
192	3.43e – 136	116	Large diverse cluster, transferases, hydrolases
120	3.13e – 129	15	Diverse cluster, lysases, transferases, ligases
206	4.04e – 123	19	Hydrolases
139	3.17e – 117	8	Mitosis
129	3.36e – 117	9	tRNA synthetases
153	1.99e – 116	50	Ribosomal proteins, helicases
202	6.01e – 111	5	Diverse cluster with oxidoreductases
56	2.40e – 100	31	Diverse cluster, hydrolases, oxidoreductases Mitochondrial proteins
115	6.94e – 99	21	RNA processing, growth regulation
63	1.68e – 95	8	Oxidoreductases, Fe-containing proteins

To determine the significance of a pathway distribution among the clusters, we can again compute the *p*-value as in **Section 3.3**. The five most significant mappings are listed in **Table 21.3**. A similar analysis of gene distribution can also be done for the homology clusters.

To make our discussion more concrete, we examined two significant pathways and the cluster profiles that correspond to the largest groupings of genes from these pathways. First, we considered the aminoacyl-tRNA biosynthesis pathway (*p*-value of $1.06e^{-42}$). Interestingly, the amino acid synthetases are clustered into four major clusters, each with a unique expression profile. In other words, different amino acids are synthesized at different

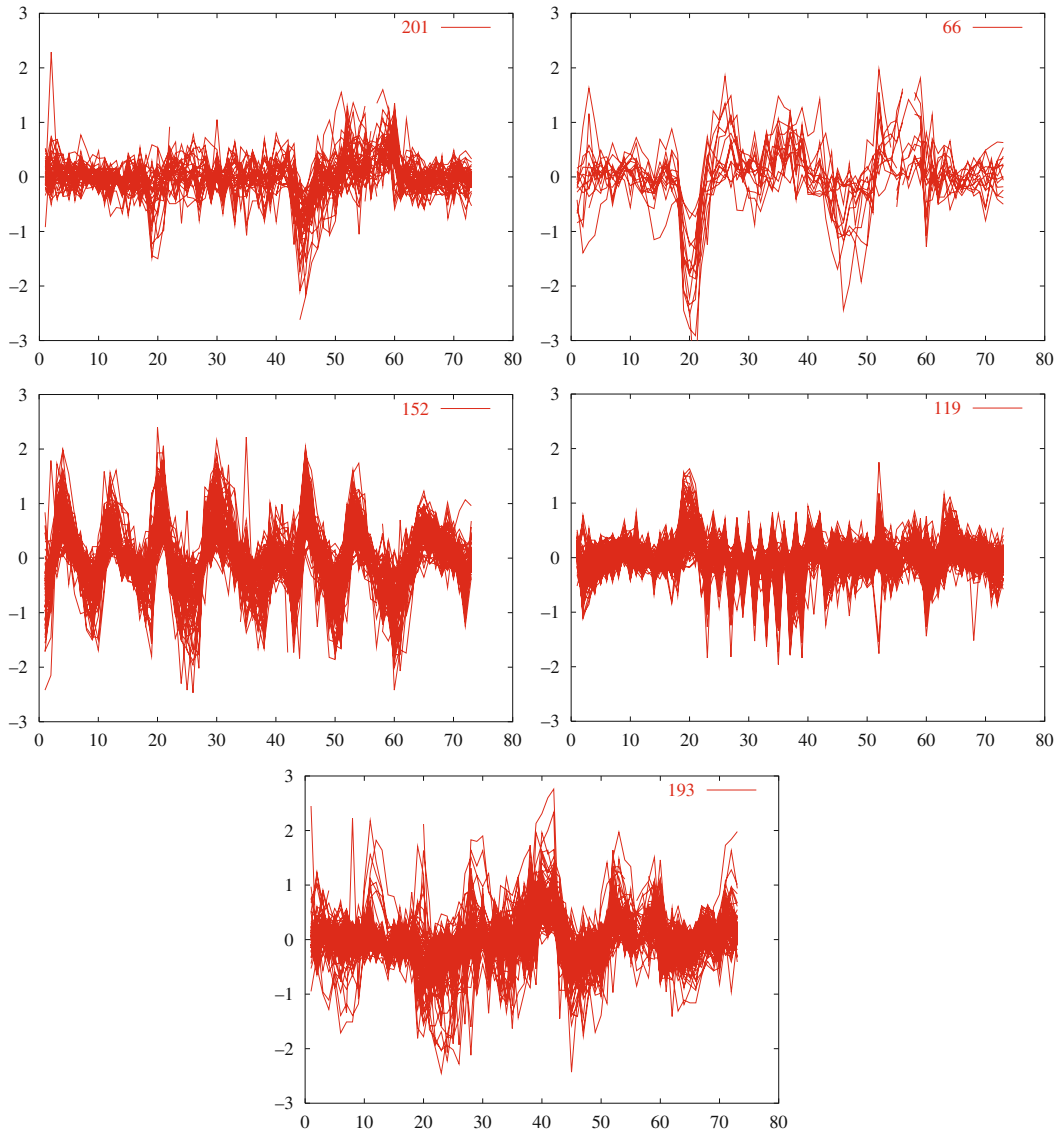


Fig. 21.12. Clusters of expression profiles. The collective expression profiles for the five most significant clusters are shown. Note the very distinctive expression profile of each cluster.

frequencies and in different parts (times) of the cell cycle. It is unclear why amino acid production is implemented this way, and this phenomena requires further study.

We also examine the oxidative phosphorylation pathway. This pathway is composed of five major complexes. While some genes from the clusters intermingle between the complexes, there is some separation of the different complexes into different clusters, suggesting again that their production is being coordinated. This analysis of the distribution of pathway genes among the different

Table 21.3
Pathways that are strongly correlated with clusters. The *P*-value indicates the significance of the mapping of pathway proteins into clusters

Pathway	<i>P</i> -value
Purine metabolism	6.2e – 75
Oxidative phosphorylation	1.8e – 64
Pyrimidine metabolism	3.3e – 58
Glycolysis/Gluconeogenesis	1.2e – 45
Starch and sucrose metabolism	1.2e – 45

clusters could be performed with more rigor for more complicated pathways. This may aid in the understanding of known biochemical pathways and the discovery of new ones.

6. Conclusions

Similarly expressed gene pairs can suggest functional relationships between genes. The degree or extent of the similarity under different conditions will determine the co-dependency of gene pairs. However, it is the coordinated nature of the cell regulation mechanism that differentiates expression analysis from protein sequence or structure analysis. The coordinated pattern of expression hints at more global processes, the nature of which cannot be determined based on individual gene pairs.

In search for groups of genes that are similarly expressed, many studies employed clustering algorithms to analyze expression data over the past decade. However, different algorithms produce different results, and it is hard to tell which solutions are more reliable. Moreover, microarray data are noisy, and it is hard to discern real signals from random fluctuations and coincidental regularities.

The goal of our study is to develop means for assessing the results of clustering algorithms that were applied to microarray expression data. We suggest two indices that when combined can help to pick the best model of many suggested models. The two indices assess two different aspects of the clustering results. The internal validation is based on the MDL principle while the external validation is using the experimentally verified data sets to assess the quality of the clustering results. The external measure directly

assesses the correlation of the clusters with the underlying structure of functional links, while the internal measure addresses issues such as model complexity and generalization. It is the combination of the two measures that leads to a selection of a better model overall. We test them on several clustering algorithms, including clustering algorithms that are especially effective in the presence of noise, as is the case for expression arrays. This objective assessment of the results can be used to assess other clustering algorithms or to compare different solutions that are generated by the same algorithm. Our benchmark can also be used to evaluate clustering algorithms using other expression data sets.

The conclusions are clearly data set dependent. Here we tested the algorithms on the yeast time-series data set. Spectral clustering produces interesting clusterings but it is very sensitive to the *Ncut* parameter value, and to obtain the right model one has to repeatedly apply this algorithm with different threshold values. *k*-means is fast and is more controllable (as far as the number of clusters), but is very susceptible to the initial starting point (this can be handled by applying a hierarchical version of *k*-means). Furthermore, it inherently uses the Euclidean metric, which has been shown to perform poorly on expression data compared to other metrics. Nevertheless, it produces very good results. Hierarchical clustering algorithms such as average-linkage and single-linkage are deterministic and as such always produce the same results when applied to the same data set. However, they can be sensitive to noise and outliers in the data set itself and not all variations perform equally well. For example, single-linkage generates many singletons. Since it is unclear a priori what is the typical shape of the clusters in the data set, it is recommended to test different variants (for spherical clusters average linkage will perform better, while for elongated clusters single-linkage will perform better). Moreover, to determine the right model one has to keep checkpoints of all the intermediate clusterings (not necessarily produced with standard implementations of pairwise clusterings).

Noise-tolerant algorithms, such as iterative clustering, provide an appealing alternative. We discuss two clustering algorithms that were especially optimized to deal with noisy data. The iterative clustering algorithm has a major advantage that makes it most appealing in our case; it is highly robust. By repeatedly estimating distance profiles, the geometry of the gene space is being re-evaluated and fluctuations are being averaged out. One way of explaining the success of this method is the fact that $\binom{n}{2}$ constraints (the set of distances with respect to the complete set of genes) are being considered when evaluating the distances between pairs of genes. Thus, even when the direct distance between genes *i* and *j* is “corrupted”, the distance profiles of these genes with respect to all other genes constrain the relative position of genes *i* and *j* in the gene space so that their true distance can be measured indirectly.

These distance profiles themselves can be noisy as well (assuming random noise) and yet induce reliable distances, even for small references sets as in demonstrated in (15).

Finally, we tested the premise that clusters are correlated with cellular pathways. Our results confirm that a significant correlation exists between co-expressed gene clusters and some cellular pathways.

Despite the relative success with clustering algorithms, one should keep in mind that conclusions based on expression data should be made with caution. Expression data are not only noisy because of technical reasons but also does not necessarily provide us with a true snapshot of the cell machinery, and there are many factors that affect protein abundance levels which are not reflected in mRNA expression data (35). Moreover, correlations between expression profiles might prevail only for a limited time along the cell cycle or for a subset of experimental conditions, thus suggesting that a local comparison mode might be more effective than a global mode in some cases. Indeed, several studies focused on local comparisons of subsets of expression profiles (32, 36). Nevertheless, there is substantial information in these arrays and when exploited carefully, one can derive meaningful conclusions with high confidence.

7. Notes



1. Absurdly, in the first few years to the introduction of microarray technology, papers on new clustering algorithms appeared in a pace that exceeded by far the number of expression data sets that were available for analysis.
2. The squared error function estimates the error for each sample point by measuring the squared Euclidean distance from the sample point to the centroid of the closest cluster (this can be perceived as the loss in information that is incurred by using the centroid to represent the sample point). The total error is the sum over all sample points.
3. Other normalizations can be applied but without any apparent significant impact (15).
4. The choice of the metric used to compare the rows can vary. Here, we employed the Euclidean metric.
5. This algorithm inherently assumes that the sample points reside in Euclidean space and non-trivial modifications (with possible impact on the convergence properties) are necessary to tune it to other spaces.

Acknowledgments

This work is supported by the National Science Foundation under Grant No. 0218521, as part of the NSF/NIH Collaborative Research in Computational Neuroscience Program.

References

1. Spellman, P.T., Sherlock, G., Zhang, M., Iyer, V., Eisen, M., Brown, P., Botstein, D. & Futcher, B. (1998). Comprehensive identification of cell cycle-regulated genes of the yeast *Saccharomyces cerevisiae* by Microarray Hybridization. *Mol. Bio. Cell.* 9, 3273–3297.
2. Hughes, T., Marton, M., Jones, A., Roberts, C., Stoughton, R., Armour, C., Bennett, H., Coffey, E., Dai, H., He, Y., Kidd, M., King, A., Meyer, M., Slade, D., Lum, P., Stepaniants, S., Shoemaker, D., Gachotte, D., Chakraburty, K., Simon, J., Bard, M. & Friend, S. (2000). Functional discovery via a compendium of expression profiles. *Cell.* 102, 109–126.
3. Liu, E.T. (2003). Classification of cancers by expression profiling. *Curr. Opin. Genet. Dev.* 13, 97–103.
4. McCormick, S.M., Frye S.R., Eskin, S.G., Teng, C.L., Lu, C.M., Russell, C.G., Chittur, K.K. & McIntire L.V. (2003). Microarray analysis of shear stressed endothelial cells. *Biorheology*, 40, 5–11.
5. Yeatman, T.J. (2003). The future of clinical cancer management: one tumor, one chip. *Am. Surg.* 69, 41–44.
6. Yoo, M.S., Chun, H.S., Son, J.J., DeGiorgio, L.A., Kim, D.J., Peng, C. & Son J.H. (2003). Brain research. *Mol. Brain Res.* 110, 76–84.
7. Jain, A.K. & Dubes, R.C. (1988). "Algorithms for clustering data". Prentice Hall, Englewood Cliffs, NJ.
8. Jain, A.K., Murthy, M.N. & Flynn, P.J. (1999). Data clustering: a review. *ACM Comput. Surv.* 31, 264–323.
9. Boutros, P.C. & Okey, A.B. (2005). Unsupervised pattern recognition: an introduction to the whys and wherefores of clustering microarray data. *Brief Bioinform.* 6, 331–343.
10. D'haeseleer, P. (2005). How does gene expression clustering work? *Nat. Biotechnol.* 23, 1499–1501.
11. Gray, R. M., Kieffer, J. C. & Linde, Y. (1980). Locally optimal block quantizer design. *Inf. Control* 45, 178–198.
12. Rose, K., Gurewitz, E. & Fox, G. (1990). A deterministic annealing approach to clustering. *Patt. Rec. Lett.* 11, 589–594.
13. Wu, Z. & Leahy, R. (1993). An optimal graph theoretic approach to data clustering: theory and its application to image segmentation. *PAMI.* 15, 1101–1113.
14. Shi, J. & Malik, J. (1997). Normalized cuts and image segmentation. *Proc. CVPR.* 731–737.
15. Dubnov, S., El-Yaniv, R., Gdalyahu, Y., Schneidman, E., Tishby, N. & Yona, G. (2002). A new non-parametric pairwise clustering algorithm based on iterative estimation of distance profiles. *Mach. Learn.*, 47, 35–61.
16. Rissanen, J. (1989). *Stochastic Complexity in Statistical Inquiry*. World Scientific, Singapore.
17. Bolshakova, N., Azuaje, F. & Cunningham, P. (2005). A knowledge-driven approach to cluster validity assessment. *Bioinformatics.* 21, 2546–2547.
18. Ashburner, M., Ball, C.A., Blake, J.A., Botstein, D., Butler, H., Cherry, J.M., Davis, A.P., Dolinski, K., Dwight, S.S., Eppig, J.T., Harris, M.A., Hill, D.P., Issel-Tarver, L., Kasarskis, A., Lewis, S., Matese, J.C., Richardson, J.E., Ringwald, M., Rubin, G.M., Sherlock, G. (2000). Gene ontology: tool for the unification of biology. *Gene Ontol. Consortium. Nat Genet.* 25, 25–29.
19. Speer, N., Spieth, C. & Zell, A. (2004). A memetic clustering algorithm for the functional partition of genes based on the gene ontology. In *Proceedings of the 2004 IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB 2004)*, San Diego, USA IEEE Press, 252–259.
20. Raychaudhuri, S., Schutze, H. & Altman, R.B. (2002). Using text analysis to identify functionally coherent gene groups. *Genome Res.* 12, 1582–1590.

21. Gat-Viks, I., Sharan, R. & Shamir, R. (2003). Scoring clustering solutions by their biological relevance. *Bioinformatics* 19 2381–2389.
22. Bolshakova, N. & Azuaje, F. (2003). Machaon CVE: cluster validation for gene expression data. *Bioinformatics* 19, 2494–2495.
23. Bertoni, A. & Valentini, G. (2006). Randomized maps for assessing the reliability of patients clusters in DNA microarray data analyses. *Artif. Intell. Med.* 37 85–109.
24. Olman, V., Xu, D. & Xu, Y. (2003). CUBIC: identification of regulatory binding sites through data clustering. *J. Bioinform. Comput. Biol.* 1, 21–40.
25. McShane, L.M., Radmacher, M.D., Freidlin, B., Yu, R., Li, M.C. & Simon, R. (2002). Methods for assessing reproducibility of clustering patterns observed in analyses of microarray data. *Bioinformatics*. 18, 1462–1469.
26. Yeung, K.Y., Haynor, D.R. & Ruzzo, W.L. (2001). Validating clustering for gene expression data. *Bioinformatics*. 17, 309–318.
27. Smolkin, M. & Ghosh, D. (2003). Cluster stability scores for microarray data in cancer studies. *BMC Bioinformatics*. 4, 36.
28. Dudoit, S. & Fridlyand, J. (2003). Bagging to improve the accuracy of a clustering procedure. *Bioinformatics*. 19 1090–1099.
29. Zhang, K. & Zhao, H. (2000). Assessing reliability of gene clusters from gene expression data. *Funct. Integr. Genomics*. 1, 156–173.
30. Schwarz, G. (1978). Estimating the dimension of a model. *Ann. Stat.* 6, 461–464.
31. Bejerano, G. (2003). Efficient exact p-value computation and applications to biosequence analysis. In the proceedings of RECOMB 2003, 38–47, ACM press, New York.
32. Yona, G., Dirks, W., Rahman, R. & Lin, M. (2006). Effective similarity measures for expression profiles. *Bioinformatics*. 22, 1616–1622.
33. Dirks, W. & Yona, G. (2003). A comprehensive study of the notion of functional link between genes based on microarray data, promoter signals, protein-protein interactions and pathway analysis. Technical report TR2004-1921, Computing and Information Science, Cornell University.
34. Kanehisa, M. (1996). Toward pathway engineering: a new database of genetic and molecular pathways. *Sci. Technol. Jpn.* 59, 34–38.
35. Gygi, S.P., Rochon, Y., Franza, B.R. & Aebersold, R. (1999). Correlation between protein and mRNA abundance in yeast. *Mol. Cell Biol.* 19, 1720–1730.
36. Qian, J., Dolled-Filhart, M., Lin, J., Yu, H. & Gerstein, M. (2001). Beyond synexpression relationships: local clustering of time-shifted and inverted gene expression profiles identifies new, biologically relevant interactions. *J. Mol. Biol.* 312, 1053–1066.

Chapter 22

The Bioverse API and Web Application

Michal Guerquin, Jason McDermott, Zach Frazier, and Ram Samudrala

Abstract

The Bioverse is a framework for creating, warehousing and presenting biological information based on hierarchical levels of organisation. The framework is guided by a deeper philosophy of desiring to represent all relationships between all components of biological systems towards the goal of a wholistic picture of organismal biology. Data from various sources are combined into a single repository and a uniform interface is exposed to access it. The power of the approach of the Bioverse is that, due to its inclusive nature, patterns emerge from the acquired data and new predictions are made. The implementation of this repository (beginning with acquisition of source data, processing in a pipeline, and concluding with storage in a relational database) and interfaces to the data contained in it, from a programmatic application interface to a user friendly web application, are discussed.

Key words: Bioverse, framework, systems biology, proteomics, interaction, protein structure, functional annotation, prediction, visualization, server, programming interface, data warehouse.

1. Introduction

The Bioverse project evolved over many years, with the initial idea of a wholistic systems approach to catalogue, predict and present biological information remaining at the heart of the effort. The biological information in the Bioverse is presently specific to the field of ‘proteomics’; these include the protein amino acid sequences, known and predicted structures, known and predicted functions and relationships to other proteins such as functional associations in the case of complexes or metabolic pathways and homology.

The Bioverse implementation consists of a pipeline in which this information is transformed and relationships are established, a database where it is organised in an efficient manner (*see Chapter 23*),

an Application Programming Interface (API) that allows specific queries to be issued against the database and a web application that utilises the API to present the data in a browser to Internet users.

In this chapter, we discuss the highly modular Bioverse framework at a conceptual level, detail the implementation of its API and web application components and provide example uses of the framework. We first discuss the data organisation and sources. We then describe the usage of the Bioverse Web Application that is a primary front end to these data. We conclude with an overview of the API and examples of implementing useful programs that use the Bioverse framework.

2. Audience

The Bioverse framework is designed with three audiences in mind: our collaborators who require organism specific information to solve the biological research problems they are working on; bioinformatics experts who are performing large-scale systems analyses of our data; and end users who are seeking detailed information about a particular protein or a small set of proteins. Our collaborators work closely with us and include the Pacific Northwest National Laboratory in Richland, Washington; the National Center for Genetic Engineering and Biotechnology, Thailand (BIOTEC); and the Beijing Genomics Institute who are using our framework for whole genome annotation and comparison (1, 2). Communication with these collaborators occurs through the web site and also by exchange of raw data, so they are the ones likely to obtain the most appropriate results for solving a specific biological problem. The bioinformatics users expect our data to be accessible in a consistent manner and exportable into a format easily transformed into their existing system(s), to the extent that the Bioverse API is developed, they have access to all our data. The end users expect input and output to be simple, easily comprehensible, and to offer rapid insights for specific genes or proteins of interest. Satisfying end users' expectations is a daunting task, since there are many communities with different, ambiguous, and sometimes conflicting desires. In the following sections, we detail how our framework is designed to be valuable to all these user communities.

3. Bioverse Data

3.1. Organisation

Representation of biological systems requires striking a balance between the level of detail and abstraction to solve an intended biological problem. An overly abstract representation will hide the

essential differences between systems, while an overly detailed representation will narrow the scope of the answer to a biological question. Ontologies offer one way of representing relationships between detailed components as concepts in the system. They are therefore one resource to use when presenting related data.

The Bioverse is our first step towards representing the details and organisation of entire biological systems *in silico*. The Bioverse presently operates on the level of proteins. We organise and describe them in the following ways:

- Individual molecules. These are proteins detected in the sequences of many genomes and characterised in public databases.
- Molecule attributes. Each molecule is uniquely identified by its amino acid sequence. A molecule is assigned one or more names, one or more functional annotations and one or more structure definitions. The latter two may be experimentally determined or predicted *in silico*. These attributes are in turn described by various meta-data and can be related through organisation methods like the gene ontology (GO) (3).
- Molecule relationships. Relationships between molecules include explicit physical protein–protein interactions, implicit relations such as those present in regulatory complexes and evolutionary relationships based on sequence similarity. Such relations are rich with information when studied in terms of graph theoretic algorithms.
- Collections of molecules. The molecules, or proteins, are associated with an organism they are expressed in.

While the molecules are grouped into sets or collections in organisms, and organisms in turn can be grouped into taxonomic hierarchies, a ‘systems biology’ perspective encourages thinking that liberally relates many components to one another. This organisation arises from the structure of hierarchies and results in a network of connected components. The components in the Bioverse are currently proteins and some nucleotide sequences but the representational framework is extensible to include DNA, RNA, and biologically important small molecules and ions, and their relationships with each other. The benefits of an inclusive data warehouse become more pronounced as the amount of interrelated data grows. For example, structural and functional genomics projects rely upon the statistical significance of structural and functional feature co-occurrence in large data sets.

3.2. Sources

Studies of individual organisms, or systems in organisms, are being conducted in parallel all around the world. Costly experiments conclude with information about the observed functions or structures of individual proteins or small sets of related proteins. These dispersed pieces of data are meticulously collected into

organism-specific databases such as *Saccharomyces* Genome Database (SGD) (4), WormBase (5), FlyBase (6) and Human Protein Reference Database (HPRD) (7). Specific systems-level information is generated in a similar fashion, some focusing on global protein properties like functional annotations, others on catalytic or indirect interactions, and yet others on physical interactions. Here we describe the origins of protein properties, like their names, functional annotations, structures, and interactions as they are catalogued by the Bioverse.

By convention, protein molecules are assigned names. Biologists seeking information about a named molecule can find it in the Bioverse if the protein database where that name occurs has been integrated. The NCBI Gene Identifiers are an example of such names that become searchable name attributes in the Bioverse. There is, of course, the inherent challenge of having multiple naming systems and conventions. Settling on one system simplifies the design but limits the usefulness to users unfamiliar with that system, while adopting many naming systems introduces an excess of data that appears as noise to an uninitiated observer.

Functional annotations are human-readable labels assigned to characterise the behaviour of a protein. Such labels can refer to classes of proteins that are being studied. The landscape of possible functions can be organised into a hierarchy, as demonstrated by GO (3). We use GO for protein functional classification both from predictive methods (such as InterPro (8)) and from manually assigned functional annotations from source databases (such as SGD).

Molecule structures, if known, are obtained from RCSB Protein Data Bank (PDB) (9). Classifications of these structures are published by the Structural Classification of Proteins (SCOP) (10–13) and Superfamily (14, 15) projects, both of which are inherited by the Bioverse. Alternatively, protein structures can be predicted. A prediction of the secondary structure (a positional description indicating sheets, helices and coils) can be obtained by using a program like PSIPRED (16). The three-dimensional tertiary structure can be predicted through various comparative and de novo methods (17–20).

Experimentally observed protein interactions are catalogued in databases like Biomolecular Interaction Network Database (BIND) (21), Human Protein Reference Database (HPRD) (7), Munich Information Center for Protein Sequences (MIPS) (22) and Database of Interacting Proteins (DIP) (23), which we use extensively. Sequence similarity computations are performed with BLAST against all molecules in the Bioverse.

These known and computable attributes of molecules form the basis of our predictions. The Interolog method (24) is an example of combining sequence similarity information with BIND data to predict novel interactions. Similar methods are applied to function prediction. The specifics of these methods determine our certainty of these predictions and a confidence value is derived accordingly (25).

3.3. Creation and Presentation

The initial data in the Bioverse originates from various sources. After being gathered, it is normalised and collated using a ‘pipeline’. The results are stored in a relational database, and a programming interface allows queries to be issued. The Bioverse Web Application utilises this interface and provides a user-friendly interface to the data (Fig. 22.1).

3.4. Pipeline

The pipeline distributes data processing across many nodes in a computer cluster (26). These data are organised into a uniform format in preparation for loading into a centralised database, and the execution of algorithms to summarise the data and build

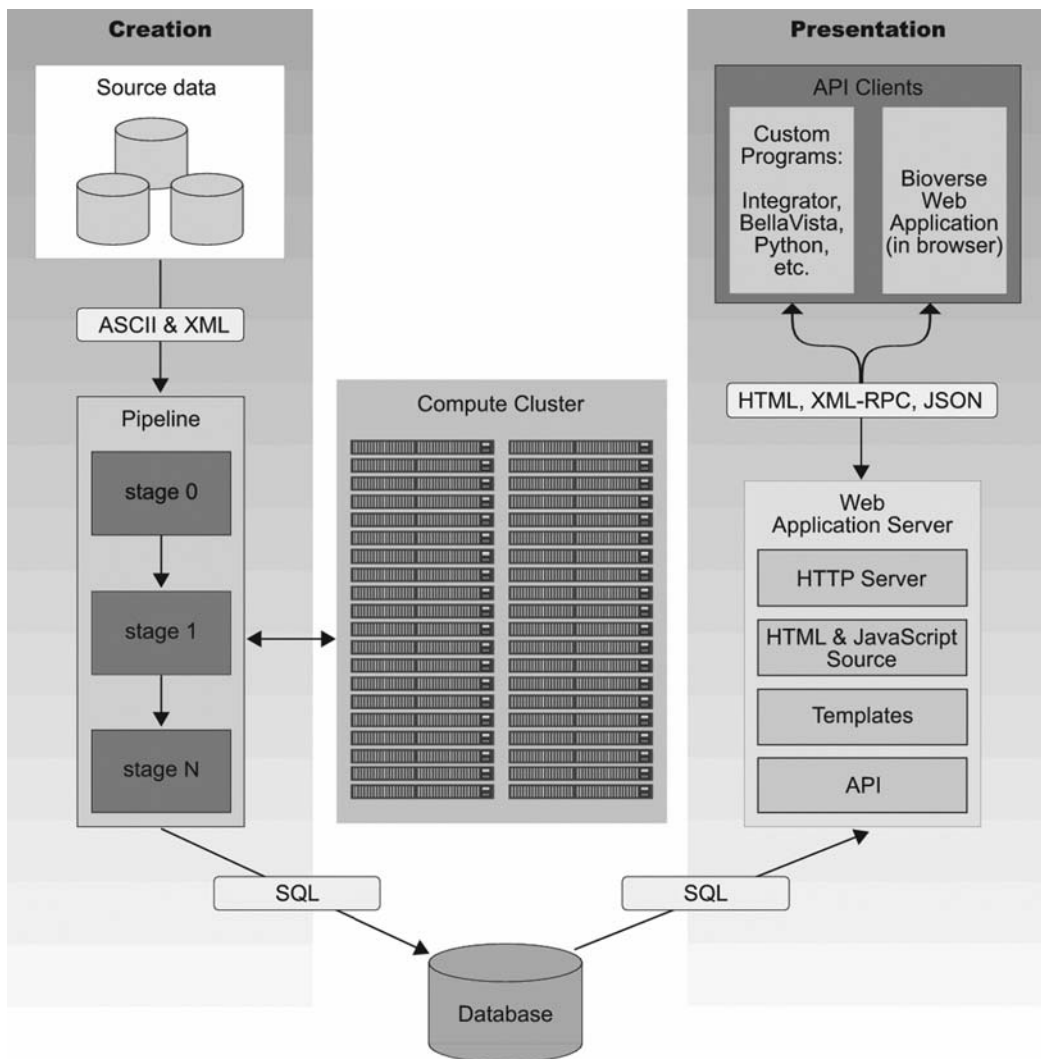


Fig. 22.1. The Bioverse infrastructure described in terms of components and interfaces. External sources are processed in a pipeline and results stored in a database. This database is accessible to the web application server that makes its contents available to the world.

relationships. For example, an all-versus-all profile based PSI-BLAST (27) search is executed across all molecule sequences; PSIPRED (16) is used for secondary structure prediction and HMMER (28) is used to assign proteins to functional families. Novel functional annotations are predicted based on known annotations and the topology of molecule similarity and interaction networks. The pipeline concludes by depositing data into a centralised database (Table 22.1).

Table 22.1

The Bioverse holds individual and relationship information for many molecules or proteins. For 54 organisms (486,520 molecules) we require five TB of distributed storage space, two months of dedicated work by 160 CPUs to process and one week to write to a centralised relational database and create indexes on relevant tables. The names, sequences and functional annotations of all molecules are searchable in the Bioverse Web Application and the interaction networks are browsable with Integrator(42)

Organism	Molecules	Interaction edges
<i>Agrobacterium tumefaciens</i> (A348 hypothetical)	5,368	5,190
<i>Agrobacterium tumefaciens</i> (C58 Cereon)	5,290	5,067
<i>Agrobacterium tumefaciens</i> (C58 UW)	5,396	4,934
<i>Arabidopsis thaliana</i>	27,833	88,211
<i>Bacillus anthracis</i> (Ames)	5,309	1,965
<i>Bacillus subtilis</i>	4,105	2,556
<i>Bordetella pertussis</i>	3,248	2,690
<i>Brucella melitensis</i>	3,188	2,161
<i>Brucella suis</i>	3,256	2,195
<i>Caenorhabditis elegans</i>	20,936	973,608
<i>Campylobacter jejuni</i>	1,634	1,506
<i>Canis familiaris</i>	16,817	900,459
<i>Chlamydia trachomatis</i>	894	357
<i>Clostridium perfringens</i>	2,722	1,191
<i>Drosophila melanogaster</i>	16,475	5,582,634
<i>Encephalitozoon cuniculi</i>	1,908	6,079
<i>Escherichia coli</i>	4,208	13,196

(continued)

Table 22.1 (continued)

Organism	Molecules	Interaction edges
<i>Halobacterium sp</i> (NRC-1)	2,425	682
<i>Helicobacter pylori</i> (26695)	1,562	9,523
<i>Homo sapiens</i>	26,741	9,362,672
<i>Listeria monocytogenes</i>	2,844	1,588
<i>Magnaporthe grisea</i>	11,042	83,055
<i>Methanococcus jannaschii</i>	1,785	500
<i>Methanococcus maripaludis</i> (C58 UW)	1,722	453
<i>Mus musculus</i>	26,181	10,427,816
<i>Mus musculus</i> (BGI)	12,412	16,135,715
<i>Mycobacterium bovis</i>	3,911	1,812
<i>Mycobacterium tuberculosis</i> (CDC1551)	4,178	1,767
<i>Neisseria meningitidis</i> (mc58)	2,020	1,114
<i>Oryza sativa</i> (indica BGI 9311)	57,135	13,867,984
<i>Oryza sativa</i> (japonica KOME cDNAs)	25,875	699,232
<i>Oryza sativa</i> (japonica Syngenta)	60,017	1,277,025
<i>Pan troglodytes</i>	21,685	3,478,283
<i>Plasmodium falciparum</i>	5,252	29,776
<i>Pseudomonas aeruginosa</i>	5,555	4,977
<i>Pyrococcus abyssi</i>	1,896	64
<i>Pyrococcus furiosus</i>	2,053	834
<i>Rattus norvegicus</i>	22,642	11,810,162
<i>Rhodopseudomonas palustris</i>	4,806	4,556
<i>Rickettsia conorii</i>	1,374	886
<i>Rickettsia prowazekii</i>	834	801
<i>Saccharomyces cerevisiae</i>	5,801	467,989
<i>Salmonella typhimurium</i>	4,532	4,795
<i>Shewanella oneidensis</i> (2a)	4,300	2,908
<i>Shigella flexneri</i> (2a)	4,080	3,983
<i>Staphylococcus aureus</i> (mw2)	2,632	1,385
<i>Thermotoga maritima</i>	1,845	1,031

(continued)

Table 22.1 (continued)

Organism	Molecules	Interaction edges
<i>Vibrio cholerae</i>	3,788	3,249
<i>Vibrio parahaemolyticus</i>	4,821	3,673
<i>Vibrio vulnificus</i> (CMCP6)	4,484	3,925
<i>Yersinia pestis</i>	3,898	4,216
<i>Yersinia pestis</i> (BGI 91001)	4,143	4,448
<i>Yersinia pestis</i> (BGI CO92)	3,708	4,026
<i>Yersinia pestis</i> (BGI KIM)	3,954	4,082
Total	486,520	75,304,986

3.5. Protinfo

The Protinfo suite of servers (29), available at <http://protinfo.comp-bio.washington.edu>, comprises several computational techniques for protein structure and function prediction developed by the Samudrala group (29–41). The results of these techniques, as they are applied to all the proteins in the Bioverse, are integrated into the pipeline.

4. Database

The Bioverse stores information in a relational database. Relationships are stored in a space-efficient way to avoid redundancy. In practice, however, we have found the database in a “Write Once, Read Many” pattern of access, so organisational optimisations for purposes of query efficiency are implemented. The nature of this is in the form of a data warehouse. This is discussed in greater detail in **Chapter 23**.

5. Web Application

A traditional web page represents some specialised information with links to other related pages. The intent of a web application, however, is different. Rather than offering small bits of data that are loosely hyperlinked, we cohesively present a large amount of semantically relevant biological information.

The Bioverse Web Application, available at <http://bioverse.compbio.washington.edu>, is implemented as a single web page. This HTML document is manipulated through JavaScript code executed by the web browser. Interface events, such as button clicks or form submissions, trigger JavaScript functions. These functions may issue calls to the Bioverse API (Section 6) to retrieve relevant data and visualise it on the page.

For example, we consider the single molecule view of the Bioverse Web Application (Fig. 22.2). The intent of this view is to emphasise the relationship between all information known about a single protein, presented relative to the amino acid sequence of the molecule. For each molecule, this wealth of information is initially presented in a compact form and organised into sections. To expand these sections and see more detailed

The screenshot shows the Bioverse web application interface in a Firefox browser window. The URL is <http://bioverse.compbio.washington.edu>. The page title is "Bioverse" and the search criteria are "H. sapiens (26,741 molecules)". The main content area displays "H. sapiens molecule 123" with a direct link to the molecule. The sequence is shown as "RPEPSKSLAPAPKGGSKKATKAQNDGKRRSRKESYSLVYVKVLRKVPDGTGSSKARGLRHSFVHDFERLAGEASRLAHYHWRSTLTSREIQTAVRLLLPGLAKHAYSEGTRAVTKYTSK". Below the sequence, there is a bar chart showing the amino acid composition, with a total of 126 amino acids. The chart shows the relative frequency of each amino acid, with a 16% conservation level indicated. The amino acid composition is listed as "A I L H V F Y W M S T Q N C H D E K R G P". The structure section shows "Structure - secondary" as "histone-fold" and "Structure - tertiary" as "histone-fold". The function section lists 19 functional annotations, including "histone h2b", "transcription factor cbf/nf-y/archaeal histone", "histone core", "amidation site", "n-myristoylation site", "casein kinase ii phosphorylation site", "bipartite nuclear targeting sequence", "protein kinase c phosphorylation site", "camp/cgmp-dependent protein kinase phosphorylation site", "histone-fold", "transcription initiation factor tfiid", "histone h2a", "nucleosome", "dna binding", "nucleus", "nucleosome assembly", "chromosome organization and biogenesis (sensu eukarya)", "transcription factor tfiid complex", and "transcription initiation".

Fig. 22.2. Single molecule view in the Bioverse. The molecule's sequence, structure and function information is shown in three sections. Statistics about sequence composition, as well as amino acid conservation, are expanded. Other sections may be expanded to reveal molecule relationships and evidence supporting predicted functional annotations.

information, buttons can be clicked on the page. Such clicks initiate requests to the server for that information and the page is modified in place by populating the relevant sections only. This reduces the overhead of retrieving increasing quantities of data as more sections are expanded, which occurs in a more traditional web page model.

The traditional meaning of a web page is no longer useful when describing the visible content of a web application. The web browser becomes a platform for executing application logic and rendering content onto an initially empty canvas. Shaping the contents of the page in a piecemeal fashion is a dynamic process and results in tight integration between all the data that can be displayed, and the operations that can be performed on it.

The initial page is empty and contains an onload function call attached to the HTML body element. This function's job is to recover information about which tabs were open from the last session (if any) and to re-open them.

The Bioverse Web Application composes its interface by rendering small sections of content from a pool of predefined HTML templates (Fig. 22.3). These are application elements such as the list of organisms, the search form, the search results list, molecule sequence information, and others. The content of these templates is highly specific to their use and templates will often contain placeholders for other templates.



The screenshot shows the Bioverse web application. At the top, there is a search bar with the text "Find" followed by an input field containing "S. cerevisiae (5801 molecules)" and a "Start new query" button. Below the search bar, a link is visible: "(a) Read about Saccharomyces cerevisiae...". This link is highlighted with a box. Below the link, a new tab is open, labeled "(c) S. cerevisiae x1". The content of this tab is a detailed view of Saccharomyces cerevisiae, including a heading "Saccharomyces cerevisiae", a paragraph stating "Saccharomyces cerevisiae was last updated November 6, 2005.", and another paragraph stating "There are 5801 molecules and 467,989 contextual relationships in this organism.".

(a) `
Read about ${details.long_name}...
`

(b) `function show_organism(organism_id)
{
...
}`

(c) `<div class="sectionheading">${details.long_name}</div>
<p>${details.long_name} was last updated ${details.updated}</p>
<p>There are ${details.molecule_count.commify()} molecules and
${details.interaction_count.commify()} contextual relationships
in this organism.</p>`

Fig. 22.3. A link on the page (a) has an onclick property. When clicked, the browser calls the show_organism function (b). This creates a new tab on the page (c) with content derived from a template (d). Note the use of the template syntax to build the link in (a).

In this paradigm of a dynamic web application, the meaning of a “web page” is different. It is no longer possible to accurately represent the state of the web application with a single address. This is resolved by storing a limited subset of state information on the server and recovering it when the web application is loaded. This allows for a browser to follow the traditional model of navigating away from or towards the Bioverse Web Application while relying on the interface initialisation methods in JavaScript to rebuild the application state most recently created by the user.

There are some shortcuts for affecting the state by requesting certain URLs, specifically the referencing of organisms and molecules. For example, the request for the path /about will append the ‘About’ tab to the set of available tabs in the web application. Other such shortcut URLs are /about/credits, /help, and /preferences. Similarly, visiting /oryza-sativa/123 will add molecule number 123 of *Oryza sativa* (rice) to the set of open tabs.

5.1. Using the Web Application

The hierarchical organisation of most of the data in the Bioverse lends itself well to a ‘drill down’ presentation where one chooses an organism, then a molecule in the organism and finally the attributes of the molecule to inspect. However, the number of proteins in each organism makes this impractical for the end user. It is therefore necessary to introduce a filtering mechanism, like a search, to focus on a smaller subset of proteins. For example, from a systems perspective, it makes sense to select molecules based on a protein relationship criteria. Searching the data with a free-form user query was inspired by the success of web search engines. The idea is to present the user with a means of writing an expressive description of molecules of interest in a defined syntax and focusing on only those molecules.

The search query is a whitespace separated list of tokens. The tokens may be terms (like binding site, CCR5, GO:000451, MQMRSRMVRLLMML) that can match the name, functional annotation or sequence of a molecule. To restrict which meta-data field of a molecule is searched, the term may be preceded with a meta-data qualifier like name:, function: or sequence:. If a term is not preceded with such a qualifier, then its qualifier is inferred. For example, ABCC-CEFH cannot match an amino acid sequence because it contains the letter B, which is not part of the amino acid alphabet. Similarly, dna binding cannot match a molecule name because names cannot contain spaces.

In addition, a token may be preceded with a sign (plus, tilde, or minus character) to indicate the rule (must, may or must not, respectively) for matching molecules (**Fig. 22.4**). A term without a sign indicates that it must occur in the annotation of a molecule. Some example queries are: dna ~binding, +kinase -atp, ‘binding

Plus, minus and tilde symbols may precede a term:

+term	means that the term must occur
~term	means that the term may occur
-term	means that the term must not occur

Fig. 22.4. Symbols precede a term to indicate the term's role in the molecule matching algorithm.

site' -function:iron ~rna. This query syntax is sufficiently expressive to allow for quite sophisticated searches to take place. The web interface provides an explanation (Fig. 22.5).

Molecules matching these criteria are listed in order of relevance. Relevance is computed by inspecting the confidence value of matching functional annotations and listing molecules with high matching confidence before those with a low matching confidence. Name or sequence matches do not contribute directly to this relevance measure, except to affect the ordering such that molecules with matching functional annotations appear after those matching the name or sequence terms (Fig. 22.6).

Explanation of query

The query "binding site" .function:iron ~rna means that all of these conditions must be satisfied:

- ☉ "binding site" **must** occur in the **functional annotations** of the matching molecule
- ☉ "iron" **must not** occur in the **functional annotations** of the matching molecule
- ☉ "rna" **may** occur in the **names** or **functional annotations** of the matching molecule

Matching functional annotations must have a confidence of at least 0.3.

It matched **518** molecules in **11.86** seconds in **Homo sapiens**.

Refine the query

Find in with confidence ≥

Show query rules...

Fig. 22.5. Explanation of a search query. Note that due to ambiguity, the term "rna" may be either a name or a functional annotation.

Showing molecules 1-20 of 23 for **actin ~name:CAA81841 ~sequence:MDSEVAALVIDNGSGMCKAGFAGDDAPRAVFPVIVGRP** in **S. cerevisiae** with **confidence ≥ 0.3** - explain and refine this query .

Next >

- ✖ Saccharomyces cerevisiae 3355: matches **1 name** (CAA81841.1) and **4 functions** actin binding, F-actin capping protein complex, actin cytoskeleton organization and biogenesis, F-actin capping protein, alpha subunit .
- ✖ Saccharomyces cerevisiae 1700: matches **the sequence** and **2 functions** actin cytoskeleton, Actin/actin-like .
- ✖ Saccharomyces cerevisiae 3349: matches **1 function** actin filament polymerization .
- ✖ Saccharomyces cerevisiae 2727: matches **4 functions** actin binding, F-actin capping protein complex, actin cytoskeleton organization and biogenesis, F-actin capping protein, beta subunit .
- ✖ Saccharomyces cerevisiae 3857: matches **1 function** regulation of actin filament polymerization .

Fig. 22.6. Molecules matching a search query.

Other means of presenting results are being implemented, which involve the algorithmic classification of matching molecules into groups or categories by inspecting shared properties and relationships. This would narrow the focus interactively to a more refined set of molecules, which can be especially useful for exploring result sets that encompass hundreds or thousands of molecules. Superimposing such result sets onto protein interaction networks and visualising these with tools such as Integrator (42) can aid in comprehending the structure of the data.

6. Application Programming Interface (API)

The Application Programming Interface (API) is the glue between the data in the database and an application designed to use it. The programming interface is in the form of an Internet service. A programmer with an Internet connection can issue queries using the API and retrieve Bioverse data. In this section we discuss the general idea of the API and devote the next section to describing the API usage.

6.1. The Role of the Bioverse API

Data-intensive web services traditionally create a database and set up a web site to present its contents. To prune the interesting data, a query form exists for a person to fill out, submit to the server and have the results presented. This is ideal for an individual user with questions that can be asked in a predictable way, or for a quick overview of the data in the database.

From the perspective of developing a web site, such a form is actually interfaced to a library of internal server routines that are customised to retrieve the data from the database. These routines are invisible but accessible indirectly through the web form mentioned earlier. We have exposed our library of these internal routines to the world in the form of the Bioverse API.

Applications have already been written to utilise the data in our repository (*see Section 6.5*). To emphasise the importance we place upon the API, the most prominent application to utilise it is the Bioverse Web Application itself.

6.2. The Role of the Client

An application that utilises the Bioverse API is considered to be the ‘client’ of the Bioverse server. The API provides data in a raw but structured format. It is not immediately meaningful for a user, so the application programmer must transform this raw data into a useful form. For example, the amino acid sequence of a protein should have its positions enumerated for visualisation. This is implemented in the Bioverse Web Application by drawing a ruler



Fig. 22.7. (a and b) Sequence position ruler and confidence bars generated by the client.

above the sequence with positions numbered periodically. Similarly, the confidence of functional annotations (25) is a real number between 0 and 1 and can be visualised with coloured images (Figs. 22.7a and b).

A notable use of the API by a client application is for the visualisation of relationship networks. The Bioverse API presents fragments of such networks in the form of parent–child relationship lists. It is the role of the client application, such as Integrator (42), to visualise such a list in the form of a graph and accumulate network fragments to ‘grow’ the graph during exploration (Figs. 22.8 and 22.9).

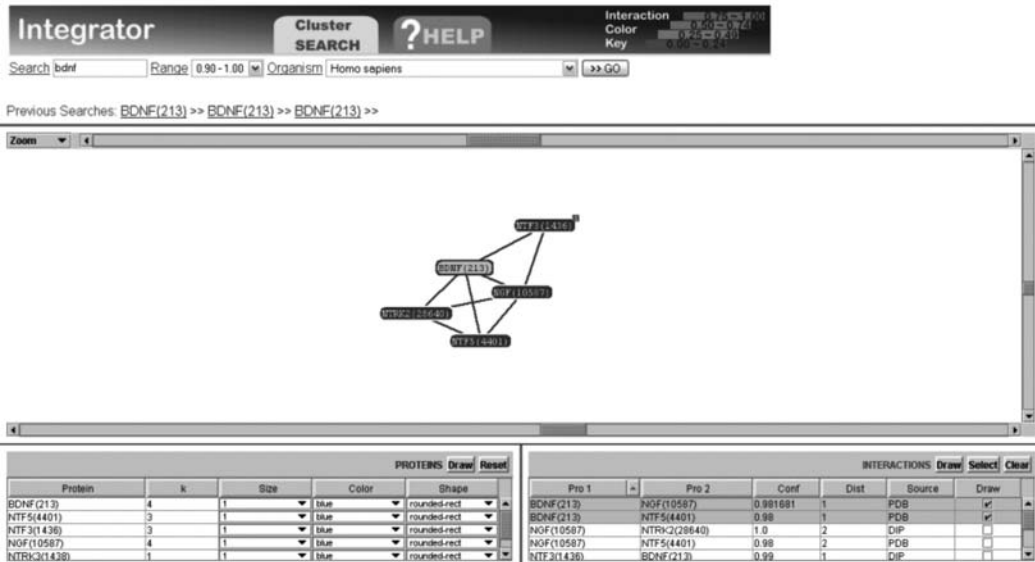


Fig. 22.8. The Bioverse Integrator (42) showing a small network of related proteins. The lower interface elements allow the inclusion or exclusion of nodes from the network based on criteria and the adjustment of visual properties like node colours and sizes.

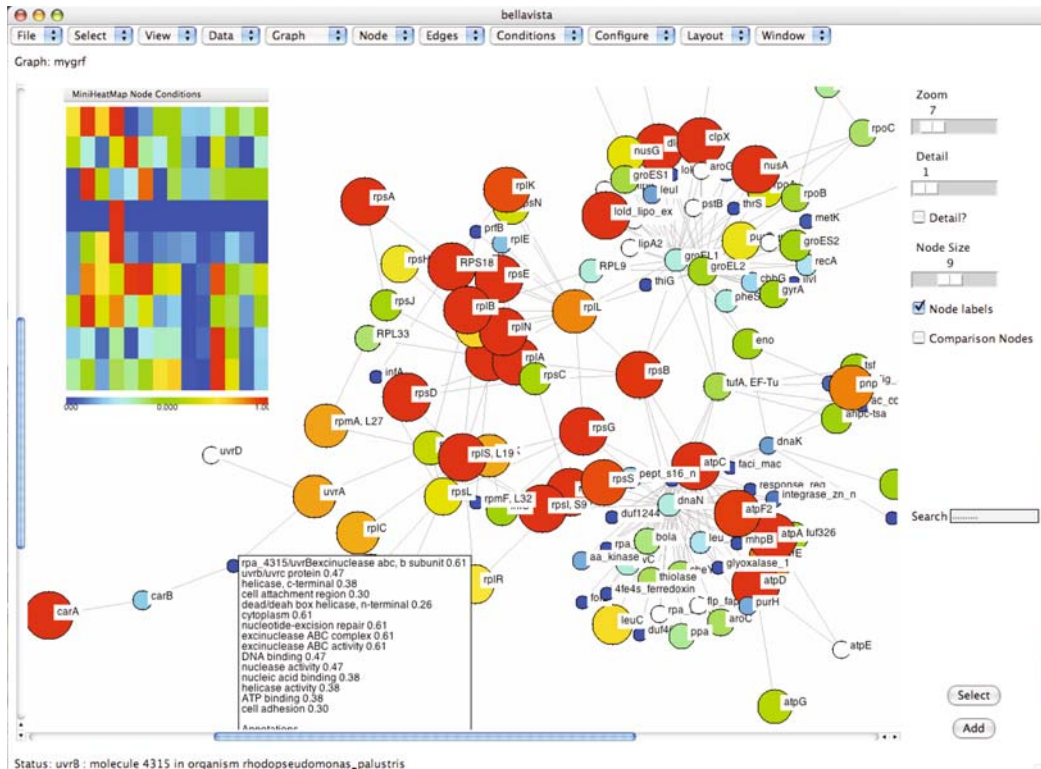


Fig. 22.9. BellaVista (44), a standalone biological information viewer written in Python, visualises proteins, protein properties and protein relationships as nodes in a graph, node attributes and edges between nodes. This flexible application can present information loaded from local files or obtained via the Bioverse API. This includes manual and predicted annotations, protein identifiers, protein similarity relationships (incorporating sources such as the PDB) and Interolog relationships. This screenshot shows a network of protein relationships from *Rhodospseudomonas palustris* overlaid with experimentally obtained proteomics data. Protein abundance levels under different organism growth conditions are integrated into this view from a local file and represented as shades and sizes of nodes. The inset window shows a heat map of members of a putative protein complex under six growth conditions. A pop-up box dynamically lists Bioverse annotations of the selected protein retrieved via the Bioverse API.

6.3. API Usage

The Bioverse Application Programming Interface is a language-agnostic interface to the data and methods implemented in the Bioverse. It allows a programmer to query the Bioverse in many ways. Its capacity is sufficient to allow someone to recreate the Bioverse Web Application in the form of a desktop application.

At the time of writing, there are more than two dozen functions, or methods, publicly available. Because the API is updated periodically, the online documentation should be referenced for the latest information. The examples in the following section are valid at the time of writing, but the API may have changed since then. However, the general idea of accessing the API remains the same. (See Section 6.6 regarding historical changes to the API.)

All methods are available for use by a programmer, but some are intended for use by the Bioverse Web Application and return data that are specific to the way the web application expects it, making them less useful in the general case. Labels, such as `general` or `webapp`, assigned to methods in the API documentation hint at the intended usage.

There are presently two ways of accessing the API. One way is via the standardised XML-RPC protocol and the other is via a customised JSON interface.

6.3.1. XML-RPC

The XML-RPC protocol (43) hides the complexity of encoding method requests and decoding method responses into and from XML. The result is that a natural syntax can be used to interface with the server. Applications which utilise the XML-RPC protocol include BellaVista (44) and the new Integrator code base (42). The API usage examples (Section 6.5) utilise the XML-RPC interface.

6.3.2. JSON

JavaScript Object Notation (JSON) (45) allows encoding of the same data structures that XML-RPC can encode: numbers, strings, lists, and associative arrays. Unlike XML-RPC, JSON is native to JavaScript and, incidentally, to Python. This convenient confluence makes it a useful dialect of communication between the Bioverse server and the Bioverse Web Application implemented in JavaScript.

The JSON interface is an in-house solution that addresses the problem of the Bioverse Web Application needing to access the same methods as are exposed by the XML-RPC interface. The solution is to encode the request into an HTTP GET query. The server responds to this query with a data structure encoded in the JSON format. That format can be natively and efficiently interpreted by JavaScript.

Incidentally, due to the simplicity of the encoding syntax, it is feasible for a programmer to inspect a JSON response to explore the API or prepare for application implementation. The online API documentation provides example method requests that return JSON data for this purpose (Section 6.4).

6.4. API Documentation

The API documentation for each method is formatted for the programmer in a convenient way (Fig. 22.10). The three sections of this documentation describe the method's operation and provide usage examples. The examples for the 'Web' section are URLs that return JSON-formatted data. These URLs can be clicked in a browser (to see the JSON-formatted data), embedded into a web application (such as the Bioverse Web Application) or used anywhere the JSON format is convenient. The 'XML-RPC' example code is implemented in Python and serves to illustrate the basic idea of calling the method.

Method reference for **molecule_by_name**

Labels: general

Documentation:

Molecules having specified molecule name

Input: keys: molecule_name (string), organism_id (int), db_name (string), limit (int)
Output: a list of matching molecules and their names

- molecule_name is required
 - it is used as a substring and checked for containment in the molecule name field
 - it is case insensitive.
- organism_id and db_name are exclusive.
 - if both given, organism_id overpowers.
 - if neither given, all organisms are considered.
- limit is optional; if omitted, default is 10.

Web example:

```
http://bioverse.compbio.washington.edu/api/web/molecule_by_name?organism_id=7&molecule_name=hox
http://bioverse.compbio.washington.edu/api/web/molecule_by_name?molecule_name=hox
http://bioverse.compbio.washington.edu/api/web/molecule_by_name?db_name=homo_sapiens&molecule_name=H&limit=77
```

XML-RPC example, in Python:

```
import xmlrpclib
server = xmlrpclib.ServerProxy("http://bioverse.compbio.washington.edu/api/xmlrpc/")

print server.molecule_by_name({"organism_id":7, "molecule_name":"hox"})
print server.molecule_by_name({"molecule_name":"hox"})
print server.molecule_by_name({"db_name":"homo_sapiens", "molecule_name":"H", "limit":77})
```

Fig. 22.10. Documentation for the molecule_by_name API method.

6.5. API Usage Examples

Here we provide and explain usage examples currently on the web site. Care should be taken to ensure that the latest API documentation is being referenced. Current examples of using the API are present online at <http://bioverse.compbio.washington.edu/api>.

In the Python programming environment, accessing the API is trivial. Python will be used throughout this section to illustrate API principles. For example, Program 1 shows how to retrieve a list of all organisms catalogued by the Bioverse.

```
import xmlrpclib
B = xmlrpclib.ServerProxy("http://bioverse.compbio.washington.edu/api/xmlrpc/")
print B.organism()
```

Program 1 Listing all organisms catalogued by the Bioverse.

6.5.1. Molecules Annotated by a Single Function

A molecule is annotated by one or more functions. Each function is described by a function identifier (function_id) and a text label. For example, ribulose-phosphate binding barrel is a description of InterPro entry I1060 and is given the Bioverse function identifier of 153787. Program 2 illustrates how we can find all molecules annotated with this function in the organism *Homo sapiens*.

```

(a)
mols = B.molecule_by_function({ "organism_id":35,
    "function_id":153787} )

(b)
[{"bioverse_id": 64,
  "function_confidence": 0.0,
  "function_desc_1": "RibP_bind_barrel",
  "function_desc_2": "Ribulose-phosphate binding
    barrel",
  "function_id": 153787,
  "function_name": "IPR011060",
  "organism_id": 35} ,

{"bioverse_id": 413,
  "function_confidence": 0.25673899999999999,
  "function_desc_1": "RibP_bind_barrel",
  "function_desc_2": "Ribulose-phosphate binding
    barrel",
  "function_id": 153787,
  "function_name": "IPR011060",
  "organism_id": 35} ,

...
]

```

Program 2 (a) Retrieving molecules in the organism *Homo sapiens* (identified by the organism identifier `organism_id` 35), which are annotated with function identifier 153787 (ribulose-phosphate binding barrel). (b) Partial list of resulting dictionaries stored in the variable `mols`.

The Bioverse function identified by `function_id` 153787 might have been found earlier with the `function_search` method, which finds functions that have a description containing a search string. In anticipation of the two-step process of obtaining matching function identifiers and molecules that are annotated with those identifiers, the `molecule_by_function` method can accept a text string as an argument (like `function_search`), match it against function descriptions and return molecules that are annotated by those functions (Program 3).

```

mols = B.molecule_by_function(
    { "organism_id":35,
      "function_text":"Ribulose-phosphate binding
        barrel"} )

```

Program 3 Alternative version of code in Program 2.

It is important to keep in mind that ribulose-phosphate binding barrel is just a text string with which a simple text search is performed. This means that a search for ribulose will match all

functions containing this string like ribulose-phosphate 3-epimerase, L-ribulose-phosphate 4-epimerase activity, bifunctional ribulose 5-phosphate reductase/CDP-ribitol pyrophosphorylase and, by connection, many more molecules. The string ribulose-phosphate binding barrel was chosen to match only one function in the above example query.

6.5.2. Molecules Annotated by Multiple Functions

In this example we are interested in finding all molecules in *Drosophila melanogaster* (fruit fly), which are annotated by functions containing kinase or phosphate in their descriptions. The `function_text` search argument of `molecule_by_function` will be inadequate because of the simple text containment search it performs; it does not accept Boolean expressions. We must instead accumulate a list of functional annotations that match the description kinase or phosphate separately (Program 4).

```
an = []
an += B.function_search({ "q": "kinase" })
an += B.function_search({ "q": "phosphate" })
```

Program 4 Accumulate functional annotations that contain kinase and phosphate in their description.

Because each function has a unique `function_id` integer associated with it, which we will need later, we can extract this unique list using standard Python techniques (Program 5).

```
function_ids = list(set([ a["function_id"] for a
in an ]))
```

Program 5 Get unique list of function identifiers.

To retrieve a list of molecules matching any of these functions, we will use the `molecule_by_function` method, but now provide it with a list of function identifiers instead of a single value (Program 6). Given this set of molecules, we can print out their various properties (Program 7).

```
mols=B.molecule_by_function({ "organism_id":7,
                               "function_id":function_ids,
                               "limit":0})
```

Program 6 Get all molecules matching functional annotations (as identified in the list `function_ids`) in *Drosophila melanogaster*, organism 7.

More sophisticated searches can be performed by the method `molecule_search`, and additional molecule information can be extracted with methods such as `molecule_function` and `molecule_interaction`, which are documented online.

```
for m in mols:
    items = [ m["bioverse_id"],
             round(m["function_confidence"],2), #
```

```

        confidence to 2 decimal places
m[ "function_name" ], # function name like
GO:12345
m[ "function_desc_1" ], # function
description 1
m[ "function_desc_2" ] # function
description 2 (if available)
]
print "\t".join([ str(x) for x in items])

```

Program 7 Display identifiers (*bioverse_id*) of each molecule in the list *mols*, and a confidence, a name and two descriptions for each function annotation separated by tabs.

6.6. Versioning of the API Methods

Over time, changes to the API are expected. To keep historical perspective, a list of changes is documented at <http://bioverse.compbio.washington.edu/api/versions> and old versions of the API remain accessible as long as possible and necessary. This allows for work on an updated version to be underway while maintaining continuity in application behavior.

7. Comparison to Other Similar Projects

The goal of our efforts is a common shared dream among all biologists: to understand how the genome of an organism characterises the development and behaviour of the organism. From a bioinformatics viewpoint, the goal is to organise all the world's biological information to provide semantic meaning through complex models that ultimately model all relationships that occur in life, from atomic level interactions to organismal ones. To this end, several groups have created resources to accomplish goals similar to those outlined here. Some examples include Ensembl (46), Biozon (47), BIND (21), MIPS (22), GRID (48), DIP (23), KEGG (49), 3D-Genomics (50), InterPro (8), PEDANT (51, 52), STRING (53), and Predictome (54). A variety of methods also exist for protein structure, function, and interaction prediction (*see* web server issues of *Nucleic Acids Research*), which can be applied in a large-scale manner to whole proteomes, but in many cases that has not been done or the resulting data are not made available over the web.

In general, annotation databases can be grouped into two categories: those that are both sequence- and structure-oriented and those that are only sequence-oriented. The latter databases

can process larger amounts of data since the amount of structural data is limited, and structural calculations may be time-consuming. Pathway and other network-type databases, such as Predictome (54), are mostly sequence-oriented. Not all databases are comprehensive, i.e. they do not seek to mine data from all completely sequenced genomes simultaneously, or they limit themselves to proteins that are well characterised. Many of the interaction databases are limited to experimentally derived data or manual annotations (21, 23, 48). The databases and software differ in terms of ease-of-use and access to data; some provide bare tables and lists of information whereas others provide some form of abstraction (such as depictions of networks and cellular systems). Few provide programming interfaces, though this trend is changing. There are software-only projects such as Cytoscape (55), which provides a reasonable user interface, but the data for analysis must be explicitly provided to the program instead of referring to a centrally maintained and frequently updated database. Still others that provide predicted interaction information (56) are limited to only a few organisms or do not perform novel structural and functional annotation of the interacting proteins (54).

Compared to these projects, the strength of the Bioverse is primarily in our background of developing three-dimensional protein structure and function modelling tools for the past 14 years (17–20, 29–41), which augment the integration of existing data with novel predictions. However, in perspective, all the current and future projects yield complementary information to the scientific community, and it is the synergy of these efforts that is most valuable to the bench biologist seeking to solve a particular domain-specific research problem.

8. Bioverse Technology

All core components of the Bioverse are written in the Python(57) programming language running on the Linux operating system. The data warehouse is implemented upon the PostgreSQL(58) relational database that resides on a RAID storage array and presently occupies 1.3 terabytes. The web server daemon utilises various free software packages such as CherryPy(59) and HTML Templates(60). The web application is written in-house with limited support from external libraries. JavaScript templates (61) are used for in-browser content rendering.

Acknowledgements

We acknowledge the invaluable help in the form of comments, contributions, and critiques of the Bioverse from all members of the Samudrala group and the Department of Microbiology at the University of Washington.

Many researchers have helped in the creation of the Bioverse and Protinfo web servers. We thank the scientific community (more properly attributed in **Section 3.2**) for making available data and techniques we have used and relied on.

This work was and is currently supported in part by the University of Washington's Advanced Technology Initiative in Infectious Diseases, Puget Sound Partners in Global Health, NSF CAREER Grant, NSF Grant DBI-0217241, NIH Grant GM068152 and a Searle Scholar Award to Ram Samudrala.

References

1. J. Yu, J. Wang, W. Lin, et al. The genomes of *Oryza sativa*: a history of duplications. *Public Libr. Sci. Biol.* 3: e38 (2005).
2. S. Kikuchi, K. Satoh, T. Nagata, et al. Collection, mapping, and annotation of over 28,000 cDNA clones from japonica rice. *Science.* 301: 376–379 (2003).
3. The Gene Ontology Consortium. Gene ontology: tool for the unification of biology. *Nat. Genet.* 25: 25–29 (2000).
4. J. Cherry, C. Adler, C. Ball, et al. SGD: Saccharomyces genome database. *Nucl. Acids Res.* 26: 73–79 (1998).
5. T. Harris, N. Chen, F. Cunningham, et al. WormBase: a multi-species resource for nematode biology and genomics. *Nucleic Acids Res.* 32: D411–D417 (2004).
6. F. Consortium. The FlyBase database of the *Drosophila* genome projects and community literature. *Nucleic Acids Res.* 31: 172–175 (2003).
7. S. Peri, J. D. Navarro, R. Amanchy, et al. Development of human protein reference database as an initial platform for approaching systems biology in humans. *Genome Res.* 13(10): 2363–2371 (2003).
8. R. Apweiler, T. Attwood, A. Bairoch, et al. InterPro-an integrated documentation resource for protein families, domains and functional sites. *Bioinformatics.* 16: 1145–1150 (2000).
9. H. M. Berman, J. Westbrook, Z. Feng, et al. The protein data bank. *Nucl. Acids Res.* 28: 235–242 (2000).
10. A. G. Murzin, S. E. Brenner, T. Hubbard, C. Chothia. SCOP: a structural classification of proteins database for the investigation of sequences and structures. *J. Mol. Biol.* 247: 536–540 (1995).
11. T. Hubbard, A. Murzin, S. Brenner, C. Chothia. SCOP: a structural classification of proteins database. *Nucleic Acids Res.* 25: 236–239 (1997).
12. L. Lo Conte, S. E. Brenner, T. J. P. Hubbard, C. Chothia, A. G. Murzin. SCOP database in 2002: refinements accommodate structural genomics. *Nucl. Acids Res.* 30(1): 264–267 (2002).
13. A. Andreeva, D. Howorth, S. E. Brenner, et al. SCOP database in 2004: refinements integrate structure and sequence family data. *Nucl. Acids Res.* 32 (2004).
14. J. Gough, K. Karplus, R. Hughey, C. Chothia. Assignment of homology to genome sequences using a library of hidden Markov models that represent all proteins of known structure. *J. Mol. Biol.* 313: 903–919 (2001).
15. J. Gough, C. Chothia. SUPERFAMILY: HMMs representing all proteins of known structure. SCOP sequence searches,

- alignments and genome assignments. *Nucleic Acids Res.* 30: 268–272 (2002).
16. L. McGuffin, K. Bryson, D. Jones. The PSIPRED protein structure prediction server. *Bioinformatics.* 16: 404–405 (2000).
 17. R. Samudrala, J. Moult. A graph-theoretic algorithm for comparative modelling of protein structure. *J. Mol. Biol.* 279: 287–302 (1998).
 18. R. Samudrala, Y. Xia, E. Huang, M. Levitt. *Ab initio* protein structure prediction using a combined hierarchical approach. *Prot.: Struct. Funct. Genet.* S3: 194–198 (1999).
 19. E. Huang, R. Samudrala, J. Ponder. *Ab initio* fold prediction of small helical proteins using distance geometry and knowledge-based scoring functions. *J. Mol. Biol.* 290: 267–281 (1999).
 20. Y. Xia, E. Huang, M. Levitt, R. Samudrala. *Ab initio* construction of protein tertiary structures using a hierarchical approach. *J. Mol. Biol.* 300: 171–185 (2000).
 21. G. Bader, D. Betel, C. Hogue. BIND: the biomolecular interaction network database. *Nucleic Acids Res.* 31: 248–250 (2003).
 22. H. Mewes, D. Frishman, U. Guldener, et al. MIPS: a database for genomes and protein sequences. *Nucleic Acids Res.* 30: 31–34 (2002).
 23. I. Xenarios, L. Salwinski, X. Duan, et al. DIP, the database of interacting proteins: a research tool for studying cellular networks of protein interactions. *Nucleic Acids Res.* 30: 303–305 (2002).
 24. L. Matthews, P. Vaglio, J. Reboul, et al. Identification of potential interaction networks using sequence-based searches for conserved protein-protein interactions or “interologs”. *Genome Res.* 11: 2120–2126 (2001).
 25. J. McDermott, R. Bumgarner, R. Samudrala. Functional annotation from predicted protein interaction networks. *Bioinformatics.* 21: 3217–3226 (2005).
 26. Computing. <http://compbio.washington.edu/computing.html>.
 27. S. Altschul, T. Madden, A. Schaffer, et al. Gapped BLAST and PSI-BLAST: a new generation of database programs. *Nucleic Acids Res.* 25: 3389–3402 (1997).
 28. HMMER: biosequence analysis using profile hidden Markov models. <http://hmmer.janelia.org>.
 29. L.-H. Hung, R. Samudrala. PROTIINFO: secondary and tertiary protein structure prediction. *Nucleic Acids Res.* 31: 3736–3737 (2003).
 30. L. Hung, S. Ngan, T. Liu, R. Samudrala. PROTIINFO: new algorithms for enhanced protein structure predictions. *Nucleic Acids Res.* 33: W77–W80 (2005).
 31. L.-H. Hung, R. Samudrala. An automated assignment-free Bayesian approach for accurately identifying proton contacts from NOESY data. *J. Biomol. NMR.* 36: 189–198 (2006).
 32. L.-H. Hung, R. Samudrala. Accurate and automated assignment of secondary structure with PsiCSI. *Protein Sci.* 12: 288–295 (2003).
 33. K. Wang, J. A. Horst, G. Cheng, D. C. Nickle, R. Samudrala. Protein Meta-Functional Signatures from Combining Sequence, Structure, Evolution, and Amino Acid Property Information. *PLoS Computational Biology* 4(9): e1000181 (2008).
 34. G. Cheng, B. Qian, R. Samudrala, D. Baker. Improvement in protein functional site prediction by distinguishing structural and functional constraints on protein family evolution using computational design. *Nucleic Acids Res.* 33: 5861–5867 (2005).
 35. K. Wang, R. Samudrala. FSSA: a novel method for identifying functional signatures from structural alignments. *Bioinformatics.* 21: 2969–2977 (2005).
 36. G. Cheng, R. Samudrala. An all-atom geometrical knowledge-based scoring function to predict protein metal ion binding sites, affinities and specificities. *manuscript in preparation* (2007).
 37. E. Jenwitheesuk, K. Wang, J. Mittler, R. Samudrala. PIRSpred: a web server for reliable HIV-1 protein-inhibitor resistance/susceptibility prediction. *Trends Microbiol.* 13: 150–151 (2005).
 38. E. Jenwitheesuk, R. Samudrala. Prediction of HIV-1 protease inhibitor resistance using a protein-inhibitor flexible docking approach. *Antiv. Ther.* 10: 157–166 (2005).
 39. R. Jenwitheesuk, K. Wang, J. Mittler, R. Samudrala. Improved accuracy of HIV-1 genotypic susceptibility interpretation using a consensus approach. *AIDS.* 18: 1858–1859 (2004).
 40. K. Wang, E. Jenwitheesuk, R. Samudrala, J. Mittler. Simple linear model provides highly accurate genotypic predictions of HIV-1 drug resistance. *Antiv. Ther.* 9: 343–352 (2004).
 41. K. Wang, R. Samudrala. Automated functional classification of experimental and

- predicted protein structures. *Bioinformatics*. 7: 278–277 (2006).
42. A. Chang, J. McDermott, Z. Frazier, M. Guerquin, R. Samudrala. INTEGRATOR: interactive graphical search of large protein interactomes over the web. *Bioinformatics*. 7: 146–110 (2006).
 43. XML-RPC Home Page. <http://www.xmlrpc.com>.
 44. J. McDermott, M. Guerquin, Z. Frazier, R. Samudrala. BellaVista: a flexible visualization environment for complex biological information. *manuscript in preparation* (2007).
 45. JSON. <http://www.json.org/>.
 46. E. Birney, D. Andrews, P. Bevan, et al. Ensembl 2004. *Nucleic Acids Res.* 32: D468–D470 (2004).
 47. A. Birkland, G. Yona. BIOZON: a hub of heterogeneous biological data. *Nucl. Acids Res.* 34: D235–D242 (2006).
 48. B. Breitkreutz, C. Stark, M. Tyers. The GRID: the general repository for interaction datasets. *Genome Biol.* 4: 744120 (2003).
 49. M. Kanehisa, S. Goto, S. Kawashima, A. Nakaya. The KEGG databases at GenomeNet. *Nucleic Acids Res.* 30: 42–46 (2002).
 50. K. Fleming, A. Muller, R. MacCallum, M. Sternberg. 3D-GENOMICS: a database to compare structural and functional annotations of proteins between sequenced genomes. *Nucleic Acids Res.* 32: D245–D250 (2004).
 51. D. Frishman, M. Mokrejs, D. Kosykh, et al. The PEDANT genome database. *Nucleic Acids Res.* 31: 207–211 (2003).
 52. M. L. Riley, T. Schmidt, C. Wagner, H.-W. Mewes, D. Frishman. The PEDANT genome database in 2005. *Nucl. Acids Res.* 33: D308–D310 (2005).
 53. C. von Mering, M. Huynen, D. Jaeggi, et al. STRING: a database of predicted functional associations between proteins. *Nucleic Acids Res.* 31: 258–261 (2003).
 54. J. Mellor, I. Yanai, K. Clodfelter, J. Mintseris, C. DeLisi. Predictome: a database of putative functional links between proteins. *Nucleic Acids Res.* 30: 306–309 (2002).
 55. P. Shannon, A. Markiel, O. Ozier, et al. Cytoscape: a software environment for integrated models of biomolecular interaction networks. *Genome Res.* 13: 2498–2504 (2003).
 56. H. Yu, N. Luscombe, H. Lu, et al. Annotation transfer between genomes: protein-protein interologs and protein-DNA regulogs. *Genome Res.* 14: 1107–1118 (2004).
 57. Python Programming Language – Official Website. <http://www.python.org>.
 58. PostgreSQL: The world’s most advanced open source database. <http://www.postgresql.org>.
 59. CherryPy. <http://www.cherrypy.org>.
 60. htmltmpl templating engine. <http://htmltmpl.sourceforge.net>.
 61. trimpath – Google Code. <http://code.google.com/p/trimpath>.

Chapter 23

Computational Representation of Biological Systems

Zach Frazier, Jason McDermott, Michal Guerquin, and Ram Samudrala

Abstract

Integration of large and diverse biological data sets is a daunting problem facing systems biology researchers. Exploring the complex issues of data validation, integration, and representation, we present a systematic approach for the management and analysis of large biological data sets based on data warehouses. Our system has been implemented in the Bioverse, a framework combining diverse protein information from a variety of knowledge areas such as molecular interactions, pathway localization, protein structure, and protein function.

Key words: Bioverse, data integration, molecular interactions, protein structure, protein function, data warehouse, database, bioinformatics.

1. Introduction

As high-throughput and other large data sets are generated, the ability of researchers to organize and analyze these data will determine the science that can be accomplished. Successful integration of diverse data sources provides novel insight into biological processes. For example, the combination of data sets has been used to discover novel protein–protein interactions in the galactose utilization pathways of yeast (1, 2). In the Bioverse, the application described here, proteins have been annotated with functional descriptions by combining the existing and predicted interaction networks and the existing functional annotations (3).

Integrating biological resources pose many problems for researchers. Resources are designed and developed with a specific user community in mind and, with this specialization, have developed a particular data focus, storage format, and query interface.

Developing tools to utilize these resources demands both an investment of time and often specific knowledge of the resource. Objects of interest have different identifiers in different contexts, complicating accurate integration. Independent projects collect different information for similar data sets, and may use different standards of measurement. The query interfaces provided for the resource may be restrictive, not allowing for novel uses. For example, using web sites for blast queries to find similar proteins is reasonable for a handful of interesting proteins, but for a large data set it is easier to perform the queries against a local database.

The focus of many biological databases is necessarily narrow, either focused exclusively on single organisms, such as Wormbase (4), databases of structures (5, 6), or pathways (7). Manually integrating the results from many data sources may be feasible for focused questions or small studies, but is time-consuming for large data sets. Several projects have attempted to solve this problem, acting as an intermediary between databases, thereby solving the problem of integration; however, since these often work through the interfaces provided, the throughput of this approach is limited. Services such as BioMoby (8), REMORA (9), and the Bioinformatics Resource Manager (10) successfully integrate a variety of data sources and bioinformatics tools. These are excellent resources for small queries across many different databases.

For larger projects, we instead integrate the entire resource. We begin with the raw data provided by the resource maintainers, and develop our own storage system integrated with other data sources based on data warehousing principles.

Data warehouses are an approach to data integration and management, which is used for a variety of problem domains. In addition to maintaining a highly flexible storage system for data, data warehouses allow for the expression of complex relationships and ease the construction and execution of complex queries.

The solutions developed in the Bioverse (11) integrate a wide variety of biological data sources, allowing for exploration and prediction of functional, structural, and sequence-based data analysis.

2. Data Warehouses

Data warehouses organize data for analysis and data mining applications. Although they are built on relational database technology, data warehouses differ from traditional online transaction processing (OLTP) databases. Instead, they are designed to support online analytical processing (OLAP). OLTP systems typically support many concurrent users inserting, deleting, and modifying small amounts of data. OLAP systems provide management and

processing of multidimensional data for analysis. The structure and organization of the data models are different for each application type. Data warehouses are built on an OLAP model. An excellent review of data warehouses is Kimball (12).

2.1. Relational Databases

At the center of the data management system is the relational database. Relational algebra was introduced by Codd (13). A large software industry based on his work quickly appeared, and a query language based on relational algebra, Structured Query Language (SQL), has become the standard for most commercial relational databases.

For our purposes, a database is a collection of tables, indexes, relations, and functions. The tables are collections of objects with identical attributes. The attributes are represented as the columns of the tables, and the objects are stored as the rows of the table. Data indexing and custom database functions optimize the access patterns. Interactions with the system are based on transactions, which guarantee the data integrity in the face of unexpected system or process failures. Transactions represent a fundamental atomic action in the database. In the event of an error a transaction is aborted and all changes can be rolled back to assure data consistency.

2.2. Dimensional Models

As specializations of relational databases, the distinguishing feature of a data warehouse is the organization of the data. Traditional OLAP database design methodology focuses on normalized tables. Normalization provides logical separation of data, moving all redundant data to tables, which are referenced as foreign keys. Since the activity in these databases consists of many small transactions, normalization localizes the effects of changes on the database. This design goal is relaxed for data warehouses that have different requirements. Normalization is sacrificed for expressive and efficient queries across large data sets.

Query writing for the dimensional model is straightforward. Queries can easily be constructed, since the relationships between tables are simple and designed for flexibility. Queries against the central fact table will use filters on the linked dimension tables to narrow the focus of the query.

Using a data warehouse provides several benefits. The approach makes the information accessible to more general queries than traditional data schemas, and it is flexible with changes and updates to the underlying data model having minimal impact on the data model organization. New data types can be added without disturbing the existing table structure, and new dimensions can be added to a fact with minimal disruption.

2.2.1. Facts

Facts are the data points of the system. They are the generated or computed measurements that are the focus of the representation, and are defined in terms of the measurement conditions and

parameters. Each fact type is stored in a corresponding “Fact Table”. Fact tables are large, many containing millions of rows. Several columns of the fact table will be foreign keys, describing in detail “dimensions” of the facts. Other columns will be numeric or categorical data, the “measures” of the fact. The redundancy of storing most of the data in a large central table, with a handful of satellite tables, allows for flexibility at the cost of some redundancy. As an example of a fact table, consider the `molecule_sequence` table of **Fig. 23.2**.

2.2.2. Dimensions

Dimensions represent the complex attributes of the facts. These columns of the fact table are pointers to other tables or foreign keys. These are the features of facts that themselves have many features, which would be useful query filters. The features are stored in “dimension tables” that describe a particular feature of the fact in detail. These tables rarely change, and encapsulate a small set of specific data.

2.2.3. Measures

Measures are the parameters that make up the facts. These are discrete values and are usually numeric and additive. Being additive allows for summary queries on the fact table. Simpler than dimensions, these do not have associated attributes. These are simple columns of the fact table.

2.2.4. Star Schemas

This organization yields a “star schema” with the fact table at the center, surrounded by many dimension tables. These structures are the goal of data warehouse design. While not highly normalized like many database designs, the star schema allows for complex queries to be made over fact tables efficiently. Filters on the associated dimension tables and measures provide a flexible constraint-based search system, which can adapt to a wide variety of questions, allowing researchers to identify and isolate relevant facts of interest. In some cases the star schema may have a depth of more than a single table. These snowflake schemas, although sometimes necessary, should be avoided, as they make query writing complicated and can impact performance.

A data warehouse will contain several fact tables, which may share dimensions. Each fact table and the associated dimensions are considered a distinct “data mart”. Generally, the data warehouse will consist of several independent data marts, which have an independent focus, but which share a few dimension tables.

There are many efforts to standardize data representation in systems biology. Systems Biology Markup Language (SBML) (14) is a data exchange format designed for pathway

and network models. Another particularly successful example is the PSI-MI (15) format for interaction data sets. In the cases where a well-defined data specification exists, it is easier to design the database tables with the given specification as a guide. While this method may not provide a true warehouse design, it is a step toward the goal of integration and efficiency. The staging data of the Bioverse, for example, closely mirrors in structure the PSI-MI format.

3. Data Warehouse Construction

Development of the data warehouse from source data to completed database is an integrated process which includes both the development of a data model and the development of the tools required to load large quantities of data.

3.1. Model Design

3.1.1. Facts and Granularity

The first decision to be made in model design is the focus of the fact tables. Collecting and storing data at the wrong resolution will impact the ability of the warehouse to answer research questions. If highly specific data are collected, it may be impossible to construct queries on relevant aggregates. On the other hand, queries will not be able to filter well if the fact tables or the dimension tables are too general.

3.1.2. Dimensions

The characteristics of the fact table are stored in the dimension tables. The choice of columns in the dimension tables determines the queries that are supported. Verbosity and redundancy are acceptable since support for rich analysis is the goal. Although storage space is a consideration for large databases, the dimension tables even without normalization will not represent a major storage problem. Typically, the fact tables that are relatively compact will have orders of magnitudes more rows than dimension tables.

3.2. Extracting, Transforming, and Loading

The migration of data from its native source format into a warehouse is commonly referred to as the Extraction, Transformation, and Loading step (ETL). Data can be extracted from other databases or text files.

The design and implementation of ETL tools is one of the most time-consuming aspects of data warehouse development. Complex rules and transformations must be applied and errors must be dealt with intelligently. Many of these steps take place in a staging area of the warehouse. **Fig. 23.1.**

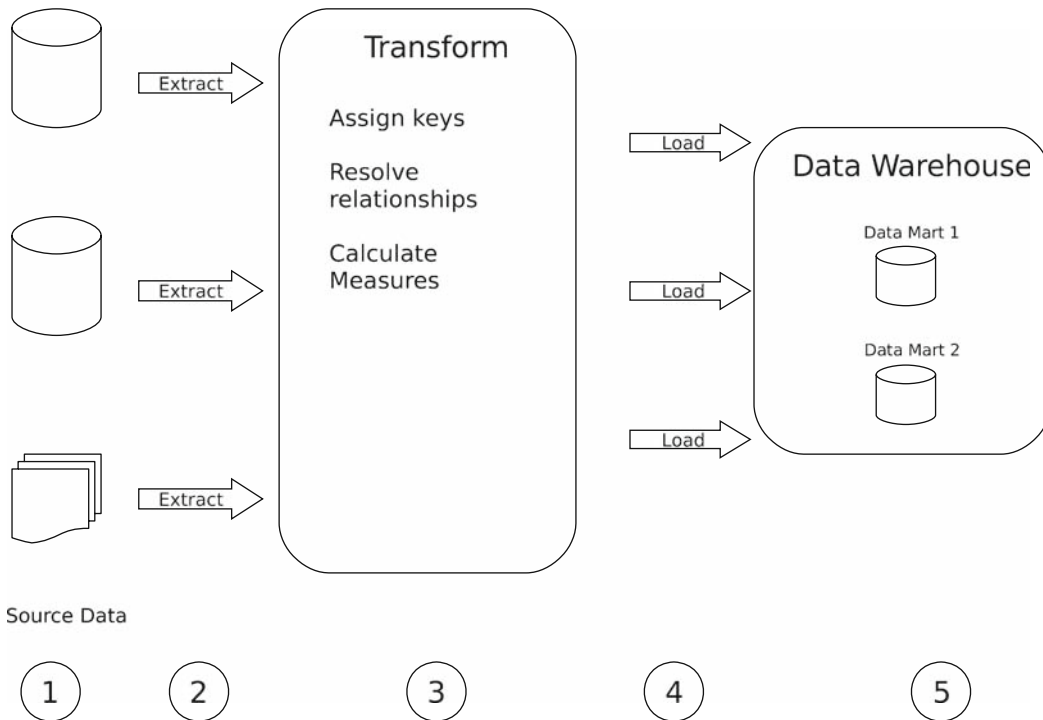


Fig. 23.1. The ETL process. (1) Data sources such as downloaded versions of projects, and the results of bioinformatics tools that have been run against local data sets. (2) All of these data are extracted from their original storage formats and parsed. (3) The data are transformed. Additional fields are calculated, lookups to the database are made, and data sets are validated against existing data. (4) The data are loaded in the data warehouse. (5) The data exist in a set of data marts, which are accessed from a range of applications and analytical tools.

3.2.1. Extraction

Programs process the source data, extracting the relevant information and filtering data that will not be used or have already been loaded. The data will be moved to either text files or to a separate area of the database, depending on the transformation steps necessary.

3.2.2. Transformation

Transformations cover many different facets of data management and are often the most complex part of the ETL process. A variety of operations are carried out in the transformation step. Validation and verification ensure that the data are well formed and adhere to any range or value constraints and that, if attributes are referenced, new attributes of the data can be computed based on the existing data. Data can be grouped and merged or split to provide for data at a more appropriate scale. Lookups can be made against a database to fill in a variety of columns such as foreign key values, or unique ids generated by the database.

3.2.3. Loading

Finally, when all of the data are prepared, they are loaded into the data warehouse. This step is optimized for speed. Lookups in the database are avoided, having been accomplished in the transformation step.

3.2.4. Metadata

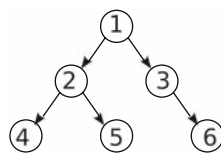
It is important to keep track every piece of data during processing. In the Bioverse we track a number of features of data including its source, what transformations have been carried out, any errors or warnings encountered, time required to process, and more. This metadata provides important information that helps with data management. In our case this is stored in the database where all the ETL tools responsible for data processing can log activity in relation to the data sets they operate on.

4. Bioverse Model Design

The Bioverse includes a variety of information types as well as sources. Interaction data, functional annotations, structure classification, and sequence similarity data types are present. Each type of data requires unique tools for ETL as well as distinct storage in the staging area and the warehouse. Here we will look at a handful of tables and discuss the design choices that were made.

4.1. Sequence Data

Sequence data is stored in the `molecule_sequence` table (Fig. 23.2), which contains information about the type of the sequence as well as dimensions about the source of the sequence.



Transitive closure table				
Parent	Child	Distance	Is root	Is leaf
1	2	1	T	F
1	4	2	T	T
1	5	2	T	T
2	4	1	F	T
2	5	1	F	T
3	6	1	F	F
3	6	1	F	T
1	6	2	T	T

Adjacency Table	
Parent	Child
1	2
1	3
2	4
2	5
3	6

Fig. 23.2. A portion of the `molecule_sequence` data mart. The molecule sequence table is the central fact. There are three-dimension tables shown, **taxon**, **molecule_type**, and **alphabet**. Each of these has many interesting features, which can be used to limit searches on the `molecule_sequence` table. Additionally, the `molecule_sequence` table includes several measures, such as `seq_len` shown as the last visible column.

A number of indices are also created on the columns of the sequence data and the associated dimensions to accommodate fast queries of features. The MD5 hash of the sequence is stored for fast exact matches, along with a hash of the first 10 and the reverse of the last 10 residues in the case of proteins for fast matching of the start and end of sequences.

4.2. Taxonomy Data: A Hierarchy Example

Hierarchical data occurs throughout the Bioverse. Representation of these structures is particularly difficult in relational databases. The choice of representation scheme depends on the relevant queries. The most straightforward storage solution is to store all parent-child relationships of the hierarchy. This is efficient in storage space, but only allows for a narrow range of queries about table structure. Different table structures are more useful for queries about depth and relationships between entities in the hierarchy. Here we present an approach for storing hierarchical data in the database, which is based on the topological closure of the paths in the hierarchy. This storage mechanism, which is relatively large and expensive to compute, provides a fast mechanism for a wide variety of hierarchical queries. This approach is described in Chapter 5.6 of Kimball (12). Several dimensions in the Bioverse are structured in hierarchies including the Gene Ontology (16), SCOP (17), and the NCBI taxonomy database (18). In each case we use this technique to increase the potential filters on these dimensions.

The taxonomy data provided by the NCBI is used by many different data marts in the Bioverse. As a dimension it needs to be filtered in several ways. A simple list of taxon entries is stored in the taxon table in **Table 23.1**. To provide for more complex queries about relative positions, we calculate the topological closure of the taxonomy tree. This closure is a collection of all paths in the tree. We record the ancestor and child node, as well as the distance between them, and other information about their place in the tree. An example topological closure calculation is given in **Fig. 23.3**. Depending on the query, a fact table can refer to either the original table or the table holding the topological closure. Since the topological closure table has an additional reference to the original table, certain filters will go through two tables, a use of a snowflake schema.

When representing the topological closure, additional columns are necessary whether or not this path is the shortest path, the number of paths between these nodes, and the number of shortest paths between the nodes. While this information is redundant for the taxonomy example where every node only has a single parent, in the event of more complex topologies, these columns are useful for filtering as well. As an example the Gene Ontology hierarchy allows for many parents, making it a directed acyclic graph (DAG) rather than a tree.

Table 23.1

The basic taxon table. No hierarchical information is present. This is a simple dimension table describing taxonomy entries. Columns such as `division_id`, `genetic_code_id`, and `mito_genetic_code_id` refer to other tables built on data provided by the taxonomy database. The various ranks such as kingdom, phylum, and family are stored for each entry to allow for queries about subtree position. To enable some tree queries, relevant information such as `is_leaf` is in the `taxon` table. This table can be used to search for nodes at a known rank depth, or which has a certain ancestor. Relative queries are not available from this table

Taxon table

Column	Description
<code>Taxon_id</code>	Primary Key
<code>parent_taxon_id</code>	Reference to the parent node.
<code>rank</code>	NCBI rank, genus, species, etc.
<code>division_id</code>	NCBI division
<code>genetic_code_id</code>	Codon table
<code>mito_genetic_code</code>	Codon table of mitochondria
<code>embl_code</code>	EMBL 2 letter code.
<code>is_leaf</code>	If this is a leaf node
<code>is_root</code>	If this is the top of the tree
<code>distance_to_root</code>	Number of elements to the root
<code>distance_to_leaf</code>	...
<code>kingdom</code>	
<code>phylum</code>	
...	...
<code>genus</code>	
<code>species</code>	
<code>scientific_name</code>	Common name
<code>other_names</code>	A hash of other names

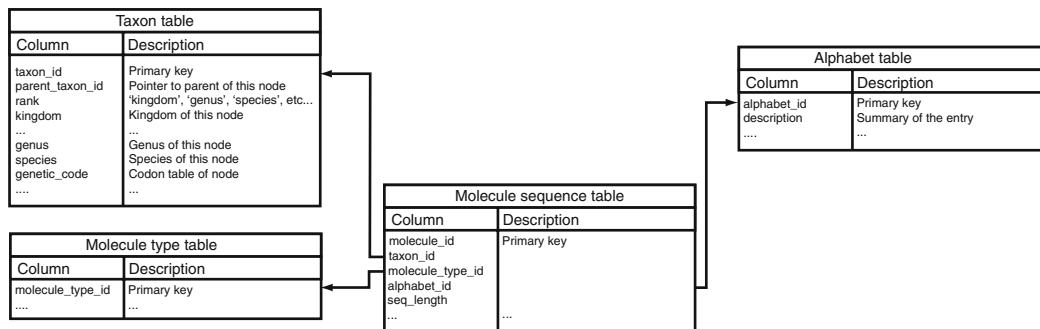


Fig. 23.3. An example of the topological closure data stored for a tree. All paths in the original tree are enumerated in the topological closure.

In addition to hierarchies, graphs and networks are common structures in biological systems including protein–protein interaction networks (19–22), biochemical pathways (23), and others. However, the techniques outlined for trees and directed acyclic graphs are no longer appropriate for graphs.

Answering any more than very basic graph queries is hard in relational databases. One approach is to use a specialized application, which is designed for graph and network representation and queries. In the Bioverse database we have implemented such an application at the interface between the database and the application. It supports a simple breadth first search, as well as searches for graph motifs. While this makes certain tasks simple, specialized applications will not alleviate all problems. Finding the shortest path between two nodes is an expensive operation that may be better done outside of the database. The Bioverse does not have a pressing need for these types of expensive queries, so they have not yet been implemented.

4.3. Functional Data: Affinity Grouping Example

Combining or comparing data from different rows of the same table is expensive in relational databases. This is one reason why the choice of granularity is so important for the fact tables. As previously discussed if the chosen resolution is too fine, queries will require comparing rows. In some cases, there is no alternative because the queries are at many scales. One approach for storing data at different resolutions is affinity grouping, which enables the analysis of individual records as well as certain groups of records.

In the Bioverse, we have both functional annotations from trusted sources and predicted functional annotations.

Table 23.2
A simple record of GO annotations

GO annotation table

Column	Description
<code>taxon_id</code>	Foreign Key to the Taxon table
<code>molecule_id</code>	The identifier of the molecule
<code>go_id</code>	Foreign Key to the Gene Ontology table
<code>confidence</code>	Confidence assigned to the annotation

A simple table for the GO annotations stored in the Bioverse is shown in **Table 23.2**. While this table provides information on protein annotations, it is a limited view of the data. Since each protein annotation is a unique fact, it is not easy to see which annotations are related or which are the most specific annotations in the DAG of GO features. To provide access to this data we have calculated a second table that gives information about which combinations of GO annotations are seen together. An added benefit to calculating a priori this data is having access to summary statistics when making queries of the affinity-grouped tables. It is desirable to have information on frequencies, which can be used to estimate the statistical significance of an observation. This affinity-grouping table is shown in **Table 23.3**.

4.3.1. Audit Dimensions

Data that are unrelated to the science of the measurements, but remains relevant, can be annotated in audit dimensions. These tables record events that are significant, interesting, or possible errors. An audit dimension can provide an overview of the fact table or relevant meta data about the measurement. For computed values an audit dimension may be used to record the version information of software or runtime parameters used to generate the fact.

For numeric columns audit dimensions may be used to mark data that are missing, provide meanings or justifications for missing data, or to identify interesting data. A flag that marks all data which is more than two standard deviations away from the mean allows the identification of problematic or interesting data cases.

Table 23.3

The `pair_count` column gives the total number of times these two annotations are seen together, and the `go_1_count` and `go_2_count` give how often annotation occurs independently. This summary table is very useful for exploring annotation pairs and other relationships. The `is_related` column stores whether one annotation of the two is an ancestor of the other in the DAG, in which case the relationship represents the frequency of different parts of the subtree of that node

GO and GO affinity grouping table

Column	Description
<code>taxon_id</code>	Foreign Key to the Taxon table
<code>go_id_1</code>	Foreign Key to the Gene Ontology table
<code>go_id_2</code>	Foreign Key to the Gene Ontology table
<code>pair_count</code>	number of co-occurrences of <code>go_id_1</code> and <code>go_id_2</code> in molecules
<code>go_1_count</code>	occurrences of <code>go_id_1</code> in organism
<code>go_2_count</code>	occurrences of <code>go_id_2</code> in organism
<code>is_related</code>	Whether one of the go identifiers is a parent of the other
<code>distance</code>	The distance between the entries in the GO tree

5. Bioverse Extraction, Transformation, and Loading

For the Bioverse we have developed in-house data-processing tools, which move the data through a pipeline architecture, processing and running algorithms at each stage.

5.1. ETL Discussion

There are a few guiding principles that influence the design of our tools:

1. Process data in large blocks. Since frequent disk access is expensive, the data are read and processed in blocks that fit into memory.
2. Filter early and often. We filter data as quickly as possible. After opening a block of data, the first steps that are taken are to eliminate any data that are not of interest, in order to minimize later workloads.

3. Expect failures and corrupted, ambiguous, and inconsistent data. All of the operations performed on the data are recorded. Any corrupted, ambiguous, and inconsistent data that are encountered will need to be dealt with robustly.
4. Log and audit every block, and store all bad data in a way that allows for restarting for failed blocks.

The ETL tool when constructed allows for short programs to be written, which will perform on disk translation of the downloaded and generated data. This data should leave in a state that is amenable to being uploaded to the database.

5.2. Bioverse Extraction

Bioverse data is extracted from a wide variety of data formats. For each data format a parser is available, which will produce all data in the file without any attempt at filtering or validation. Parsers catch formatting errors quite often, and log the failure of the data read to the database. Additionally any data that was expected or is optional that was not present is marked as a failure or a missing value.

5.3. Bioverse Transformation

The data generated from the extraction stage are transformed to prepare it for the Bioverse staging area. Blast hits against databases are checked to make sure that the matching molecule is actually in the database we have loaded. Many types of data will have unique values generated so they can be identified unambiguously later. Errors encountered at this stage are recorded in the database, and the data at fault is not processed further. If the offending program is restarted, it will not duplicate data, since everything takes place in a single database transaction and the failure aborts the transaction.

5.4. Bioverse Loading

The loader code works with large chunks of data, each of which is associated with a block of records. These data are copied into the database. In the event of failure, all of the records in the block are marked as failed in the database, and none are loaded. Again, this consistency ensures that the operation can be repeated when the problems with the data have been resolved, or the underlying data have been regenerated, and there will be no duplication of data.

6. Conclusion

We have addressed the complex problem of creating a database and representation to store a wide variety of biological information.

We expect that, in conjunction with the Pipeline, API, and web application, the database model we have created will be useful to help generate hypotheses and solve problems for biological research.

In addition to introducing the basic data warehousing concepts, we have provided a general strategy for the management and integration of biological data. Specific examples demonstrate how the Bioverse is constructed and how large volumes of data are loaded, stored, and analyzed within the data warehouse.

References

- Hwang, D., Rust, A. G. G., Ramsey, S., Smith, J. J. J., Leslie, D. M. M., Weston, A. D. D., et al. A data integration methodology for systems biology. *Proc Natl Acad Sci U S A*, 2005, 102(48):17296–17301.
- Hwang, D., Smith, J. J., Leslie, D. M., Weston, A. D., Rust, A. G., Ramsey, S., et al. A data integration methodology for systems biology: Experimental verification. *Proc Natl Acad Sci U S A* 2005,102(48); 17302–17307.
- McDermott, J., Bumgarner, R., & Samudrala, R. Functional annotation from predicted protein interaction networks. *Bioinformatics* 2005,21(15):3217–3226.
- Chen, N., Harris, T. W., Antoshechkin, I., Bastiani, C., Bieri, T., Blasiar, D., et al. (2005). WormBase: A comprehensive data resource for caenorhabditis biology and genomics. *Nucleic Acids Res* 2005,33(Supplement 1):D383.
- Haft, D. H., Selengut, J. D., & White, O. The TIGRFAMs database of protein families. *Nucleic Acids Res* 2003,31(1): 371–373.
- Madera, M., Vogel, C., Kummerfeld, S. K., Chothia, C., & Gough, J. The SUPERFAMILY database in 2004: Additions and improvements. *Nucleic Acids Res* 2004,32: D235–D239.
- Kanehisa, M., Goto, S., Kawashima, S., Okuno, Y., & Hattori, M.. The KEGG resource for deciphering the genome. *Nucleic Acids Res* 2004, 32(Database issue)
- Wilkinson, M. D., & Links, M. BioMOBY: An open source biological web services proposal. *Brief Bioinform* 2002,3(4):331–341.
- Carrere, S., & Gouzy, J. REMORA: A pilot in the ocean of BioMoby web-services. *Bioinformatics* 2006,22(7): 900–901.
- Shah A.R, Singhal M., Klicker K. R., Stephan E. G., Wiley H. S., & Waters K. M. Enabling high-throughput data management for systems biology: The Bioinformatics Resource Manager. *Bioinformatics* 2007,23(7):906–909.
- McDermott, J., Guerin, M., Frazier, Z., Chang, A., & Samudrala, R. BIOVERSE: Enhancements to the framework for structural, functional, and contextual annotations of proteins and proteome. *Nucleic Acids Res* 2005,33:W324–W325.
- Kimball, R., Ross, M. The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling 2002, Wiley, New York, NY.
- Codd, E. F. A relational model of data for large shared data banks. *Communications of the ACM* 1970,13(6):377–387.
- Hucka, M., Finney, A., Sauro, H. M., Bolouri, H., Doyle, J. C., Kitano, H., et al. The systems biology markup language (SBML): A medium for representation and exchange of biochemical network models. *Bioinformatics* 2003,19(4): 524–531.
- Hermjakob, H., Montecchi-Palazzi, L., Bader, G., Wojcik, J., Salwinski, L., Ceol, A., et al. The HUPO PSI's molecular interaction format – a community standard for the representation of protein interaction data. *Nat Biotechnol* 2004,22(2):177–183.
- Ashburner, M., Ball, C. A., Blake, J. A., Botstein, D., Butler, H., Cherry, J. M., et al. Gene ontology: Tool for the unification of biology. The gene ontology consortium. *Nat Genet* 2000,25(1):25–29.
- Lo Conte, L., Ailey, B., Hubbard, T. J., Brenner, S. E., Murzin, A. G., & Chothia, C. SCOP: A structural classification of proteins database. *Nucleic Acids Res* 2000, 28(1);257–259.
- Benson D.A., Karsch-Mizrachi I., Lipman D.J., Ostell J., Wheeler D.L. GenBank: Update. *Nucleic Acids Res* 2004,32: D23–D26.

19. Bader, G. D., Betel, D., & Hogue, C. W. BIND: The biomolecular interaction network database. *Nucleic Acids Res* 2003,31(1):248–250.
20. Breitkreutz, B. J., Stark, C., & Tyers, M. The GRID: The general repository for interaction datasets. *Genome Biol* 2003 4(3):R23.
21. Chatr-aryamontri, A., Ceol, A., Palazzi, L. M., Nardelli, G., Schneider, M. V., Castagnoli, L., et al. MINT: The molecular INTeraction database. *Nucleic Acids Res* 2007,35:D572–D574.
22. Xenarios, I., Rice, D. W., Salwinski, L., Baron, M. K., Marcotte, E. M., & Eisenberg, D. DIP: The database of interacting proteins. *Nucleic Acids Res* 2000,28(1): 289–291.
23. Kanehisa, M., Goto, S., Kawashima, S., Okuno, Y., & Hattori, M. (2004). The KEGG resource for deciphering the genome. *Nucleic Acids Res* 32(Database issue).

Chapter 24

Biological Network Inference and Analysis Using SEBINI and CABIN

Ronald Taylor and Mudita Singhal

Abstract

Attaining a detailed understanding of the various biological networks in an organism lies at the core of the emerging discipline of systems biology. A precise description of the *relationships* formed between genes, mRNA molecules, and proteins is a necessary step toward a complete description of the dynamic behavior of an organism at the cellular level, and toward intelligent, efficient, and directed modification of an organism. The importance of understanding such regulatory, signaling, and interaction networks has fueled the development of numerous in silico inference algorithms, as well as new experimental techniques and a growing collection of public databases. The Software Environment for BIological Network Inference (SEBINI) has been created to provide an interactive environment for the deployment, evaluation, and improvement of algorithms used to reconstruct the structure of biological regulatory and interaction networks. SEBINI can be used to analyze high-throughput gene expression, protein abundance, or protein activation data via a suite of state-of-the-art network inference algorithms. It also allows algorithm developers to compare and train network inference methods on artificial networks and simulated gene expression perturbation data. SEBINI can therefore be used by software developers wishing to evaluate, refine, or combine inference techniques, as well as by bioinformaticians analyzing experimental data. Networks inferred from the SEBINI software platform can be further analyzed using the Collective Analysis of Biological Interaction Networks (CABIN) tool, which is an exploratory data analysis software that enables integration and analysis of protein–protein interaction and gene-to-gene regulatory evidence obtained from multiple sources. The collection of edges in a public database, along with the confidence held in each edge (if available), can be fed into CABIN as one “evidence network,” using the Cytoscape SIF file format. Using CABIN, one may increase the confidence in individual edges in a network inferred by an algorithm in SEBINI, as well as extend such a network by combining it with species-specific or generic information, e.g., known protein–protein interactions or target genes identified for known transcription factors. Thus, the combined SEBINI–CABIN toolkit aids in the more accurate reconstruction of biological networks, with less effort, in less time.

A demonstration web site for SEBINI can be accessed from <https://www.emsl.pnl.gov/SEBINI/RootServlet>. Source code and PostgreSQL database schema are available under open source license. Contact: ronald.taylor@pnl.gov. For commercial use, some algorithms included in SEBINI require licensing from the original developers. CABIN can be downloaded from <http://www.sysbio.org/datare-sources/cabin.stm>. Contact: mudita.singhal@pnl.gov.

Key words: Network inference, transcriptional regulatory networks, signal transduction networks, protein-protein interaction networks, and exploratory data analysis.

1. Introduction

Reconstruction of regulatory and signaling networks is a critical task in systems biology (1–9). The “Software Environment for BIological Network Inference” (SEBINI) (10) and the “Collective Analysis of Biological Networks” (CABIN) (11) analysis frameworks, developed at the U.S. Department of Energy’s Pacific Northwest National Laboratory, aid in the reconstruction of the structure of such mRNA and protein networks.

High-throughput molecular biology experiments are now producing mRNA expression data in quantities large enough for researchers to attempt to reconstruct the structure of gene transcription networks based primarily on state correlation measurements (12–58). In its simplest form, such inference from state correlation could run as follows: if the expression level (state) of gene A is always high across a (large) set of experiments when the mRNA expression level of gene B is low, and low when the level of gene B is high, then we could conclude that there is strong evidence that gene A directly regulates gene B, i.e., there is a regulatory edge directed from A to B, and that gene A represses gene B. Of course, such inference is rarely so simple (59). In the absence of other information, we could conclude with equal probability that gene B represses gene A. Determination of the causal direction is an important subtask in the determination of regulatory network structure. With the possibility of multiple regulators affecting a target gene and each source gene affecting a large number of targets, as well as imperfect experimental data, computational scientists, drawing on knowledge from several fields (information theory – e.g., mutual information (60, 61), classical statistics – e.g., Pearson product-moment correlation coefficient (62), probabilistic graphical models – e.g., Bayesian network structure learning (63–68), data mining – e.g., association rule mining (69)) have built very sophisticated inference algorithms to detect regulatory edges from even partial correlations in state.

Large-scale protein activation and protein abundance measurements are soon to follow in the steps of mRNA microarray experiments, allowing similar inference based on state correlation for protein signaling networks and regulatory networks.. Further, metabolite networks can be inferred by algorithms that use correlations in large-scale measurements of metabolite levels (D. Wishart, personal communication, 2006) Thus, while reconstruction of metabolic pathways and networks will not be

discussed in this chapter, the algorithms in SEBINI may have value in that research area as well. Protein–protein interaction networks are also starting to be inferred from high-throughput data, such as from sets of mass spectrometry bait-prey experiments (70). Inference of protein interactions from bait-prey data sets is *not* based on state correlation in the same sense that correlation is used in the analysis of microarray experiments to infer transcriptional regulatory edges. But evidence from a set of mass spectrometry experiments can be tied to the set of proteins, uploaded into the SEBINI platform, and passed to an algorithm, the Bayesian Estimator of Protein–Protein Association Probabilities (BEPro) (71–73), which determines the degree of association between the bait and prey proteins based on a Bayesian analysis across the entire set of experiments, and thus determines which baits and prey truly interact, eliminating false positives. In this manner we may construct a set of interaction edges that can be stored as a protein–protein interaction network which fits comfortably within the SEBINI framework.

In a graphical representation of a transcriptional regulatory network, the genes are the nodes and the edges between the nodes are directed. A set of microarray runs measuring mRNA expression is used as the input data set to the inference algorithm that infers the set of regulatory edges. A protein–protein interaction (PPI) network has proteins as nodes with undirected edges (showing interaction, but no cause-and-effect). For example, a set of bait-prey experiments, as mentioned above, could be used as input to an appropriate analysis algorithm. Signal transduction networks also have proteins as nodes, but the edges are directed, to show causal influence on target protein activation state. The input to an algorithm inferring a signal transduction network would be a set of experiments measuring protein activation and inactivation (e.g., phosphorylation / dephosphorylation) across a set of proteins being scanned for involvement in the signaling network.

SEBINI provides an open source software platform that allows the use of many inference algorithms on many types of data sets. Experimental data in several formats can be uploaded into SEBINI for analysis by the algorithms incorporated into SEBINI's toolkit. Also, artificial data sets of different types can be created dynamically within SEBINI and used to test a growing collection of inference algorithms. Thus, the SEBINI interactive environment helps researchers (1) easily apply state-of-the-art algorithms to infer a network from experimentally generated high-throughput data and (2) evaluate and refine new algorithms for the inference of biological regulatory and signaling network structure using common data sets.

SEBINI is a framework whose database can store networks of many types, and any algorithm that works on pieces of evidence attached to nodes in a potential network, or on evidence attached

to the set of nodes as a whole, can be fitted into the SEBINI platform. The majority of algorithms in SEBINI are directed toward searching for networks of causal influence, where the state of one node affects the state of another node; the best-known example of such being transcriptional regulatory networks derived from microarray experiments that measure mRNA levels. However, the BEPro algorithm described above has also been added into SEBINI to look at evidence from protein bait-prey experiments. Such evidence allows BEPro to infer interactions (but not regulatory influence) between the bait and the prey proteins, and thereby derive from the experimental results a single protein-protein interaction network. In summary, the SEBINI platform is useful not only in inferring the topology of any network where the change in state of one node can affect the state of other nodes (regulatory networks) but also, more generally, in inferring the topology of any (interaction) network where the evidence for the existence of edges can be uploaded and stored in SEBINI's central database, where the uploaded data are tied to a network object for later use by an appropriate algorithm.

Focusing on the analysis of microarray experiments, we can note that clustering or some form of statistical classification has typically been employed to organize the high-throughput mRNA expression values derived from microarray experiments (38, 74–78). The question then arises: how can the clustering or classification results be connected to the underlying biology? Such results can be useful for pattern classification, for example, to classify subtypes of cancer or to predict differential responses to a drug (pharmacogenomics). But to understand the relationships between the genes, i.e., to more precisely define the influence of each gene on the others, the scientist usually attempts to reconstruct the transcriptional regulatory network. This can be done by using background literature or information in public databases, combined with the clustering results. It can also be done by the application of an algorithm, often based on a probabilistic graphical model or on an information-theoretic metric, to try to infer the regulatory network from the raw, high-throughput data. This is where the algorithms in SEBINI come into play.

What does the graph of such an inferred regulatory network give you? The structure of a transcriptional regulatory network can be described as a “wiring diagram,” a directed graph whose edges show regulatory influences. Such a diagram describes all the direct and indirect influences on the expression of a gene, and shows what (the product of) a gene can affect (5, 79). Ideally, one would like to have not just such a diagram but also the set of equations governing the behavior (state) of all the nodes of such a network over time. This would allow predictions to be made of the precise temporal behavior of all the (dependent) variables in the system. However, the diagram itself is an extremely important starting

point for network analysis. It constrains the possibilities and shows what can affect different aspects of the system. It is the blueprint – the starting point for later investigations for dynamic behavior. Also, the identification of the regulatory connections, or edges, in a network in and of itself answers important questions, and provides a guide to several areas of research. Some examples for transcriptional regulatory networks follow.

1.1. Gene Function Identification

Suppose gene A is known to be involved in a signal transduction or metabolic pathway and gene B, through the network regulatory edge diagram, is seen to directly influence gene A. Then the researcher can assign gene B a possible role – some involvement in that pathway. Also, gene B becomes a candidate for targeted experiments to study the pathway. (Conversely, if gene A is seen to be the direct source of a regulatory influence on a previously unsuspected gene C, then gene C also becomes a possibility for inclusion into the pathway.)

1.2. Identification of Upstream and Downstream Genes

The network diagram shows what genes lie upstream and downstream of each other, in terms of any possible regulatory effect. For any gene represented in the network, all possible sources of regulatory influence on that gene from other genes in the network, both direct and indirect, are explicit in the diagram. Conversely, the direct or indirect affect of a given gene can be found by tracing all possible downstream paths from the gene. Hence, when the transcriptional level of a gene is experimentally altered, the researcher will know how far the effect of such a change may propagate through the biological network under study, and where the effect may be strongest or weakest, based on the length of the path from the altered gene to the downstream gene and on the number of other direct and indirect influences on the downstream gene.

1.3. Target Identification

A researcher might want to modulate a cellular subsystem, such as a signaling transduction pathway, with minimal effect on the other subsystems in the cell. Knowledge of global transcriptional regulatory network provides guidance to the researcher in the selection of genes whose alteration of expression will have the least influence, direct or indirect, on genes lying outside of the pathway.

1.4. Elimination of Irrelevant Genes: Pruning the Putative Network

If there is prior belief that a subset of genes is possibly involved with each other, the availability of the network diagram will allow some genes to be eliminated from consideration. If a gene G cannot directly influence at least one other gene in the set and gene G is not a target of any regulatory connection from any other gene in the set, then gene G can be dropped from membership in the proposed subsystem.

1.5. Speed of Response The network diagram allows calculation of the average path length between nodes in the full network or in any subgraph. The length may give some indication of the relative speed of a subnetwork in its reaction to a change in the cellular environment.

1.6. Identification of Control Genes The network diagram allows answering questions such as: Are there master control genes in the network? Are there genes that serve as dominant sources of regulatory connections and, thus, act as “hubs” for regulating many other genes? If so, research can be focused on those genes whose state will directly control or affect the state of much of the network.

The inferred networks obtained from SEBINI can be validated in the CABIN software package (11), that is, interactions or regulatory connections can be verified by combining evidence from public databases such as the Database of Interacting Proteins (DIP) (80–82) and the Biomolecular Interaction Database (BIND) (83); and from computational methods such as phylogenetic profiling, Rosetta Stone, gene neighborhood, homology information (55, 84). Such verification, of course, increases our confidence in a given edge. Networks can also be annotated or extended (edges added) within CABIN. CABIN has been developed as a plug-in to Cytoscape (85), which is an open source network visualization and analysis tool. CABIN is invoked from SEBINI from a button on a web page that launches Cytoscape via Java Web Start, with the appropriate inferred network from SEBINI’s database automatically passed in for visualization and further analysis within Cytoscape and CABIN. Once the Cytoscape window comes up, CABIN can then be selected from the plug-in menu.

CABIN facilitates integrating the evidence of interaction data from multiple sources by the use of interactive visual interfaces. Multiple coordinated views within CABIN foster exploratory data analysis, allowing weighting and filtering of data sources (via slider controls and other easy-to-use controls) to create a final combined network. Use of CABIN permits high-quality human judgment on the integration of complex data sets having different levels of certainty, with limited investment of the user’s time.

2. Software Architecture

SEBINI uses a standard three-tier architecture: (1) a web-based client user interface, (2) an application logic middle tier consisting of a suite of Java servlets, and (3) a relational database storing the data required by the middle tier. Inferred networks, as well as the raw data, processed data (processing may mean binning of

microarray data, or peptide-to-protein collapse for mass spec bait-prey data), and algorithm parameter selections used to generate the networks are permanently stored in the database for visualization, topological and statistical analysis, and for later export in a human-readable or program-specific format. Inference algorithms and discretization (binning) or other data-processing algorithms can be any sort of executable program; a Java handler class is added for each new algorithm to handle communication between the invocation web page, the database, and the algorithm. Security is implemented on a project basis, with one owner and possibly multiple users per project. Upon password-protected login to the SEBINI web site, the user is assigned a 32 digit hex digit JSessionID, which is checked before display of every web page.

Major design issues included (1) the interface for user navigation among possibly huge data sets, allowing easy drill down from a network set to a specific network to a specific node or edge; and (2) producing an efficient, understandable mapping from the inferred networks and inferred edges back to the corresponding original expression or abundance data. Note that we have one-to-many relationships from a raw uploaded data set to a processed data set, as well as a one-to-many relationship between a processed data set and the inferred network and inferred edges created by the selected inference algorithm, operating on the selected processed data set. Records for each of these data types are permanently stored and connected to the appropriate records of the other data types. Note that the processed data sets are permanently stored, in addition to the raw data sets and the inferred networks. This is important for efficiency (reuse of processed data in another inference run), transparency, and verification of results. Other design decisions: all Java inter-servlet communication is routed through a CentralControl servlet, for a clear flow of control and monitoring choke point.

Each inference algorithm and each (pre)processing algorithm (preprocessing in the sense that the data are being prepared for input into an inference algorithm) is invoked in a separate Java thread that performs job posting to the database, thus allowing dynamic monitoring of job progress by the user. Jobs are timed to the millisecond, allowing comparison between algorithms of relative speed versus relative power. While an algorithm is running, any web page listing for that newly created processed data set or inferred network will say “under construction,” with the processed data set or inferred network only becomes available upon completion of the algorithm run. At any time, the user may check on the progress of a run by bringing up the job page for the processed data set or inferred network and reading the job postings that have been stored so far in the database by the processing program or by the Java wrapper around the selected inference algorithm. The computational and RAM memory requirements for any given

inference run will vary dramatically based on the number of nodes involved, the algorithm used and how efficiently it was written, the parameter settings used for a particular algorithm, and the number of data points (gene expression, protein abundance, or protein activation values) uploaded for each node. The time required may be a few seconds on desktop computer for a mutual information-based algorithm on a network of 20 nodes, given 100 gene expression values for each node, from 100 microarray experiments. Or, if we use a Bayesian network type of algorithm on a set of 1000 nodes using data from several hundred array experiments, the user may set the parameters so that algorithm will run for days, in order to return a high-quality inferred network.

Each node and each edge in an inferred network can be viewed on its own web page. Each edge carries with it the raw and processed state values for its two nodes across the entire set of experiments. These values can be viewed by the user, or later output with the inferred network topology, for use in fitting the equations used for dynamic modeling to the experimental data. Each inferred network can be viewed as a table of nodes and edges, or visualized as a graph within Cytoscape and further analyzed or annotated with CABIN. The edges of an inferred network can also be exported in Cytoscape SIF file format.

The user may delete an entire project, an uploaded (raw) data set, a processed data set, or an inferred network. Everything previously created downstream of a deleted set is also deleted. For example, all inferred networks created from a given processed data set are deleted, if the processed data set itself is deleted.

To support response speed in handling potentially huge data sets, and thus potentially some very large tables in the PostgreSQL central database, retrieval of records is almost always based on the primary key of the relevant table. Secondary indices are sometimes used, but even those are kept to a minimum, since each index added has to be updated when new data is uploaded into the system, thus possibly slowing response on the web site. Search and retrieval on unindexed fields is extremely rare and, when done, confined to tables that will stay relatively small over time. Most importantly, each project is given its own separate PostgreSQL database. Thus, a user may at any time start a new project, with a duplicate set of empty tables, and continue from there. All (old) projects remain fully accessible, but the new project allows the user to begin completely anew.

SEBINI was initially implemented on a Dell desktop running Red Hat Linux, using Java ver. 1.5, PostgreSQL ver. 7.4 (86), and Apache Tomcat 4.1 (87). Communication between the Java programs and the PostgreSQL database is done via JDBC. The Jakarta Commons file upload and IO Java libraries are used (<http://jakarta.apache.org/commons>). SEBINI has also been installed on a Windows 2003 computer and on a Mac running Mac OS X ver. 10.4.8. Machine-specific parameters are stored in an easily

changed text file read by use of the Java util.Properties class. Mathworks' MATLAB (88) is required for some of the inference algorithms, while R (89) is required for others. Note that although the user interface is web-based, SEBINI can run completely self-contained (with CABIN), without any Internet connection, by setting the host machine to "localhost" in the properties text file. A block diagram of the system is shown in Fig. 24.1.

Like SEBINI, CABIN is written in Java. It makes use of publicly available Java libraries such as Colt (<http://dsd.lbl.gov/~hoschek/colt>), JFreeChart (<http://www.jfree.org/jfreechart>), jMatrixView (<http://jmatrixview.sourceforge.net>), and BiSlider (<https://bislider.dev.java.net>) to provide rich visualization and an effective user interface. Once imported into CABIN, evidence networks are stored in a matrix model that keeps a list of the networks and their interactions. This model, provided by the high-performance Colt library, is backed by an optimized two-dimensional sparse matrix, which contains the confidence values of each interaction (row) of each network (column). These data values are visualized in multiple views as scatter plots (JFreeChart), as a heat map matrix representation (jMatrixView), and as Cytoscape networks. Each view references the matrix model and observes any changes in the model, allowing the views to update themselves when networks are imported, removed, or updated. Additionally, a view selection controller serves as an intermediary

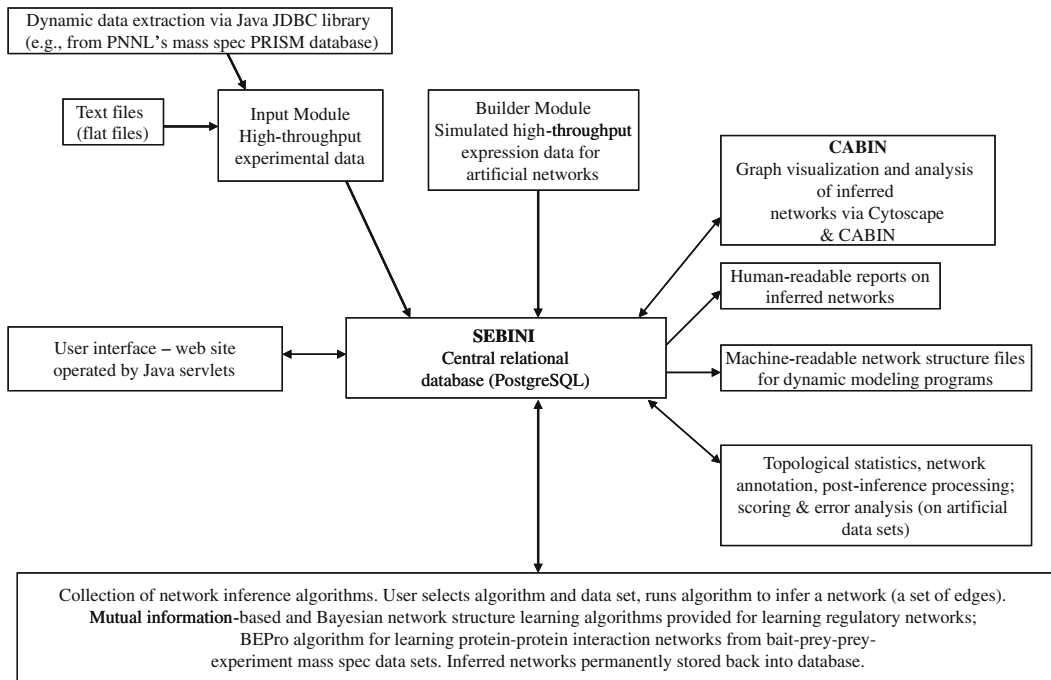


Fig. 24.1. A block diagram of the SEBINI-CABIN system.

that notifies each registered view of any data selection events. Data values can be manipulated further using the histogram range slider interface facilitated by BiSlider and JFreeChart libraries.

3. Capabilities

The capabilities provided within SEBINI include:

1. **Data Import** – Upload of several types of experimental data for input into selected processing and network inference algorithms.
2. **Network Inference** – There are several choices of inference methods. Currently, SEBINI has algorithms from classical statistics (e.g., Pearson correlation), static and dynamic Bayesian network structure learning algorithms (e.g., the BANJO toolkit at Duke University (90, 91)), and information theory (mutual information-based; e.g., basic no-frills mutual information (60), the ARACNE algorithm at Columbia University (92–95), and the CLR algorithm at Boston University (96, 97)). Also, The Bayesian Estimator of Protein-Protein Association Probabilities (BEPro) algorithm (70–72) has been added for inference of protein–protein interaction networks from bait-prey experiments.
3. **Network Storage and Analysis** – Inference networks can be permanently stored and further analyzed. For each network, the user can view a summary page; a topological characteristics and statistics page; a graph visualization using Cytoscape (85), invoked via Java Web Start; summary pages for each node and edge showing the raw (uploaded) and processed node states; and job pages that record how the processing and inference tasks proceeded.
4. **Algorithm Comparison** – Direct comparison of network inference methods on common synthetic or experimental data sets.
5. **Experimental Planning Tool** – Using simulated data sets, SEBINI can report, using different inference methods, on what can be reconstructed of the topology (regulatory connections) of a network from the inference results on such an artificial data set of a given size. Thus, SEBINI may be useful in making a rough estimate of the number of experiments (data points) required.
6. **Network Export** – Output of inferred network structures as input to other tools such as CABIN or to tools such as the Systems Biology Workbench (98) (e.g., for dynamical modeling), and export of human-readable reports on the networks, with various topological characteristics noted.

The functionalities provided within CABIN for visual analysis of multiple interaction networks include:

7. **Network Import** – Each evidence network (e.g., a network of protein–protein interactions from the BIND database) can be imported into CABIN using the Cytoscape SIF file format. In its simplest form the first and third columns in the SIF file represent proteins and the middle column represents the value for the interaction. The evidence networks can be assigned a reliability score, or weight, based on the confidence in the evidence source. CABIN has provisions to assign custom values to missing evidences for interactions. This missing value can be set to a value between 0 and 1, to the median value for that evidence network, or we can ignore that evidence source for the interaction.
8. **Exploratory Analysis** – Once loaded, the CABIN environment provides interactive visual interfaces to carry out exploratory analysis of the networks as shown in Fig. 24.2. The Weighted Scaling View window pane displays a point

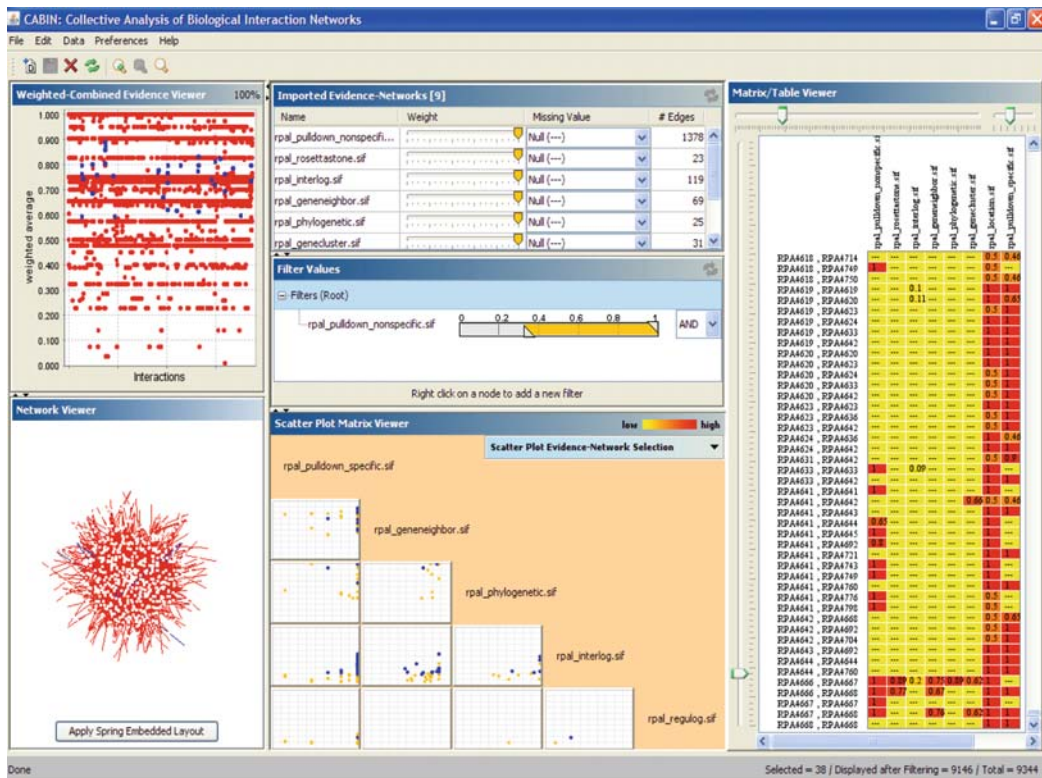


Fig. 24.2. Use of the CABIN software to validate experimental interactions for *Rhodospseudomonas palustris* obtained using tandem affinity purification technique bait-prey experiments.

for each interaction in any of the evidence networks, with the value for that interaction represented by the average of the weighted sums of confidence values in all the networks. The Cytoscape Network View window pane provides a network/graph visualization of the networks loaded into CABIN. The Matrix/Table View provides a heat map representation of confidence values of all sources for each interaction, along with the options to sort interactions based on their values. A Scatter Plot Matrix view is also provided, which shows the scatter plots of interactions in all evidence networks with respect to the others. This view facilitates estimating the weight assignment for determining the confidence in the predictions for that source. The four views are coordinated with respect to each other, so selections made in one view are reflected as selections in the other two views.

9. **Construction of Network Subsets** – Filters can be added to select an edge confidence (edge score) cutoff for the edges in an evidence network. If the value for an edge falls below the given cutoff, the edge is omitted from display and any further use. The selection of the filter cutoff value is performed by an easy-to-use slider control while viewing a histogram graph that shows the distribution of the edge values in each network. Edges displayed can also be restricted based on an OR relation or an AND relation amongst the networks; e.g., we may restrict the combined network displayed (and possibly exported later) to those edges that appear in the inferred network passed in from SEBINI and that also appear in either the evidence network from BIND or the evidence network from DIP (SEBINI AND (BIND OR DIP)). Once the filters are set and the update button is clicked, the views are updated based on interactions (edges) that pass the filters. The filtered set of edges can be saved as a new network within CABIN and assigned a confidence of its own.
10. **Find/Search Functionality** – CABIN has regular expression based find/search functionality, which allows the user to look for a specific interaction or select all interactions involving a particular molecule of interest.
11. **Export** – At any stage of the analysis process, the selected interactions can be saved to a local file for later use.

Additionally, to support algorithm developers, SEBINI also allows:

12. **Artificial Data Sets** – Topologies (99), perturbations, and node input function definitions can be dynamically created and stored. Boolean value expression sets are currently

created. The more sophisticated Java-based SynTreN network generator software is in the process of being added as an additional SEBINI module (100).

13. **Step-wise Refinement of Inference Methods** – Scoring measures (recall, precision, *F*-measure) are used to measure performance against the simulated networks with known structure. Thus, supervised training of an inference algorithm is possible on a set of known (simulated) data sets.
14. **Well-defined Expansion** – Addition of each new inference algorithm, (pre) processing technique, import uploader or export method is coded as a new Java module which fits easily into the already existing framework.
15. **Scoring Distributions** – As a guide for the interpretation of the scores produced by an inference technique, SEBINI can produce scoring distributions for a given inference method against known networks. Such distributions can then be used to determine appropriate cutoff scores for determination of the existence of a regulatory influence (an edge) to a target gene.

4. Analysis Flow of Control

In this section, we walk through the steps one would use in performing an analysis in SEBINI on experimental data, and in using SEBINI to evaluate an inference algorithm on a synthetic data set. We will then examine the experimental analysis of a protein bait-prey experiment set more closely using both SEBINI and CABIN, as representative test case.

The flow of control on the SEBINI web site for the analysis of experimental data goes as follows:

- 1) Log into SEBINI. If you have not used the SEBINI site before, you are directed to a registration page to enter your chosen user name and password. Once that is done, you may log in and create a project.
- 2) Select a project, or create a new project. Typically, a user will log in and select one of the projects to which he or she has access – a list of such is presented on a web page. All work is done within a context of a project, and a project must be selected before doing anything else. All data sets, all nodes, all edges, all networks belong to a particular project and can be accessed from that project only.
- 3) Create a network container for the data set you are uploading, for the experimental network you are trying to reconstruct. This is a simple task – one enters a name, a short optional description, and selects an uploader method for future use.

- 4) Upload the experimental mRNA expression or protein abundance data file. Using the uploader method chosen in the previous step, a web page is presented, which allows the user to upload a file or files from his/her local computer. After a file path is selected in a Browse box and the user clicks on “submit”, the data set in this file is parsed, according to the upload method selected, and permanently stored in the database.
- 5) Select a processing or binning algorithm and run it on the data. The uploaded raw data set can be processed or binned in many ways. For example, for a set of microarray experiments, each gene expression value could be binned (discretized) into two-state Boolean values (on/off), based on a cutoff value set at the halfway point of the min and max of the value range across the set of (already normalized) expression values, across the set of experiments. That is a simple example. Discretization can be done in very complex ways, and the results one obtains for the inferred network can depend strongly upon your binning technique. For data sets being used as input into inference algorithms that do not require binning or other processing (or for data sets that were already binned before being uploaded into SEBINI), the “pass-through” menu option can be used. This option will create a processed data set that is a duplicate of the raw data set. In any case, a processed data set must be created, because only a processed data set is allowed as input into an inference algorithm.
- 6) Select an inference algorithm, select the values to use for its parameters, and infer a network. This step can be considered the “heart” of SEBINI, since it is where we actually infer a network and store it back into the database. Using a set of web pages, the user selects a processed data set, selects the inference algorithm to employ on that data set, enters values into fields for all the parameters that the algorithm needs (usually, default values are given on-screen), and then clicks on a “submit” button that launches the algorithm, via its Java wrapper program, in a separate Java thread. When the algorithm finishes, the Java wrapper will parse the algorithm’s output and store the inferred edges in the database as the newly inferred network. The edge scores and any other useful information produced by that particular algorithm will also be stored, attached to the inferred edges, or to the parent inferred network record. The inferred network record is the “parent” of its child nodes and edge records, and contains pointers – unique ids – to them. Likewise, the node and edge records have a field containing the unique id of the parent network record. Thus the user can move from a web page showing a given edge to a page showing information about

the parent inferred network by clicking on a button that invokes very fast recall of the network record from the PostgreSQL inferred-network table, based on that unique primary key value – and vice versa.

- View the resulting inferred network in tabular format, and view details on individual nodes and edges. A web page table is produced, which lists the nodes; and for each node, the edges in which it participates, one edge per row. (Each edge thus appears twice in the table.) The edge scores are given, along with clickable links to pages that give details on the node (the gene or protein), on the edge, and on the node at the other end of the edge. Two versions of the table are displayed, depending upon whether the inferred edges are directed or undirected. Fig. 24.3 shows a part of such a table for an undirected protein–protein interaction network found in the bacterium *Rhodopseudomonas palustris* (*R. palustris*). Also available is a table that simply lists the inferred edges, one row per edge, with the two nodes involved and the edge score. The pages that give full descriptions on the nodes may contain background sequence level information on that gene or protein, if such information has been uploaded into SEBINI. Such a page may also contain clickable links to public databases. For example, we may retrieve the appropriate page from the NCBI Entrez web site for that gene or protein in a new window. As mentioned above, the details page for an edge will show the raw and processed node states (expression levels) for the two nodes involved, across the set of experiments in the

Launch Cytoscape		Data download - one line per node (File size = 109046 bytes)		Data download - Cytoscape SIF file format (File size = 142697 bytes)		Data download - Cytoscape SIF file format w/ info (File size = 360762 bytes)				
1) BEPro analysis summary file (HTML file, as text) 2) BEPro analysis journal file 3) BEPro prey bait freq of detection CSV file 4) BEPro prey bait posterior table CSV file 5) BEPro protein association probability CSV file		<input type="button" value="bring up alg file"/>								
# of nodes in inferred network=1668, # of (undirected) edges in inferred network=3255, # of nodes having ZERO edges=1108, # of rows required for full display=7618										
row cnt	node cnt	inferred node id / name	misc annotation	corresponding true node id / name	inferred edges					
					edge cnt	edge id	edge score	directed	type	inferred partner node id / name
-	-	-	-	-						
1	1	191817 RPA0001	class=bait desc-No description given in upload data file for this protein, sinc it is a bait which does not appear as prey in any run. Used.	114976 RPA0001	1	538935	0.979	no	pp	192974 / RPA2953 row_5953
2	-	-	-	-	2	538936	0.989	no	pp	192157 / RPA0183 row_4162
3	-	-	-	-	3	538937	0.994	no	pp	192371 / RPA0861 row_4708

Fig. 24.3. The start of a SEBINI table showing the nodes and undirected edges for an inferred protein–protein interaction network for the bacterium *Rhodopseudomonas palustris*.

data set. These web pages may be printed out as human-readable, permanent reports. Also, some basic topological statistics are reported out for each inferred network – average degree, minimum and maximum degree (undirected networks), minimum and maximum in-degree (directed), minimum and maximum out-degree (directed). Topological calculations are being expanded to include the full node in-degree and out-degree distributions, average clustering coefficient, and possibly other characteristics (graph diameter, characteristic path length, scale free exponent) important for structural analysis (41, 79, 101–103).

- 8) Export the inferred network, if so desired. The user can generate and store a text file for the inferred network in Cytoscape SIF format on the local client computer. Also, a text file with edges and associated node state values can be produced and passed to dynamic modeling tools.
- 9) Visualize the inferred network using Cytoscape. A set of Cytoscape files are automatically generated for each inferred network and stored in the directory structure for a project, for that given inferred network. In addition to the primary Cytoscape SIF file, with a listing of the edges and their scores, a large number of Cytoscape node and edge attribute files are generated, to allow the user to annotate the network graph displayed in Cytoscape. The user can invoke Cytoscape to display an inferred network via a button click from a web page listing that network. Cytoscape appears on the client computer via Java Web Start, with the inferred network automatically loaded.
- 10) Further analyze the inferred network in CABIN. From the Cytoscape plug-in menu, the user can invoke CABIN, which will then allow the user to bring in other public data sets of known edges (interactions) and compare those networks to the inferred network found in SEBINI. Or the user can then employ those data sources to extend the inferred network by various combinations of weighting and filtering. Once an extended or combined network has been created in CABIN, the user may place it back into the SEBINI database for permanent storage as a new inferred network.

The SEBINI flow of control for the analysis of algorithms operating on synthetic data sets is quite similar to the task sequence described above for the analysis of experimental data sets. One logs into the SEBINI web site and selects a project in the same manner. However, instead of creating an experimental network container and uploading an experimental data set into it, the user goes to a different set of web pages and creates a synthetic network container. The user then selects methods to build a

network topology and a set of corresponding artificial expression data sets for that topology. Once that is done, the flow of control is the same as before – we create a processed data set from the raw (artificial) data set, select an inference algorithm to use, enter values for the algorithm parameters, and run the algorithm on the processed data set. The inferred network can then be viewed in tabular format, as before, or visualized as a graph in Cytoscape. The difference here is that the user can also bring up a web page to view precision, recall, and F -measure statistics that measure how well the inference algorithm performed against the “gold standard”, the known artificial network, in terms of the number of correct and incorrect edges found.

At PNNL, algorithms in SEBINI have been used to analyze mRNA microarray and proteomics data coming from the EMSL Grand Challenge in Membrane Biology project (<http://mbgc.emsl.pnl.gov/>) focused on the study of nitrogen fixation and photosynthesis in the bacterium *Cyanobacter*. However, perhaps a more interesting case study is the use of SEBINI–CABIN in the U.S. Department of Energy’s Genomics:GTL Center for Molecular and Cellular Systems (CMCS) project, which is a joint Oak Ridge National Laboratory (ORNL) / PNNL multi-year collaboration to determine protein complexes and interaction networks in bacteria via mass spectrometry protein bait-prey experiments (<http://mippi.ornl.gov/>). SEBINI and CABIN now form the backbone of the exploratory analysis pipeline for this project. Evidence for potentially interacting prey proteins is uploaded into SEBINI for each bait experiment. Such evidence comes from the peptides that preliminary analysis via the well-known SEQUEST peptide mass spectra analysis algorithm has assigned to a particular protein. Once such a data set has been uploaded, a processing algorithm is invoked, which “collapses” the data set from possibly multiple pieces of peptide evidence for each protein in each experiment down to a single numeric value for each protein in each experiment. This collapse can be done in many ways, based on the parameters selected. The simplest means is to simply count the number of peptides that SEQUEST reported as evidence for a protein in a given experiment.

Note that while ORNL data sets are uploaded as text files, the PNNL bait-prey experiment data sets are extracted on-the-fly from an Oracle database. Thus, an uploader program that will dynamically retrieve records (e.g., a set of microarray runs) from a remote database, based on user-selected criteria, can easily be added to the set of SEBINI upload options.

Once the processed data set is created, the user selects the BEPro algorithm mentioned above to infer the set of protein–protein interactions. Such a resulting network is shown in **Fig. 24.4**. We can invoke Cytoscape and CABIN for further analysis of this inferred network.

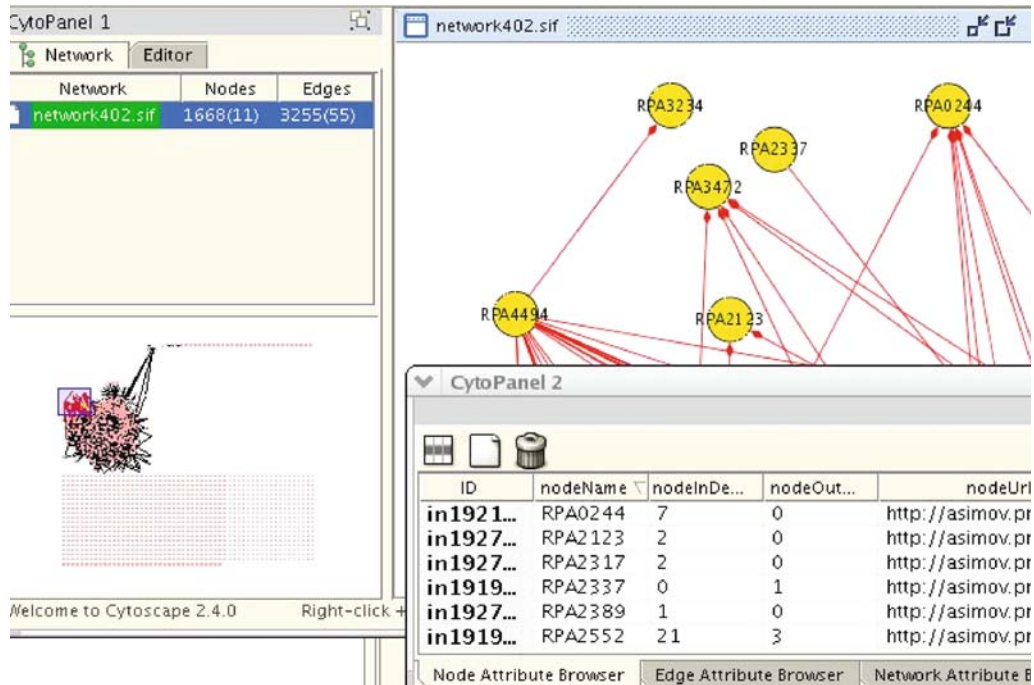


Fig. 24.4. The protein–protein interaction network shown here was inferred using the BEPro algorithm, operating on a set of 854 bait-prey experiments for the bacterium *Rhodospseudomonas palustris*. We see a part of the network graph in the Cytoscape window, along with information on the proteins (nodes) appearing in the Cytoscape attribute browser.

Figure 24.2 show the use of CABIN to validate experimental interactions for *R. palustris* obtained using the tandem affinity purification technique mass spectrometry experiments at ORNL. For the set of proteins in these interactions, we obtained evidence networks using the phylogenetic profile, gene cluster, gene neighborhood, and Rosetta stone methods of the Prolinks database (84). In addition, we obtained evidence networks using protein information from the interolog and regulog methods from the Bioverse (55, 104). The interolog method predicts an interaction between two proteins if they are both homologs of two proteins known to interact. Known protein interactions are gathered from the databases of experimentally determined protein–protein interactions (e.g., BIND (83), DIP (80–82)) and PSI-BLAST (105) is used to determine similarity between this set and all proteins in a target organism. Regulogs are regulatory interactions inferred by homology. A regulog is predicted by determining the similarity to a known transcription factor (TF) and the TF’s target protein. Finally, the nucleotide similarity in the upstream transcriptional promoter regions is determined and used to filter the regulog predictions: if there are similar promoter sequences, then a regulog is predicted.

In all, seven interaction networks are assigned a weight based on the confidence in the evidence source and imported into CABIN in this example. As can be seen from **Fig. 24.2**, the scatter plots show the correlation of the different interaction networks with respect to each other. We can clearly see the good agreement of the interolog predictions with the experimental interactions (points along the diagonal) and the low overlap of the regulog predictions with the other networks (points along the axis). This can be attributed to the fact that protein–protein interactions (and/or interologs) are not expected to overlap with regulatory interactions (and/or regulogs) since the types of interactions are very different. Regulatory interactions act through an intermediate (the promoter region) so the TF and TF target do not need to physically make contact. Only in (probably rare) cases where the protein produced from the TF gene target binds to its TF generally to inhibit its activity (an auto-regulatory loop) would you see both a protein–protein and regulatory interaction between the same pair of proteins.

Figure 24.2 show the exploratory analysis process in which we select the interactions (shown in black) that have high evidence in the pulldown experiments as well as the interolog predictions. These interactions are automatically selected in all the other views, showing their corresponding values in those views. Although the different views of the data give a deeper understanding of the multi-source data, the interpretation of an interaction network with more than a few hundred edges becomes difficult in a traditional network/graph like view. The use of filters helps in sub-setting the data by changing the cutoff for the evidence networks dynamically. As shown in **Figure 24.2** the interactions are filtered based on a value greater than 0.2 for the experimental observations. We can see a clear separation of the interactions based on the combined confidence from all the evidence sources in the Weighted Scaling View. Using the functionalities within CABIN, further exploratory data analysis can be carried out to validate the experimental interactions and, on conclusion of the analysis process, the high-confidence interactions can be saved in a local file or back into SEBINI.

5. Summary

The SEBINI–CABIN system offers an open source software platform for biological network inference and analysis. In addition to the large collection of inference algorithms the system makes available, permanent storage of the inferred networks in the

network-centric database allows for (1) post-processing and further inference and annotation via CABIN, and (2) further annotation and topological analysis within SEBINI (e.g., topology-based edge additions to existing core subnet using less stringent cutoffs).

We will continue to add to the capabilities of SEBINI and CABIN: additional inference and processing algorithms, better methods of generating artificial data sets, additional import/export techniques, and additional statistical algorithms for combining evidence from multiple sources. With new algorithms in this field being published on a regular basis, there remain a large number of promising inference algorithms (e.g., (106–112)) to add into SEBINI's toolkit. (An extensive list of relevant articles, current through mid-2006, many of which discuss new algorithms, can be found in (113)). Also, we are exploring refining or combining algorithms already present in SEBINI for improved results, using the SEBINI–CABIN system ourselves to explore improvement of the algorithms; quickly adding, testing, and comparing variants. Further, we will use our platform to develop expertise on how much data is needed, what are the appropriate parameter settings and cutoffs for each algorithm, what are the weaknesses of a given method compared to others on a common data set, what background information on a genome is most useful to supplement the primary gene expression data for a given algorithm, and so on.

We realize that adding general and genome-specific annotation from the public databases, and using such annotation as possible constraints on the edges that are inferred, is extremely important. (For example, if a gene's product is a protein that is known to be located in the membrane, that knowledge considerably lessens the probability that any inferred transcriptional regulatory edge coming out of that gene is correct.) Hence we will also be working to add to SEBINI's capabilities in this area.

It has not escaped our attention that a SEBINI-CABIN site adopted by a large community would provide a network-centric database whose edge records would constitute a resource exceptionally well-suited for investigating edge motifs in signaling, regulatory, and interaction networks (1, 114–116).

“Network biology is only in its infancy” (3). We do not yet know what inference algorithm(s) will perform best for what data sets. Theoretical guidance is lacking. But SEBINI–CABIN positions us to empirically test new algorithms, and easily modify or combine algorithms, while providing biologists much easier access to a growing collection of state-of-the-art algorithms. Moreover, as the high-throughput data sets continue to grow, the SEBINI–CABIN platform will aid in making the inference of network topologies a common starting point for further work in systems biology, such as dynamic modeling, rather than a seldom-reached end point, as is now the case.

Acknowledgments

The research described in this paper was conducted under the Laboratory Directed Research and Development Program at the Pacific Northwest National Laboratory (PNNL), a multiprogram national laboratory operated by Battelle for the U.S. Department of Energy, under Contract DE-AC06-76RL01830. Also, work for SEBINI has been supported by PNNL's William R. Wiley Environmental Molecular Science Laboratory (EMSL) and the EMSL Grand Challenge in Membrane Biology project, and by the joint ORNL / PNNL collaboration for the Genomes to Life Center for Molecular and Cellular Biology, project # 43930, US Department of Energy.

References

1. U. Alon, *An Introduction to Systems Biology – Design Principles of Biological Circuits*. Boca Raton: Chapman & Hall/CRC, 2007.
2. E. H. Davidson, *The Regulatory Genome – Gene Regulatory Networks in Development and Evolution*. Burlington: Elsevier, 2006.
3. A.-L. Barabasi and Z. N. Oltvai, “Network biology: understanding the cell’s functional organization,” *Nat. Rev. Genet.*, vol. 5, pp. 101–13, 2004.
4. A. S. N. Seshasayee, P. Bertone, G. M. Fraser, and N. M. Luscombe, “Transcriptional regulatory networks in bacteria: from input signals to output responses,” *Curr. Opin. Microbiol.*, vol. 9, pp. 511–9, 2006.
5. H. de Jong, “Modeling and simulation of genetic regulatory systems: a literature review,” *J Comput Biol*, vol. 9, pp. 67–103, 2002.
6. J. A. Papin, T. Hunter, B. O. Palsson, and S. Subramaniam, “Reconstruction of cellular signalling networks and analysis of their properties,” *Nat. Rev. Mol. Cell Biol.*, vol. 6, pp. 99–111, 2005.
7. E. P. van Someren, L. F. Wessels, E. Backer, and M. J. Reinders, “Genetic network modeling,” *Pharmacogenomics*, vol. 3, pp. 507–25, 2002.
8. L. F. Wessels, E. P. van Someren, and M. J. Reinders, “A comparison of genetic network models,” *Pac. Symp. Biocomput.*, vol. 6, pp. 508–519, 2001.
9. C.-H. Yuh, H. Bolouri, and E. H. Davidson, “Genomic cis-regulatory logic: experimental and computational analysis of a sea urchin gene,” *Science*, vol. 279, pp. 1896–902, 1998.
10. R. C. Taylor, A. Shah, C. Treatman, and M. Blevins, “SEBINI: software environment for biological network inference,” *Bioinformatics*, vol. 21, pp. 2706–8, 2006.
11. M. Singhal and K. Domico, “Collective analysis of biological interaction networks (CABIN),” www.sysbio.org/dataresources/cabin.stm, 2006.
12. M. Bansal, V. Belcastro, A. Ambesi-Impiombato, and D. di Bernardo, “How to infer gene networks from expression profiles,” *Mol. Syst. Biol.*, vol. 3, 2007.
13. V. Filkov, “Identifying gene regulatory networks from gene expression data (Chapter 27),” in *Handbook of Computational Molecular Biology*. Boca Raton: Chapman & Hall/CRC, 2005.
14. C. L. Barrett and B. O. Palsson, “Iterative reconstruction of transcriptional regulatory networks: an algorithmic approach,” *PLoS Comput. Biol.*, vol. 2, p. e52, 2006.
15. A. J. Butte and I. S. Kohane, “Mutual information relevance networks: functional genomic clustering using pairwise entropy measurements,” *Pac. Symp. Biocomput.*, vol. 5, pp. 418–429, 2000.
16. A. de la Fuente, P. Brazhnik, and P. Mendes, “Linking the genes: inferring quantitative gene networks from microarray data,” *Trends Genet.*, vol. 18, pp. 395–8, 2002.
17. P. D’haeseleer, X. Wen, S. Fuhrman, and R. Somogyi, “Linear modeling of mRNA expression levels during CNS development

- and injury," *Pac. Symp. Biocomput.*, vol. 4, pp. 41–52, 1999.
18. N. Friedman, M. Linial, J. Nachman, and D. Pe'er, "Using Bayesian networks to analyze expression data," *J. Comput. Biol.*, vol. 7, pp. 601–20, 2000.
 19. N. Friedman, "Inferring cellular networks using probabilistic graphical models," *Science*, vol. 303, pp. 799–805, 2004.
 20. F. Gao, B. Foat, and H. Bussemaker, "Defining transcriptional networks through integrative modeling of mRNA expression and transcription factor binding data," *BMC Bioinformatics*, vol. 5, p. 31, 2004.
 21. T. S. Gardner, D. di Bernardo, D. Lorenz, and J. J. Collins, "Inferring genetic networks and identifying compound mode of action via expression profiling," *Science*, vol. 301, pp. 102–5, 2003.
 22. F. Geiger, J. Timmer, and C. Fleck, "Reconstructing gene-regulatory networks from time series, knock-out data, and prior knowledge," in *BMC Syst. Biol.*, vol. 1, 2007.
 23. R. Guthke, U. Moller, M. Hoffmann, F. Thies, and S. Topfer, "Dynamic network reconstruction from gene expression data applied to immune response during bacterial infection," *Bioinformatics*, vol. 21, pp. 1626–34, 2005.
 24. A. J. Hartemink, D. K. Gifford, T. S. Jaakkola, and Y. R. A., "Combining location and expression data for principled discovery of genetic regulatory network models," *Pac. Symp. Biocomput.*, vol. 7, pp. 37–43, 2002.
 25. D. Husmeier, "Sensitivity and specificity of inferring genetic regulatory interactions from microarray experiments with dynamic Bayesian networks," *Bioinformatics*, vol. 19, pp. 2271–82, 2003.
 26. T. Ideker, V. Thorsson, and R. M. Karp, "Discovery of regulatory interactions through perturbation: inference and experimental design," in *Pacific Symposium on Biocomputing Hawaii*, 2000, pp. 305–16.
 27. S. Liang, S. Fuhrman, and R. Somogyi, "REVEAL: a general reverse engineering algorithm for inference of genetic network architectures," *Pac. Symp. Biocomput.*, vol. 3, pp. 18–29, 1998.
 28. L. Mao and H. Resat, "Probabilistic representation of gene regulatory networks," *Bioinformatics*, vol. 10, pp. 2258–69, 2004.
 29. C. L. Myers, D. Robson, A. Wible, M. A. Hibbs, C. Chiriac, C. L. Theesfeldt, K. Dolinski, and O. G. Troyanskaya, "Discovery of biological networks from diverse functional genomic data," *Genome Biol.*, vol. 6, p. r114, 2005.
 30. N. Nariai, Y. Tamada, S. Imoto, and S. Miyano, "Estimating gene regulatory networks and protein-protein interactions of *Saccharomyces cerevisiae* from multiple genome-wide data," *Bioinformatics*, vol. 21, pp. ii206–12, 2005.
 31. D. Pe'er, A. Regev, and A. Tanay, "Minreg: inferring an active regulator set," *Bioinformatics*, vol. 18, pp. 258S–67S, 2002.
 32. D. Pe'er, A. Regev, G. Elidan, and N. Friedman, "Inferring subnetworks from perturbed expression profiles," *Bioinformatics*, vol. 17, pp. 215S–24S, 2001.
 33. B. E. Perrin, L. Ralaivola, A. Mazurie, S. Bottani, J. Mallet, and F. d'Alche-Buc, "Gene networks inference using dynamic Bayesian networks," *Bioinformatics*, vol. 19, pp. ii138–48, 2003.
 34. J. Qian, J. Lin, N. M. Luscombe, H. Yu, and M. Gerstein, "Prediction of regulatory networks: genome-wide identification of transcription factor targets from gene expression data," *Bioinformatics*, vol. 19, pp. 1917–26, 2003.
 35. D. J. Reiss, N. S. Baliga, and R. Bonneau, "Integrated bi-clustering of heterogeneous genome-wide datasets for the inference of global regulatory networks," *BMC Bioinformatics*, vol. 7, 10.1186/1471-2105-7-280, 2006.
 36. K. Sachs, O. Perez, D. Pe'er, D. A. Lauffenburger, and G. P. Nolan, "Causal protein-signaling networks derived from multiparameter single-cell data," *Science*, vol. 308, pp. 523–9, 2005.
 37. E. E. Schadt, J. Lamb, X. Yang, J. Zhu, S. Edwards, D. Guhathakurta, S. K. Sieberts, S. Monks, M. Reitman, C. Zhang, P. Y. Lum, A. Leonardson, R. Thieringer, J. M. Metzger, L. Yang, J. Castle, H. Zhu, S. F. Kash, T. A. Drake, A. Sachs, and A. J. Lusis, "An integrative genomics approach to infer causal associations between gene expression and disease," *Nat. Genet.*, vol. 37, pp. 710–7, 2005.
 38. M. Schena, D. Shalon, R. W. Davis, and P. O. Brown, "Quantitative monitoring of gene expression patterns with a complementary DNA microarray," *Science*, vol. 270, pp. 467–70, 1995.
 39. E. Segal, R. Yelensky, and D. Koller, "Genome-wide discovery of transcriptional modules from DNA sequence and gene expression," *Bioinformatics*, vol. 19, pp. i264–72, 2003.

40. E. Segal, B. Taskar, A. Gasch, N. Friedman, and D. Koller, "Rich probabilistic models for gene expression," *Bioinformatics*, vol. 17, pp. S243–52, 2001.
41. V. A. Smith, E. D. Jarvis, and A. J. Hartemink, "Evaluating functional network inference using simulations of complex biological systems," *Bioinformatics*, vol. 18, pp. S216–24, 2002.
42. L. Soinov, M. A. Krestyaninova, and A. Brazma, "Toward reconstruction of gene networks from expression data by supervised learning," *Genome Biol.*, vol. 4, p. R6, 2003.
43. Y. Tamada, S. Kim, H. Bannai, S. Imoto, K. Tashiro, S. Kuhara, and S. Miyano, "Estimating gene networks from gene expression data by combining Bayesian network model with promoter element detection," *Bioinformatics*, vol. 19, pp. ii227–36, 2003.
44. R. C. Taylor, "Reconstruction of metabolic and genetic networks from gene expression perturbation data using a Boolean model: construction of a simulation testbed and an empirical exploration of some of the limits," *Doctoral Dissertation*, George Mason University, Fairfax, VA, 2003.
45. O. G. Troyanskaya, K. Dolinski, A. B. Owen, R. B. Altman, and D. Botstein, "A Bayesian framework for combining heterogeneous data sources for gene function prediction (in *Saccharomyces cerevisiae*)," *Proc. Natl. Acad. Sci. U.S.A.*, vol. 100, pp. 8348–53, 2003.
46. A. Wagner, "Estimating coarse gene network structure from large-scale gene perturbation data," *Genome Res.*, vol. 12, pp. 309–15, 2002.
47. A. Wagner, "How to reconstruct a large genetic network from n gene perturbations in fewer than $n(2)$ easy steps," *Bioinformatics*, vol. 17, pp. 1183–97, 2001.
48. A. Wagner, "Reconstructing pathways in large genetic networks from genetic perturbations," *J. Comput. Biol.*, vol. 11, pp. 53–60, 2004.
49. Y. Wang, T. Joshi, X. Zhang, D. Xu, and L. Chen, "Inferring gene regulatory networks from multiple microarray datasets," *Bioinformatics*, vol. 22, pp. 2413–20, 2006.
50. D. C. Weaver, C. T. Workman, and G. D. Stormo, "Modeling regulatory networks with weight matrices," *Pac. Symp. Biocomput.*, vol. 4, pp. 112–123, 1999.
51. A. V. Werhli, M. Grezegorczyk, and D. Husmeier, "Comparative evaluation of reverse engineering gene regulatory networks with relevance networks, graphical gaussian models and bayesian networks," *Bioinformatics*, vol. 22, pp. 2523–31, 2006.
52. C. J. Wolfe, I. S. Kohane, and A. J. Butte, "Systemic survey reveals general applicability of "guilt-by-association" within gene coexpression networks," *BMC Bioinformatics*, vol. 6, p. 227, 2005.
53. C.-C. Wu, H.-C. Huang, H.-F. Juan, and S.-T. Chen, "GeneNetwork: an interactive tool for reconstruction of genetic networks using microarray data," *Bioinformatics*, vol. 18, pp. 3691–3, 2004.
54. B. Xing and M. J. Van der Laan, "A statistical method for constructing transcriptional regulatory networks using gene expression and sequence data," *J. Comput. Biol.*, vol. 12, pp. 229–46, 2005.
55. H. Yu, V. A. Smith, P. P. Wang, A. J. Hartemink, and E. D. Jarvis, "Advances to bayesian network inference for generating causal networks from observational biological data.," *Bioinformatics*, vol. 20, pp. 3594–603, 2004.
56. W. Zhao, E. Serpedin, and E. R. Dougherty, "Inferring gene regulatory networks from time series data using the minimum description length principle," *Bioinformatics*, vol. 22, pp. 2129–35, 2006.
57. X. Zhou, X. Wang, R. Pal, I. Ivanov, M. Bittner, and E. R. Dougherty, "A Bayesian connectivity-based approach to constructing probabilistic gene regulatory networks," *Bioinformatics*, vol. 20, pp. 2918–27, 2004.
58. M. Stetter, G. Deco, and M. Dejori, "Large-scale computational modeling of genetic regulatory networks," *Artif. Intell. Rev.*, vol. 20, pp. 75–93, 2003.
59. T. Chu, C. Glymour, R. Scheines, and P. Sprites, "A statistical problem for inference to regulatory structure from associations of gene expression measurements with microarrays," *Bioinformatics*, vol. 19, pp. 1147–52, 2003.
60. T. M. Cover and J. A. Thomas, *Elements of Information Theory*, 1st ed. New York: John Wiley & Sons, 1991.
61. C. O. Daub, R. Steuer, J. Selbig, and S. Kloska, "Estimating mutual information using B-spline functions – an improved similarity measure for analysing gene expression data," *BMC Bioinformatics*, vol. 5, 2004.
62. Wikipedia, "Pearson product-moment correlation coefficient," 2006, p. Wikipedia entry.
63. K. Murphy, "Bayes net toolbox for matlab (open source project on Source Forge)," 2006.

64. F. V. Jensen, *An Introduction to Bayesian Networks*. New York: UCL Press Limited, 1996.
65. M. I. Jordan, "Learning in graphical models," 1st ed. Cambridge: MIT Press, 1998.
66. R. E. Neapolitan, *Learning Bayesian networks*. Upper Saddle River: Pearson Education Inc., 2004.
67. D. Pe'er, "Bayesian network analysis of signaling networks: a primer," in *Science STKE*, 2005, pp. on-line primer.
68. S. G. Bottcher and C. Dethlefsen, "Deal: a package for learning Bayesian networks," *J. Stat. Softw.*, vol. 8, p. i20, 2003.
69. J. Han and M. Kamber, *Data Mining Concepts and Techniques*. San Diego: Morgan Kaufman Publishers, 2001.
70. Oak Ridge National Laboratory and Pacific Northwest National Laboratory, "Microbial Protein-Protein Interactions (MiPPI) project," *project web site*: <http://mippi.ornl.gov>, 2007.
71. J. L. Sharp, K. K. Anderson, D. S. Daly, D. L. Auberry, W. R. Cannon, A. M. White, and V. Kery, "Inferring protein-protein associations with affinity isolation LC-MS/MS assays," *J. Proteome Res.*, vol. 6(9), pp. 3788–95, 2007.
72. M. A. Gilchrist, L. A. Salter, and A. Wagner, "A statistical framework for combining and interpreting proteomic datasets," *Bioinformatics*, vol. 20, pp. 689–700, 2004.
73. J. Gilmore, D. L. Auberry, A. M. White, J. L. Sharp, K. K. Anderson, and D. S. Daly, *Bayesian Estimator of Protein-Protein Association Probabilities (BEPro) web site*: <http://www.pnl.gov/statistics/bepro3/index.htm>, 2006.
74. P. T. Spellman, G. Sherlock, M. Q. Zhang, V. R. Iyer, K. Anders, M. Eisen, P. O. Brown, D. Botstein, and B. Futcher, "Comprehensive identification of cell cycle regulated genes of the yeast *Saccharomyces cerevisiae* by microarray hybridization," *Mol. Biol. Cell*, vol. 9, pp. 3273–97, 1998.
75. M. Eisen, P. Spellman, P. O. Brown, and D. Botstein, "Cluster analysis and display of genome-wide expression patterns," *Proc. Natl. Acad. Sci. U.S.A.*, vol. 95, pp. 14863–8, 1998.
76. B. S. Everitt, *Cluster Analysis*, 3rd ed. New York: Arnold, 1993.
77. P. Baldi and G. W. Hatfield, *DNA Microarrays and Gene Expression*, 1st ed. Cambridge: Cambridge University Press, 2002.
78. T. Speed, "Statistical analysis of gene expression microarray data," Boca Raton: Chapman & Hall, 2003.
79. A. V. Lukashin, M. E. Lukashev, and R. Fuchs, "Topology of gene expression networks as revealed by data mining and modeling," *Bioinformatics*, vol. 19, pp. 1909–16, 2003.
80. I. Xenarios, E. Fernandez, L. Salwinski, X. J. Duan, M. J. Thompson, E. M. Marcotte, and D. Eisenberg, "DIP: the database of interacting proteins: 2001 update," *Nucleic Acids Res.*, vol. 29, pp. 239–41, Jan 1 2001.
81. I. Xenarios, D. W. Rice, L. Salwinski, M. K. Baron, E. M. Marcotte, and D. Eisenberg, "DIP: the database of interacting proteins," *Nucleic Acids Res.*, vol. 28, pp. 289–91, Jan 1 2000.
82. I. Xenarios, L. Salwinski, X. J. Duan, P. Higney, S. M. Kim, and D. Eisenberg, "DIP: the database of interacting proteins: a research tool for studying cellular networks of protein interactions," *Nucleic Acids Res.*, vol. 30, pp. 303–5, Jan 1 2002.
83. G. Bader and H. CW., "BIND: a data specification for storing and describing biomolecular interactions, molecular complexes and pathways," *Bioinformatics*, vol. 16, pp. 465–77, 2000.
84. P. Bowers, M. Pellegrini, M. Thompson, J. Fierro, T. Yeates, and D. Eisenberg, "Prolinks : a database of protein functional linkages derived from coevolution," *Genome Biol.*, vol. 5, p. R35, 2004.
85. P. Shannon, A. Markiel, O. Ozier, N. S. Baliga, J. T. Want, D. Ramage, N. Amin, B. Schwikowski, and T. Ideker, "Cytoscape: a software environment for integrated models of biomolecular interaction networks," *Genome Res.*, vol. 13, pp. 2498–504, 2003.
86. PostgreSQL Global Development Group, *PostgreSQL web site*: <http://www.postgresql.org>
87. Apache Software Foundation, *Apache Tomcat web site*: <http://tomcat.apache.org>, 2007.
88. MathWorks, 2007, p. MATLAB home web site.
89. R. Gentleman and R. Ihaka, "The R project for statistical computing," *R home web site*: <http://www.r-project.org>, 1997.
90. A. J. Hartemink, "Bayesian network inference with Java Objects (Banjo)," 2005, Banjo web site at Duke.

91. A. Hartemink, "Banjo: Bayesian network inference with Java Objects," *web site*: <http://www.cs.duke.edu/~amink/software/banjo/>, 2005.
92. A. A. Margolin, K. Wang, W. K. Lim, M. Kustagi, I. Nemenman, and A. Califano, "Reverse engineering cellular networks," *Nat. Protoc.*, vol. 1, pp. 663–72, 2006.
93. A. A. Margolin, I. Nemenman, K. Basso, C. Wiggins, G. Stolovitzky, R. Dalla-Favera, and A. Califano, "ARACNE: an algorithm for the reconstruction of gene regulatory networks in a mammalian cellular context," *BMC Bioinformatics*, vol. 7, pp. S1–7, 2006.
94. A. J. Hartemink, "Reverse engineering gene regulatory networks," *Nat. Biotech.*, vol. 23, pp. 554–5, 2005.
95. MDeC Bioinformatics core facility at the Columbia Genome Center, "ARACNE (Algorithm for the Reconstruction of Accurate Cellular Networks)," *ARACNE Algorithm Download Site*: <http://amdec-bioinfo.cu-genome.org/html/ARACNE.htm>, 2006.
96. J. J. Faith, B. Hayete, J. T. Thaden, I. Mogno, J. Wierzbowski, G. Cottarel, S. Kasif, J. J. Collins, and T. S. Gardner, "Large-scale mapping and validation of escherichia coli transcriptional regulation from a compendium of expression profiles," *PLoS Biol.*, vol. 5, pp. 54–66, 2007.
97. Gardner lab, "Context likelihood or relatedness (CLR) algorithm," *CLR Algorithm Download Site*: <http://gardnerlab.bu.edu>, 2006.
98. H. Sauro, "Systems biology workbench sysbio.org web site," SBW home web site, 2006.
99. P. Mendes, W. Sha, and K. Ye, "Artificial gene networks for objective comparison of analysis algorithms.," *Bioinformatics*, vol. 19, pp. ii122–9, 2003.
100. T. Van den Bulcke, K. L. Van Leemput, B. Naudts, P. van Remortel, M. Hongwu, A. Verschoren, B. De Moor, and K. Marchal, "SynTReN: a generator of synthetic gene expression data for design and analysis of structure learning algorithms," *BMC Bioinformatics*, vol. 7, 2006.
101. D. Tuck, H. Kluger, and Y. Kluger, "Characterizing disease states from topological properties of transcriptional regulatory networks," *BMC Bioinformatics*, vol. 7, p. 236, 2006.
102. A. Vazquez, R. Dobrin, D. Sergi, J.-P. Eckmann, Z. N. Oltvai, and A.-L. Barabasi, "The topological relationship between the large-scale attributes and local interaction patterns of complex networks," *Proc. Natl. Acad. Sci. U.S.A.*, vol. 101, pp. 17940–5, 2004.
103. A. Wagner and D. Fell, "The small world inside large metabolic networks," *Proc. Roy. Soc. Lond. Ser. B*, vol. 268, pp. 1803–10, 2001.
104. J. McDermott and R. Samudrala, "Bioverse: functional, structural, and contextual annotation of proteins and proteomes," *Nucleic Acids Res.*, vol. 31, pp. 3736–7, 2003.
105. S. Altschul, T. Madden, A. Schaffer, J. Zhang, Z. Zhang, W. Miller, and D. Lipman, "Gapped blast and psi-blast: a new generation of protein database search programs.," *Nucleic Acids Res.*, vol. 25, pp. 3389–402, 1997.
106. M. Bansal, G. D. Gatta, and D. di Bernardo, "Inference of gene regulatory networks and compound mode of action from time course gene expression profiles," *Bioinformatics*, vol. 22, pp. 815–22, 2006.
107. R. Bonneau, D. J. Reiss, P. Shannon, M. Facciotti, L. Hood, N. S. Baliga, and V. Thorsson, "The Inferelator: an algorithm for learning parsimonious regulatory networks from systems-biology data sets de novo," *Genome Biol.*, vol. 7, 2006.
108. X.-W. Chen, G. Anantha, and X. Wang, "An effective structure learning method for constructing gene networks," *Bioinformatics*, vol. 22, pp. 1367–74, 2006.
109. A. de la Fuente, N. Bing, I. Hoeschele, and P. Mendes, "Discovery of meaningful associations in genomic data using partial correlation coefficients," *Bioinformatics*, vol. 20, pp. 3565–74, 2004.
110. B. Hayete, T. S. Gardner, and J. J. Collins, "Size matters: network inference tackles the genome scale," *Mol. Syst. Biol.*, vol. 3, 2007.
111. J. J. Rice, Y. Tu, and G. Stolovitzky, "Reconstructing biological networks using conditional correlation analysis," *Bioinformatics*, vol. 21, pp. 765–73, 2004.
112. M. Zou and S. D. Conzen, "A new dynamic Bayesian network (DBN) approach for identifying gene regulatory networks from time course data," *Bioinformatics*, vol. 21, pp. 71–9, 2005.
113. F. Markowetz, "A bibliography on learning causal networks of gene interactions (July 31, 2006)," Princeton University,

- Lewis-Sigler Institute of Integrative Genomics, 2005.
114. R. Milo, S. S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, and U. Alon, "Network motifs: simple building blocks of complex networks," *Science*, vol. 298, pp. 824–827, 2002.
 115. F. Schreiber and H. Schwobbermeyer, "MAVisto: a tool for the exploration of network motifs," *Bioinformatics*, vol. 21, pp. 3572–4, 2005.
 116. S. S. Shen-Orr, R. Milo, S. Mangan, and U. Alon, "Network motifs in the transcriptional regulation network of *Escherichia coli*," *Nat. Genet.*, vol. 31, pp. 64–8, 2002.

INDEX

A

- aaRSs, *see* Aminoacyl-tRNA synthetases
- aaRS-tRNA complexes structure,
 - usage..... 443–444
- ABL-BCR pathway of human, network transition
 - probabilities of..... 263–264
- Acute myeloid leukemia 257
- Additive approximation..... 25, 34
- A-GLAM software package
 - hardware..... 4
 - intergenic regions of cluster files 17–21
 - network comparison operation method
 - functional similarity network construction..... 8–9
 - identification of cis-regulatory elements in
 - promoter regions..... 11–13
 - identification of dense gene clusters 10–11
 - identification of functional motifs..... 5
 - identification of proximal promoter regions..... 11
 - individual score and E-value..... 6–7
 - intersection network construction 9–10
 - size of co-expression networks..... 5
 - software 4–5
- Algorithm comparison and SEBINI..... 560
 - See also* Software Environment for Biological Network Inference
- Alzheimer’s Disease (AD)..... 451
 - protein, interaction of..... 458–459
 - related protein
 - SNP and disease mutation in..... 454
 - SVM and SIFT in 457
- Aminoacyl-tRNA biosynthesis pathway 505
- Aminoacyl-tRNA synthetases 443–446
- AML, *see* Acute myeloid leukemia
- API, *see* Application Programming Interface
- Apolipoprotein, decision trees for 393
- Application Programming Interface..... 512, 523
 - client, role of..... 523–525
 - documentation 526–527
 - role of bioverse 523
 - usage of 525–526
 - versioning of..... 530
- Arabidopsis* Genome Initiative locus identifier
 - (AGI ID)..... 110
- Arabidopsis thaliana* 108, 114–115
- Array data, pathway inference
 - and modeling of..... 232–237
 - See also* Genome-wide data generation, experimental methods for

B

- Bacillus subtilis* 108, 440
- Bacterial two-component system, data collection
 - and alignments for 428–429, 434–435
 - See also* Specificity determining residues
- Bacteriophages, role of..... 337–338
- Bagging method, in gene clusters..... 487
- Bayesian approach, for gene expression..... 488–490
 - See also* Cluster algorithms, for gene expression
- Bayesian estimation technique, in intron
 - evolution 358–359
- Bayesian Estimator of Protein–Protein Association
 - Probabilities 553
- Bayesian expression network
 - definition of 226
 - physical interaction networks,
 - measurement of..... 240–241
 - TF-gene regulation, usages of..... 239–240
- Bayesian marginal log 5
- Bayesian network approach, features of..... 237
- Bayes nets, *see* Bayesian expression network, definition of
- BellaVista protocol..... 526
- Benjamini & Hochberg False Discovery Rate (FDR)
 - correction 9
- BEPro, *see* Bayesian Estimator of Protein–Protein Association Probabilities
- Bergey’s Manual of Determinative and Systematic Bacteriology 173
- Biased reference set, in GO biological process..... 468
 - See also* Supervised learning
- Biclustering method..... 205–206
- Bidirectional best-hit procedure, of network
 - reconstruction 168
- BIND, *see* Biomolecular Interaction Network Database
- BIND database 69, 84
- BiNGO plug-in 9
- Binomial tau-leaping method..... 322, 323
- Biochemical control theory (BCT), for biochemical
 - network 275–276

- Biochemical control (*continued*)
- dynamic motifs
 - bistable systems 298–299
 - feedback and oscillatory systems 300–304
 - and engineering control theory 290
 - frequency response 291–295
 - stability analysis 295–297
 - linear perturbation analysis
 - control coefficients 278–279
 - elasticities and control coefficient, relationship 279–281
 - elementary processes 276–277
 - pathway motifs, linear analysis of
 - branched pathways 283–285
 - cyclic systems 285–287
 - negative feedback of 287–290
 - straight chains 281–283
- Biochemical network, steady-state solution, for 275
- Biochemical pathways
 - construction of 227–228
 - definition of 226
- Biochemical Systems Theory 276
- Biological network inference and analysis 552–555
- control genes, identification of 556
 - gene function and target identification 555
 - irrelevant genes, elimination of 555
- Biological networks
 - benefits of 146–147
 - dynamic behavior of 251
 - emergence of 146
 - hierarchical module structure 147–149
 - metabolic networks 147
 - reconstruction, goals of 229
 - structure, determination of 421
 - topological organization of cellular metabolism 147
 - metabolic network of *Escherichia coli*, case study 151–156
 - method for finding 150–151
- Biological replicate 203
- Biological systems
 - computational representation of 535–536
 - kinetic modeling of
 - discrete stochastic simulation, acceleration of 320–324
 - historical perspectives of 311–315
 - network size, scaling properties of 326–327
 - simulation methods for 316–319
 - software tools for 328–330
 - SSA fundamentals 319–320
 - thermodynamic limit 324–325
- BioMoby web services, in bioinformatics tools 536
- Biomolecular Interaction Database 556
- Biomolecular Interaction Network
 - Database 104, 106–107, 129, 514
- Bioverse
 - API
 - client application in 523
 - usage of 523, 525–526
 - XML-RPC protocol in 526
 - audiences in 512
 - data
 - creation and presentation 515
 - organisation 512–513
 - pipeline 515–518
 - protinfo 518
 - sources 513–515
 - database in 518
 - data tracking and 541
 - ETL 546–547
 - function identifier 527–528
 - importance of 511–512
 - integrator 524
 - model design
 - functional data 544–546
 - sequence data 541–542
 - taxonomy data 542–544
 - technology 534
 - web application
 - HTML templates and 520
 - implementation of 519
 - role of 515
- See also* Application Programming Interface
- Biozon database 69
- BL2SEQ 117
- BLAST 82, 118
 - in enzyme detection 374
- BLASTclust procedure, of network
 - reconstruction 168, 176
- BLOCK matrix 33
- BMP7, *see* Bone morphogenetic protein 7
- BMP7 and ACVR2B interaction, in lung cancer
 - co-expression 260
- BOND database 106–107, 129
- Bone morphogenetic protein 7 260
- Bottom-Up modelling, experimental design for
 - building of 344–347
 - decouple key components 343–344
 - mutations effects on 344
 - network subset isolation 343
 - in silico experiments, model predictions in 347–348
- B-spline approximation 252
 - See also* Gene co-expression
- B τ L method, *see* Binomial tau-leaping method
- C**
- C4.5 algorithm, in protein modeling 385
- CABIN, *see* Collective Analysis of Biological Interaction Networks

<i>Caenorhabditis elegans</i>	107–110, 113–114, 146
Cancers and normal tissue, differential co-expression of	257–261
CCL23, <i>see</i> Chemokine ligand 23	
CCR1, <i>see</i> Chemokine receptor 1	
Cellular network dynamics, quantitative approaches of	270–271
Center for Molecular and Cellular Systems	567
CERs, <i>see</i> Co-evolving residues	
CHARMM.....	28, 34–36
BLOCK module in.....	31–32
CPT keyword.....	30
and dual-topology.....	31
script file.....	30
CHARMM's HBUILD module.....	28
Chemical Langevin equation.....	325
Chemical master equation	314, 319
Chemical transformation, in biological networks.....	276–277
Chemokine ligand 23	260
Chemokine receptor 1	260
ChIP-chip.....	24
CHIP on-chip data.....	239
CHIP on chip technique, in protein-DNA interactions measurement.....	229
Chromosomal single-copy reporters, usages of.....	340
Chronic myeloid leukemia.....	261
CLE, <i>see</i> Chemical Langevin equation	
ClustalW algorithm.....	85, 118–119
Cluster algorithms	
in co-regulated genes identification.....	230–232
for gene expression.....	485–488
algorithms	494
best clustering, determination of.....	500–501
cellular pathways, association of clusters in.....	502–506
double index assessment	488
external index of validity.....	490–491
hierarchical clustering algorithms.....	482–483
individual clusters, assessment of.....	501–502
internal index of validity	488–490
noisy data, algorithms for	484–485
parameter optimization.....	495–500
partitional clustering algorithms.....	481–482
pre-processing similarity data	494
reference set	492
similarity measures.....	493
spectral clustering algorithms	483–484
CMCS, <i>see</i> Center for Molecular and Cellular Systems	
CME, <i>see</i> Chemical master equation	
CML, <i>see</i> Chronic myeloid leukemia	
<i>cMonkey</i>	206
CO4A5_HUMAN protein.....	454
CoD, <i>see</i> Coefficient of determination	
Coefficient of determination	250
estimation of.....	252–254
<i>See also</i> Gene co-expression	
Co-evolving residues.....	424
in ATPase and DD domains.....	437–440
in class Ia aaRSs.....	443–446
conservation and variation, sources.....	425
in double alignments, determination.....	433
genomic proximity in.....	426
modeled structures.....	440–441
procedures in determining.....	426–427
protein duplication, consequences of.....	425–426
in RR domain	435–437
statistical significance estimation and computation	434
CoExMiner algorithm.....	250
biological applications of	
disease genes identification and drug targets of leukemia.....	261–265
ligand-receptor pairs, co-expression of.....	257–261
Co-expressed genes.....	205
Co-expression network, <i>see</i> Correlation network, definition of	
Collective Analysis of Biological Interaction Networks.....	551
development and advantages of.....	556
exploratory analysis and	561–562
find/search functionality and	562
inferred network in	566
inferred network structures.....	560
java and	559
network import in.....	561
network subsets and	562
<i>Rhodopseudomonas palustris</i> and.....	568
role of.....	552
and SEBINI system	559
Computational modelling, LacZ units in.....	341
Computational systems biology, challenges in.....	249–250
Condition-specific sub-networks, categories of.....	235
Connectivity theorem	280
<i>See also</i> Biochemical control theory (BCT), for biochemical network	
Conservation index (C.I.), for network motifs.....	172, 177
Context-specific clustering technique	237
Co-regulated genes	205
identification, cluster analysis in.....	230–232
<i>See also</i> Genome-wide data generation, experimental methods for	
Correlated mutations, <i>see</i> Co-evolving residues	
Correlation network, definition of	226
Cytoscape, usages of	4, 10, 233
D	
DAEs, <i>see</i> Differential-algebraic equations	
DAG, <i>see</i> Directed acyclic graph	

- Database of Interacting Proteins 69, 514, 556
Database of Ligand– Receptor Partners 257–258
Data warehouse 536–537
 construction of
 extracting, transforming and loading 539–541
 model design 539
 dimensional models 537–539
 relational databases 537
DBNs, *see* Dynamic Bayesian networks
DBTBS 104, 108
DD, *see* Dimerization domain
Decision tree learning model, for protein modeling
 classification of 383–384
 training procedure for 384–387
Diabetes Mellitus (DM) 193
Differential–algebraic equations 312
Dimerization domain 434
DIP, *see* Database of Interacting Proteins
Directed Acyclic Graph (DAG) 193, 204, 542
Discrete stochastic simulation
 acceleration of 320–324
 in biological systems study 313
 See also Biological systems, kinetic modeling of
DLRP, *see* Database of Ligand– Receptor Partners
DNA microarrays, usages of 230
Dollo parsimony, in intron evolution 358
Double index assessment, for gene expression 488
 See also Cluster algorithms, for gene expression
Drosophila melanogaster 108, 115, 146, 529
Dynamical stiffness, definition of 312
Dynamic Bayesian networks 241
Dynamic motifs
 bistable systems 298–299
 feedback and oscillatory systems 300–304
 See also Biochemical control theory (BCT),
 for biochemical network
E
EBI UniProt database, usage of 227
EC classification test, for protein domains 400–402
EC hierarchy 376
 See also Enzyme class prediction
EC number, *see* Enzyme Commission number
Elasticity coefficient, definition of 277
EM-like algorithm, for protein subset selection 80–81
Engineering control theory and BCT 290
 frequency response 291–295
 stability analysis 295–297
 See also Biochemical control theory (BCT),
 for biochemical network
Enzyme class prediction
 EC hierarchy 376
 proteins
 alternate representation of 378–379
 multiple aspects of 380–381
 sequence similarity 376–377
 structure-based approaches 377–378
Enzyme Commission number 373, 376
Enzyme function prediction
 data preparation and feature extraction method
 database characteristics of 382–383
 predicted features of 382
 sequence characteristics of 381–382
 decision tree learning model, methods for
 decision tree model 383–384
 decision trees, training procedure for 384–387
 learning algorithm, optimization
 method of 390–392
 stochastic decision trees, mixture of
 decision trees, evaluation of 388–390
 stochastic framework 387–388
Enzymes in organisms, role 374
eQTL mapping, *see* Expression quantitative trait locus
 mapping
Equilibrium 2023
Escherichia coli 302, 338
 etabolic network, case study
 color-coding of hierarchical tree 153–154
 data source 151
 network generation 152–153
 pathways of pyrimidine metabolism 154–156
Escherichia coli K-12 108
ET, *see* Evolutionary tracing
ETL, *see* Extraction, Transformation, and Loading
Euclidean metric, in mRNA expression 493
Eukaryotes 112
Eukaryotes, prediction and identification of 3–4
Eukaryotic gene structure
 bayesian estimation technique in 358–359
 expectation-maximization algorithm 359
 maximum likelihood estimation techniques in
 materials in 360–361
 methodologies in 361–368
 properties of 360
Evolutionary tracing 423
Expectation-maximization algorithm, in introns
 evolution 359–360
Expression quantitative trait loci (eQTL), analysis
 building molecular interaction graph 213
 drawbacks 221
 finding linked loci 213
 identifying linear pathways 214–215
 random walks across an interaction graph 215–216
 using gene knockouts 216–217
Expression quantitative trait locus mapping 237
Extraction, Transformation, and
 Loading 539, 540, 546

F

FA8_HUMAN protein 454
 FA9_HUMAN protein 454
 Fact table
 characteristics of 539
 definition of 538
 False negative/positive 203
 FASTA files 11
 Felsenstein's pruning algorithm 364
 Flux control coefficients 288, 289
 Flux summation theorem 280
 See also Biochemical control theory (BCT),
 for biochemical network

G

β -galactosidase 203
 β -galactosidase, measurement 340
GAL gene expression 232–233
 Gaussian cumulative distribution function 432
 Gene co-expression
 computational model of
 mixed patterns 251–254
 pathway dynamics 254–257
 study of 250
 Gene expression
 cluster algorithms for 485–488
 algorithms 494
 best clustering, determination of 500–501
 cellular pathways, association
 of clusters in 502–506
 double index assessment 488
 external index of validity 490–491
 hierarchical clustering algorithms 482–483
 individual clusters, assessment of 501–502
 internal index of validity 488–490
 noisy data, algorithms for 484–485
 parameter optimization 495–500
 partitional clustering algorithms 481–482
 pre-processing similarity data 494
 reference set 492
 similarity measures 493
 spectral clustering algorithms 483–484
 Gene functional annotation, current state
 assessment of 226–227
 Gene function identification 555
 See also Biological network inference and analysis
 Gene functions, databases for 464
 Gene–gene similarity 8
 Gene network model
 functional bias effects on 463–465
 circumventing functional bias 472
 functional annotations 465–468
 genomics data set functional bias 470–472

 reference set functional bias 468–470
 Gene ontology (GO) 465, 466
 annotations, for yeast gene products 8
 biological process annotations 467
 evidence codes 466
 hierarchy in 472–473
 Gene regulatory networks, data collection
 and modeling in 338–340
 bottom-up modelling, experimental design for 342
 building of 344–347
 decouple key components 343–344
 mutations effects on 344
 network subset isolation 343
 in silico experiments, model predictions in 347–348
 model organisms for 337–338
 phage λ , promoter regulation in
 computational modeling of data 350–353
 experimental design in 348–350
 prokaryotic promoters, measurement of 340–341
 Gene-specific model 358
 Genetic regulatory networks 182
 Genome-wide data generation, experimental
 methods for 228
 physical interaction, measurement methods of 229–230
 regulation and predictive networks creation,
 methods for
 array data, pathway inference and
 modeling of 232–237
 co-regulated genes identification,
 cluster analysis in 230–232
 gene expression and genetics 237–239
 Genome-wide genetic mapping, usages 238
 GGB, *see* Gibson-Bruck method
 Gibbs sampling algorithm 5
 Gibson-Bruck method 326
 Gillespie method 271
 Glycine biosynthesis pathway 375
 GO, *see* Gene ontology
 Graph-based clustering algorithms, *see* Pairwise
 clustering algorithms
 Graph Merge plug-in 9

H

Halobacterium NRC-1 195
 Heat-cool-cycle 32–33
Helicobacter pylori 146
 Hierarchical clustering algorithms 482–483
 See also Cluster algorithms
 Hierarchical clustering, of expression data 497–498
 See also Cluster algorithms
 High blood glucose (HBG) 193
 High-throughput protein–protein interaction,
 measurements of 229
 Histidine kinase 434

HK, *see* Histidine kinase
HMMER algorithm 516
Homo sapiens..... 108, 112–113
HPRD, *see* Human Protein Reference Database
Human Protein Reference Database 69, 514

I

ID3 learning algorithm, characteristics of 385
IDA, *see* Inferred from direct assay
I/GDP-mannose metabolism 375
IGI, *see* Inferred from genetic interaction
IMAGE 33
Immunoprecipitation 203
IMP, *see* Inferred from mutant phenotype
Inferelator tutorial..... 197–201
Inferred from direct assay 465
Inferred from Electronic Annotation (IEA) 8
Inferred from genetic interaction 465
Inferred from mutant phenotype 465
Inferred from physical interaction 465
Information gain, definition of 385
Information impurity, definition of 384
In silico experiments, model predictions in 347–348
 See also Gene regulatory networks, data collection
 and modeling in
Integrator, in network visualisation 524
Interolog 128
Interolog method and BIND data 514–515
Intersection network construction 9–10
Introns evolution, in eukaryotes 357
 bayesian estimation technique in 358–359
 expectation-maximization algorithm 359
 maximum likelihood estimation
 techniques in 359
 materials in 360–361
 methodologies in 361–368
 properties of 360
IPI, *see* Inferred from physical interaction
Iterative clustering method, of noisy data 485
 See also Cluster algorithms
Iterative clustering, of expression data 495–496
 See also Cluster algorithms
IUB Enzyme Commission 376
IUBMB-Sigma-Nicholson Metabolic Pathways 228

J

Java and CABIN 559
Java Runtime Environment (JRE) 4
JavaScript Object Notation 526
Jensen-Shannon measure 389
 λ -JS divergence, definition of 389
JSON, *see* JavaScript Object Notation
Jump Markov process 319

K

K connections 172
KEGG, *see* Kyoto Encyclopedia of Genes and Genomes
KEGG2 database 227
Kidney disease (KD) 193
Kinetic measurement 203
Kinetic modeling, of biological systems
 discrete stochastic simulation,
 acceleration of 320–324
 historical perspectives of 311–315
 network size, scaling properties of 326–327
 simulation methods for 316–319
 software tools for 328–330
 SSA fundamentals 319–320
 thermodynamic limit 324–325
Klemm-Eguíluz model 156–157
K-means algorithm 481–482
 See also Cluster algorithms
K-means clustering algorithm, for expression
 data 498–500
 See also Cluster algorithms
kNN algorithm 82–83
Knowledge-based network, definition of 226
Kyoto Encyclopaedia of Genes and Genomes 466, 467

L

L1-shrinkage 207
lacZ gene 340
lacZ transcriptional fusions, usage of 341
Langevin leaping formula 325
Laplace transform approach 291
LASSO 207–208
Learning algorithm, optimization of 390–392
Learning system, parts of 381
Leukemia-related BCR-ABL pathway,
 analysis of 261
L-glutamine 156
Lifestyle-based network similarity
 index (LSI) 174–175, 178
Ligand-receptor pairs, co-expression 257–261
Linear perturbation analysis
 control coefficients 278–279
 elasticities and control coefficient,
 relationship 279–281
 elementary processes 276–277
 See also Biochemical control theory (BCT),
 for biochemical network
Linux 4–5
 and python 534
LLS, *see* Log-likelihood score
Locally homogeneous approach, importance of 314–315
Log-likelihood score 468, 469

M

Machine-learning algorithms, importance of..... 470
 Markov chain algorithm..... 251
 Markov chain model, in intervention analysis..... 256–257
 Markov-chain Monte Carlo procedure..... 5
 MATLAB, usage of..... 559
 Matrix2png..... 169, 177
 Maximum likelihood estimation techniques,
 in intron evolution..... 359
 MCODE Cytoscape plug-in..... 10, 15
 MCODE plug-in for Cytoscape..... 4
 MDL, *see* Minimum description length
 MDL principle, model selection in..... 500
 Metabolic Control Analysis, *see* Biochemical
 Systems Theory
 Minimum description length..... 410, 486
 MIPS, *see* Munich Information Center for Protein
 Sequences
 Mirror-tree co-evolution-based algorithms..... 66
 Mismatch/Novel kernel, function of..... 379
 Modularity, concept of..... 145
 Module-replacement approach..... 344
 See also Gene regulatory networks, data collection
 and modeling in
 Molecular pathway, *see* Biochemical pathways, definition of
 Molecule_by_name API method, confirmation of..... 527
 Molecule_sequence table and sequence data..... 541–542
 See also Bioverse
 Monte Carlo (MC)
 algorithm, usage of..... 430–432
 simulation..... 254, 313
 mRNA expression
 euclidean metric in..... 493
 SEBINI and..... 564
 MSAs, *see* Multiple sequence alignments
 M τ L method, *see* Multinomial tau-leaping method
 “multi-copy” representation..... 31
 Multinomial tau-leaping method..... 323
 Multiple sequence alignments..... 433
 Multi-scale chemically reacting systems simulation,
 hybrid methods in..... 330–331
 Munich Information Center for Protein
 Sequences..... 470, 514
Mus musculus..... 108
 Mutation analysis, in protein..... 451
 See also Protein
 Mutual information, computation of..... 429
 See also Specificity determining residues

N

Name mapping..... 111–112
 NCBI GenBank Identifiers (GIs)..... 109, 131
 role of..... 514

Negative design constraints..... 442
 Negative feedback oscillators..... 300–302
 See also Biochemical control theory (BCT),
 for biochemical network
 Network import and CABIN..... 561
 See also Collective Analysis of Biological Interaction
 Networks
 Network inference and SEBINI..... 560
 See also Software Environment for Biological
 Network Inference
 Network motifs..... 170
 Newton-Raphson algorithm..... 368
 Normalization..... 203
 Normalized cut, definition of..... 484
 Nucleic acid..... 5

O

Oak Ridge National Laboratory..... 567
 ODEs, *see* Ordinary differential equations
 OLAP, *see* Online analytical processing
 OLTP, *see* Online transaction processing
 OMIM database and disease-specific proteins..... 453–454
 Online analytical processing..... 536–537
 Online Predicted Human Interaction Database..... 451
 Online transaction processing..... 536
 Operon..... 202
 OPHID, *see* Online Predicted Human Interaction
 Database
 Oracle DBMS sqldr and protein interactions..... 452
 See also Protein
 Ordinary differential equations..... 270, 312, 345
 ORNL, *see* Oak Ridge National Laboratory
 Orthologs..... 128, 168
 Orthology detection procedure, of network
 reconstruction..... 167
Oryza sativa..... 108, 114–115
 Oscillatory systems study, in biochemistry..... 300–304
 See also Biochemical control theory (BCT),
 for biochemical network

P

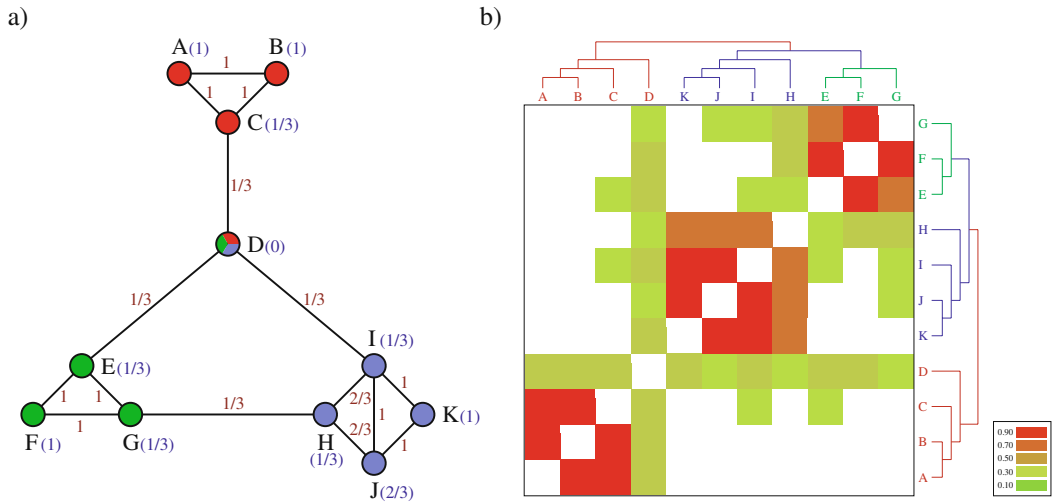
Pairwise clustering algorithms..... 482
 Paralogous pairs..... 76
 Particle transport dynamics, role of..... 314
 Partitional clustering algorithms..... 481–482
 See also Cluster algorithms
 Pathway hole, definition..... 374
 Pathway motifs, linear analysis
 branched pathways..... 283–285
 cyclic systems..... 285–287
 negative feedback of..... 287–290
 straight chains..... 281–283
 See also Biochemical control theory (BCT),
 for biochemical network

- Pathway prediction, role 374
- PathwayPro algorithm 251, 254
- biological applications of
 - disease genes identification and drug targets of
 - leukemia 261–265
 - ligand–receptor pairs, co-expression of 257–261
- PDB, *see* Protein Data Bank
- Pearson correlation
- coefficient, role of 250, 468
 - for mRNA expression 493
- Pearson correlation coefficient (PCC) values 7, 13–15
- Perl programming language 5
- Pfam classification test, for protein domains 393–397
- Phage λ
- bistable switch 338
 - promoter regulation in
 - computational modeling of data 350–353
 - experimental design in 348–350
 - See also* Gene regulatory networks, data collection and modeling in
- Physical interaction network, definition of 226
- Physical models, construction of 235
- Poisson random variable, definition of 322
- Poisson τ L method 322
- Position-specific scoring matrix 455
- Position weight matrix (PWM) 25
- Post-genomic research, challenges in 249–250
- PostgreSQL database 558
- PPI, *see* Protein–protein interaction
- Predictor 204
- P_{RM} promoter, autoregulation of 349
- Probabilistic Boolean network algorithm 251
- Probabilistic model, in intron evolution 361–363
- See also* Eukaryotic gene structure
- Probability based MDL postpruning, in model
- assigns 415
- Probability weighted dynamic Monte Carlo
- method 321
- Prokaryotes 112, 116
- Prokaryotic promoters, measurement of 340–341
- See also* Gene regulatory networks, data collection and modeling in
- Propensity function, definition of 319
- Protein
- bioverse in 516–518
 - classifications of 381
 - data visualization 458–459
 - and DNA interactions, measurement of 229
 - evolution, model of
 - conservation and variation, sources of 425
 - genomic proximity in 426
 - protein duplication, consequences of 425–426
 - family, information content of features in 397–400
 - feature selection in 457–458
 - identification of disease-specific 453–454
 - interactions
 - data integration 452–453
 - data, steps of 451–452
 - genetic variation and 450
 - mutation analysis in 451
 - and ligand interactions 421–422
 - sequence, function prediction in 376–377
 - SNP function prediction 455
 - SVM model 455–457
 - See also* Enzyme class prediction
- Protein Data Bank 514
- Protein domains and family, public databases 134–136
- protein–protein BLAST (BLASTP) 117–118
- Protein–protein interaction 553
- Protein–protein interactions, co-evolution model
- basic model
 - break-up of interactions by evidence codes 69–71
 - co-evolution algorithm 67
 - computing correlated divergence 67–72
 - construction of data set of non-interacting
 - proteins 72
 - construction of multiple sequence alignments 72
 - correlated approach 66
 - Pearson correlation coefficient, drawbacks 74–76
 - signals of correlated divergence 72–73
 - co-evolution approaches 64–65
 - hybrid approaches 65
 - improved model
 - organism subset selection 81–83
 - protein subset selection 76–81
 - relational data mining inference 63–64
 - structure-based approaches 62–63
- Protein–protein interactions, computational reconstruction
- BioGRID method 96–97
 - decision tree analysis 94
 - effect of size and bias in experimental interactome
 - maps 95–96
 - genomic features 91–92, 95
 - gold-standard positive and negative
 - data sets 91, 94–95
 - logistic regression analysis 92–93
 - Naïve Bayes assumption 92
 - nonlinear continuous and graph-based
 - features 93–94
 - random forest analysis 94
 - regularization 93
 - support vector machine (SVM)
 - and Adaboosting 93
- Protein–protein interactions, prediction
- materials 44–46
 - methods
 - domain combinations 46
 - high quality interactions 53–57

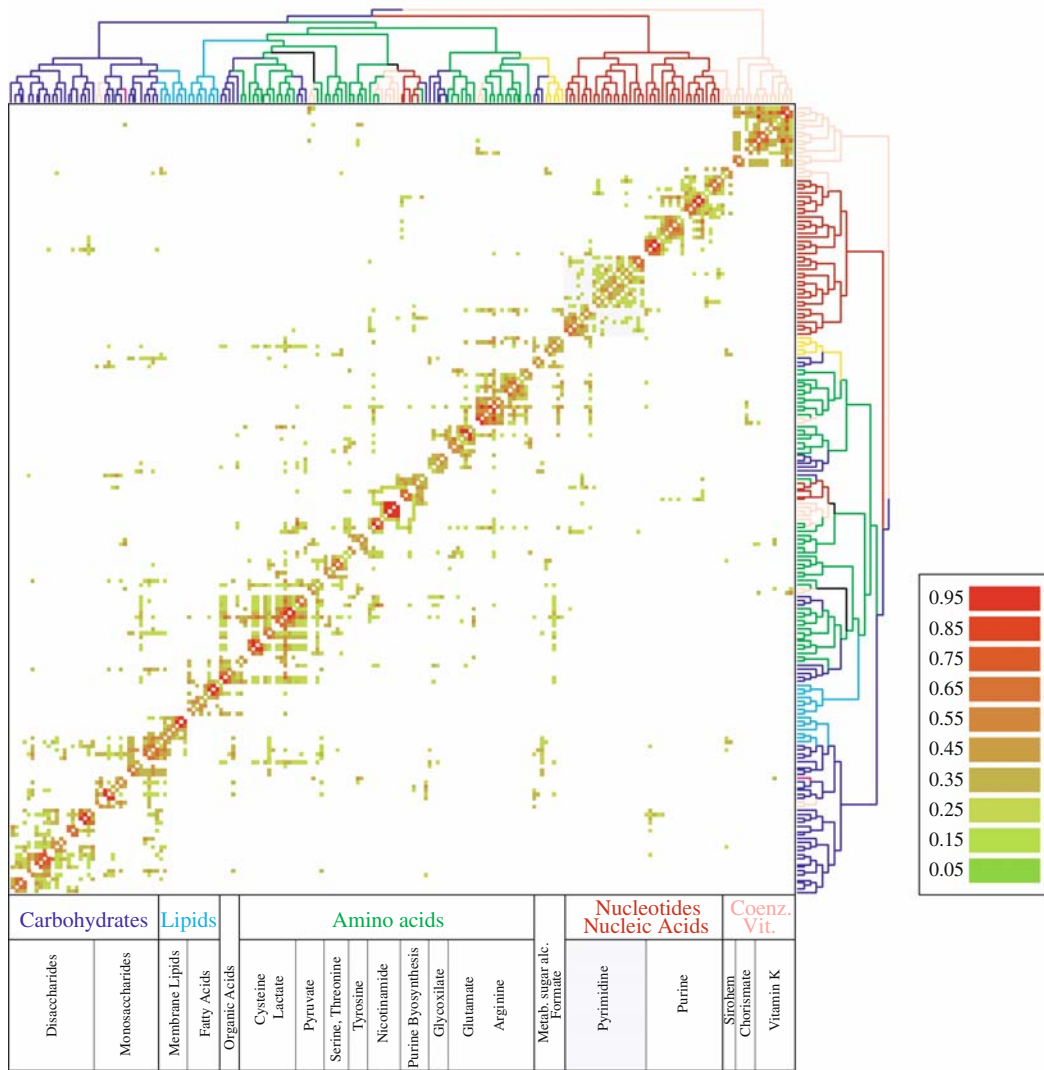
- MSSC algorithm 51–53
 MSSC problem 49–51
 of prediction 47–48
 protein cover problem 46
 set-cover problem 48–49
- Protein subcellular localization
 experiments, database 133
 sources 134
- ProteoLens database and protein interactions 458
See also Protein
- Protinfo server and protein structure 518
- PSI-BLAST 85
 algorithm 516
 database and SVM 455–456
 role of 379
- PSI-MI database 539
- PSIPRED algorithm 516
- PSSM, *see* Position-specific scoring matrix
- P τ L methods, *see* Poisson τ L method
- p*-value
 index and k-means algorithm 500–501
 pathways in 505
- PW-DMC method, *see* Probability weighted dynamic
 Monte Carlo method
- Python programming language
 API in 527
 operating system of 534
 XML-RPC code in 526
- R**
- Random access memory (RAM) 4
- Rattus norvegicus* 108
- Reaction rate equations 312
 development of 318
- Receiver operating characteristic 389, 472
- Reciprocal Best BLAST Hit (RBH) algorithm 80
- γ -Recursion, computation of 365
- Recursive feature elimination 457
- Red Hat Linux 558
- RegulonDB 104, 108
- Relaxation oscillators 302–303
 See also Biochemical control theory (BCT), for
 biochemical network
- REMORA web services, in bioinformatics tools 536
- Response regulator domain 434
 Ala-Scan of 441
- RFE, *see* Recursive feature elimination
- Rhodospseudomonas palustris*
 CABIN role in 561
 protein-protein interaction in 565, 568
- RMA, *see* Robust Multi-Array Analysis
- Robust Multi-Array Analysis 258
- ROC, *see* Receiver operating characteristic
- Routh-Hurwitz criterion 297
- Routh-Hurwitz table, advantage of 297
- RR, *see* Response regulator domain
- RREs, *see* Reaction rate equations
- S**
- Saccharomyces cerevisiae* (Yeast) 3, 108–109,
 111, 114–115, 130, 146, 227, 463, 465
- Saccharomyces cerevisiae* Genome Database 227, 514
- Saccharomyces* Genome Database (SGD) 8, 11
- SAM model, in protein domains 402
- SBML, *see* Systems Biology Markup Language
- SCOP, *see* Structural Classification of Proteins
- SCPD 104, 106
- SDRs, *see* Specificity determining residues
- SDT, *see* Stochastic decision trees
- Search query, molecules in 522
 See also Web application
- SEBINI, *see* Software Environment for Biological
 Network Inference
- SELEX 24
- Sequence analysis
 importance of 376–377
 problems in 231
- SEQUEST peptide mass spectra
 analysis algorithm 567
- SGD, *see* *Saccharomyces cerevisiae* Genome Database
- SGD 267, ORFs verification in 227
- SHAKE 30, 33
- Shrinkage parameter 209
- Simple genetic circuits 292
 cascade of 294
 low-pass frequency response of 293
- Simple interaction file (SIF) 7, 8
- Simple linear reaction chains, frequency
 analysis of 292–294
- Simple negative feedback, frequency
 analysis of 294–295
- Simulation algorithm, for steady-state analysis 256
- Size Scalable Model 326
- Smith-Waterman algorithm 118
- SNP function prediction, in protein 455
 See also Protein
- Software architecture 556–560
- Software Environment for Biological Network
 Inference 551
 advantages of 553–554
 analysis flow of control 563–569
 architecture in 556–559
 and CABIN system 559
 capabilities in 560–563
 inference algorithms and 553
 network subsets and 562
 role of 552
- Specificity-determining amino acids 425

- Specificity determining residues
 in ATPase and DD domains 437–440
 in class Ia aaRSs 443–446
 conservation and variation, sources of 425
 genomic proximity in 426
 identification methods 422–424
 modeled structures 440–441
 procedures in determining 426
 evolution pattern of 427
 multiple alignments and bacterial
 two-component system alignment 428–429
 mutual information calculation 429
 ortholog detection and clustering 427–428
 statistical significance estimation 430–433
 protein duplication, consequences of 425–426
 in RR domain 435–437
- Spectral clustering algorithms 483–484
 See also Cluster algorithms
- Spectral clustering, of expression data 496–497
 See also Cluster algorithms
- SPEL method 424
- SPR, *see* Surface patch ranking
- SQL, *see* Structured Query Language
- SSAs, *see* Stochastic simulation algorithms
- SSM, *see* Size Scalable Model
- Steady-state solution, for biochemical network 275
- Steepest Descent 30
- Stochastic approach, for cellular networks 271
- Stochastic decision trees 379
 for protein modeling
 evaluation of 388–390
 framework of 387–388
- Stochastic differential equation 325
- Stochastic simulation algorithms 314, 317, 319
 development of 320
 fundamentals of 319–320
 future perspectives of 327, 330–332
 steps of 319–320
 See also Biological systems, kinetic modeling of
- Stoichiometric networks, for biochemical network
 analysis 271–275
- Strict clustering
 of expression data 495–496
 of noisy data 484–485
 See also Cluster algorithms
- Structural Classification of Proteins 514
 database and enzyme 377
- Structured Query Language 537
- Subset of network, isolation of 343
 See also Gene regulatory networks, data collection
 and modeling in
- Summation theorem 289
- Sum of pairs (SOP) function 78, 82
- Supervised learning
 genomics data set functional bias on 470–472
 reference set functional bias on 468–470
 See also Gene network model
- Supervised learning technique, importance of 464
- Support vector machine 379, 424, 455–456
- Surface patch ranking 424
- SVM, *see* Support vector machine
- Swiss-Prot database, extraction of SNP data
 and mutation 453
- SWISS-PROT functional assignments 424
- Swiss-Prot mutations and protein interactions 450
- SynTreN network and SEBINI 563
- Systems Biology Markup Language 538–539
- ## T
- TAS, *see* Traceable author statement
- Tau-leaping and PW-DMC methods,
 difference of 321–322
- Term-term similarity 8
- TF, *see* Transcription factor
- Thermodynamic equilibrium, for biochemical
 network 275
- TIGR Rice Genome Annotation database 108
- TIP3P 29
- TP53_HUMAN protein 454
- Traceable author statement 465
- Traceable Author Statement (TAS) 8
- Trajectory analysis 317
- Transcriptional regulation networks, global models
 challenges 188–190
 design principle characteristics 187–188
 methods
 Bayesian network approaches 193–195, 204
 ordinary differential equations (ODEs) 192
 summary of the ten baseline variables 207
 using Inferelator algorithm 195–202
 properties of good solution 190–191
 review of functional-genomics
 direct assays of protein–DNA interactions 184–185
 functional annotations 185–186
 microarrays 183–184
 protein–protein interactions 186
 systems biology data quality 186–187
- Transcriptional regulation networks, reconstruction
 and comparison
 computational resources
 genome sequences 176
 hardware 175
 software 175
 templates 175–176
 procedure
 analysis of conserved regulatory interactions 170
 analysis of global network structure 172–173
 analysis of local network structure 170–172

attributes of lifestyle class of an organism ..	173–175
creation of random networks	168–169
gene analysis and regulatory interactions....	169–170
network reconstruction procedure	166–168
structure	166
Transcription factor	229
Transcription factor–binding sites (TFBSs).....	3
Transcription factor–binding sites, prediction	
benchmarking	121–127
computational costs and monetary equivalent.....	34–35
computational resources.....	27–28
conversion into sequence logo	26–27
determination of	24
homology-based.....	103
method	
heating and equilibration of structure	30–31
introduction of counter ions	30, 37
role of water in binding of protein	
and DNA	29–30, 37
simulation of “multi-copy”	
structures	31–33, 37–39
simulation protocols for free energy	
calculations.....	31–33
simulation protocols for protein–DNA	
complex	28–31
systematic and statistical errors.....	39–40
mutational free energies, calculation of	25
Python script for	120–121
regulatory protein–DNA and protein–protein	
interactions	
additional data sets.....	116–117
data sets.....	104–116
determination of similarity among protein	
sequences.....	117–120
procedures	103–104
relative binding free energy, determination of	24–25
similarities in source organism and target	
organism.....	102
sources of experimental.....	105
TRANSFAC [®] 7.0	104–106, 111, 116, 129, 132–133
Transformation	203
Transient state, for biochemical network	275
TRIPLES database.....	116
U	
UbiA family, tree for.....	396
UCSC Genome Browser	130
UniProt Knowledgebase and OPHID proteins	452
UNIX workstation	4–5
Urinating frequently (UF)	193
V	
Variable Connectivity Model	326
VCM, <i>see</i> Variable Connectivity Model	
W	
Watson–Crick base pairs.....	25
Web application	518–523
WormBase	104, 107–108
WormMart	104, 108
X	
XML-RPC protocol, in bioverse API.....	526
<i>See also</i> Application Programming Interface	
Y	
YBR059C gene	218
Yeast cell-cycle gene co-expression, and GO intersection	
network	9
Yeast galactose expression data, mapping of	234
Yeast galactose-utilization pathway, cluster	
analysis in	232
yFiles Layouts plug-in	10
YJR123W gene	217

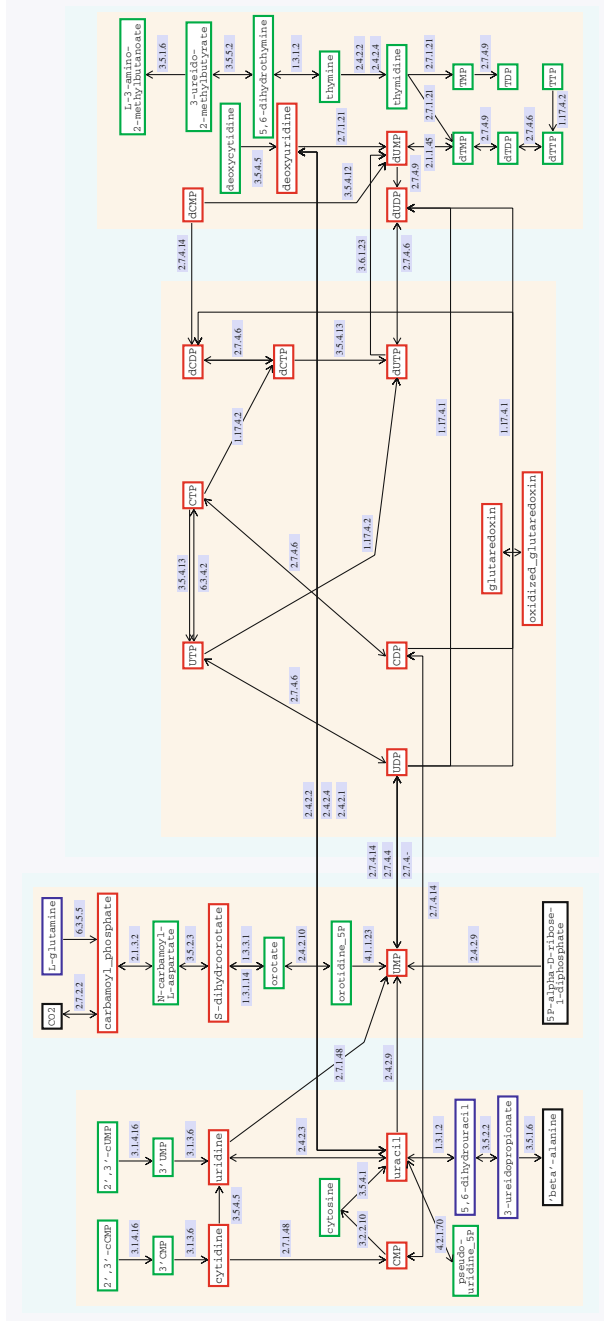
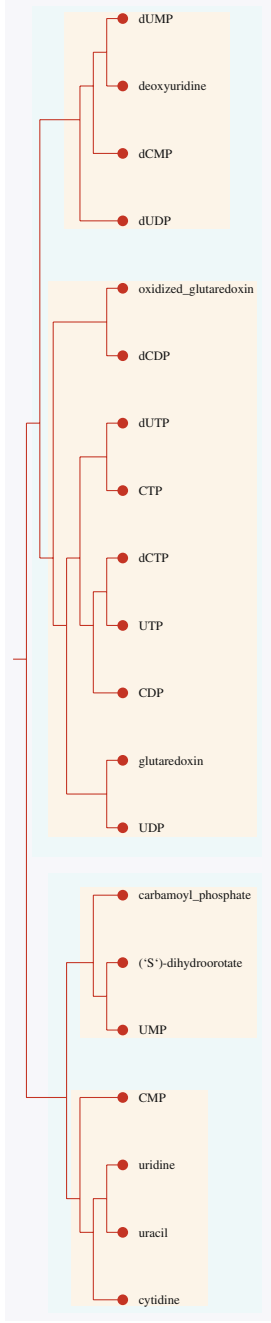


Color Plate 1. Uncovering the underlying modularity of a complex network. **(a)** Topological overlap illustrated on a small hypothetical network. On each link, we indicate the topological overlap for the connected nodes; and in parentheses next to each node, we indicate the node's clustering coefficient. **(b)** The topological overlap matrix corresponding to the small network shown in **(a)**. The *rows and columns* of the matrix were reordered by the application of an average linkage clustering method to its elements, allowing us to identify and place close to each other those nodes that have high topological overlap. The *color code* denotes the degree of topological overlap between the nodes. The associated tree reflects the three distinct modules built into the model, as well as the fact that the EFG and HIJK modules are closer to each other in the topological sense than to the ABC module (Chapter 7, Fig. 3; *see* discussion on p. 151).

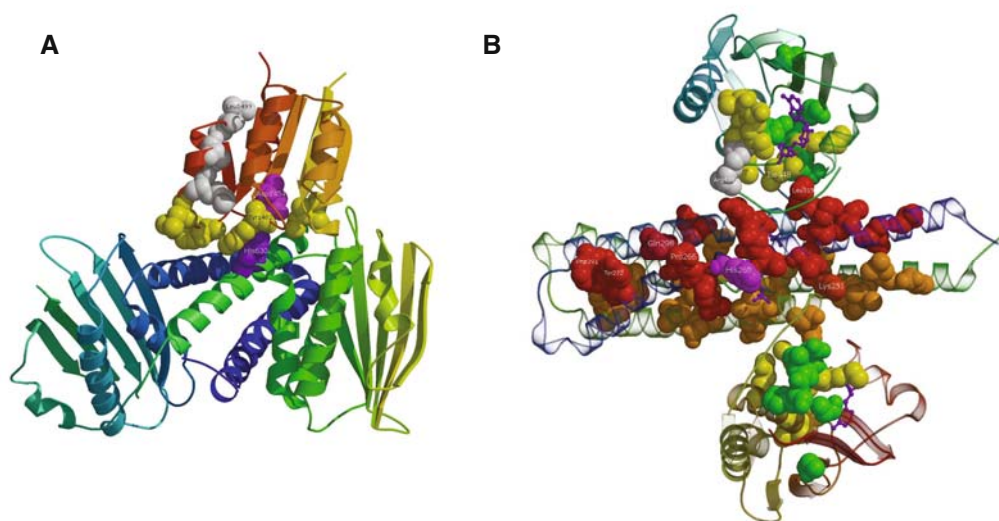


Color Plate 2. Topological modules in the *Escherichia coli* metabolism: the topologic overlap matrix, together with the corresponding hierarchical tree (*top and right*) that quantifies the relation between the different modules. The branches of the tree are *color-coded* to reflect the predominant biochemical classification of their substrates. The *color code* of the matrix denotes the degree of topological overlap shown in the matrix. The large-scale functional map of the metabolism, as suggested by the hierarchical tree, is also shown (*bottom*) (Chapter 7, Fig. 5; *see* discussion on p. 153).

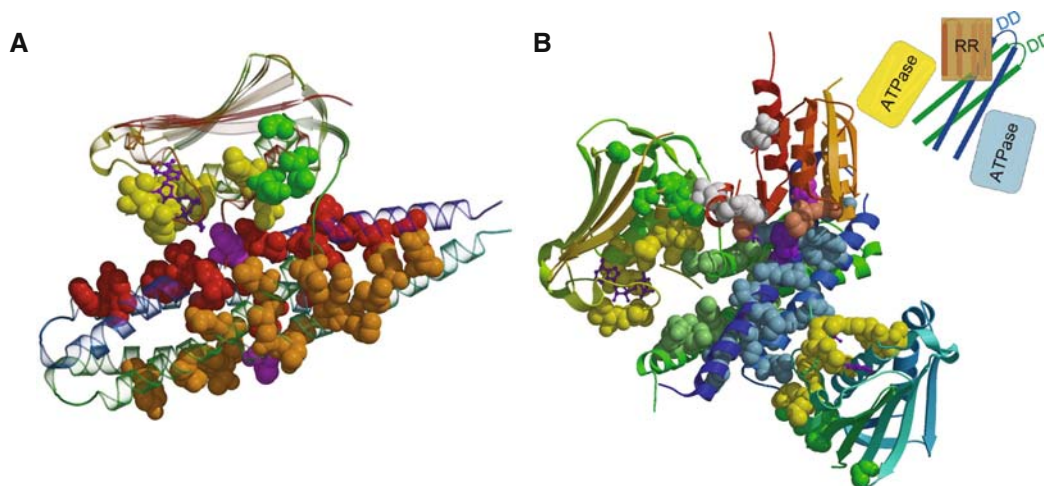
Pyrimidine Metabolism



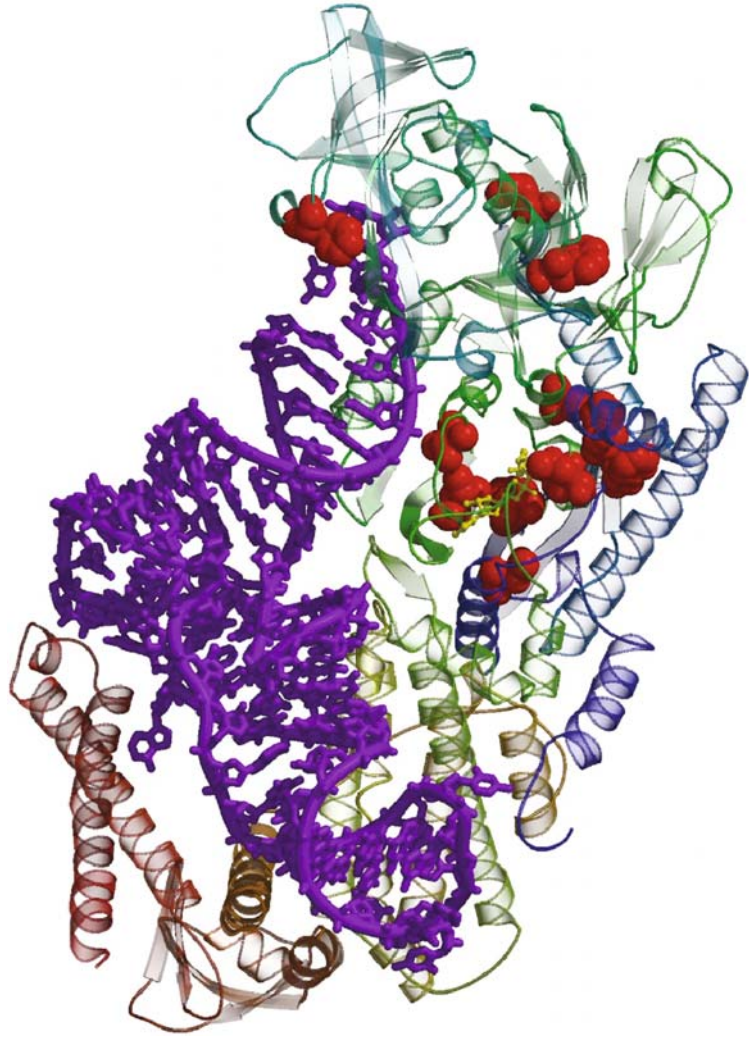
Color Plate 3. Enlarged view of the substrate module of pyrimidine metabolism, along with a detailed diagram of the metabolic reactions that surround and incorporate it. The colored boxes in the background denote the first two levels of nested modularity suggested by the hierarchical tree. Red-outlined boxes denote the substrates directly appearing in the reduced metabolism and thus on the tree (Chapter 7, Fig. 6; see discussion on p. 154 and full caption on p. 155).



Color Plate 4. Structural localization of putative SDRs and CERs in two-component system domains. (a) RR Spo0F (*red-brown ribbon*) bound to structural analog of the DD in Spo0B protein. The conserved His is shown in *purple*, the conserved Asp in RR in *magenta*. SDRs and CERs are shown in *yellow* or, when located on the $\alpha 4$ helix, in *white* (PDB entry 1F51). (b) The non-catalytic conformation of HK homodimer. ADP is shown as a *purple* wireframe, the phosphate-accepting conserved His residue in *magenta spacefill*. SDRs and CERs on the ATPase are shown in *yellow*, or in *white* if located on the unresolved ATP-lid loop that was superimposed from PhoQ kinase (PDB entry 1ID0), or in *green* in the RR-specific CERs side patch. SDRs and CERs on the DD are shown in *red* on one homodimer and *orange* on another (PDB entry 2C2A) (Chapter 18, Fig. 6; *see* discussion on p. 435).



Color Plate 5. Localization of putative SDRs and CERs on computationally obtained models (models provided by Marina et al (27)). (a) HK in the active conformation, the ATPase is docked on the DD so that transfer of the phosphoryl group is possible. SDRs and CERs on the ATPase domain are shown in *yellow* or *green* when located in the RR-specific CERs side patch. SDRs and CERs on the DD are shown in *red* on one homodimer and *orange* on another. (b) Spo0F computationally docked on HK and subsequently superimposed with RR from OmpR. RR (*brown-red ribbon*) (PDB entry 1KGS) with its $\alpha 4$ helix swung $\sim 90^\circ$; the phosphorylated Asp in the RR is shown in *magenta*, SDR and CERs are shown in *light red* or, when located the $\alpha 4$ helix in *white*. DD (*dark blue and dark green ribbon*): SDRs and CERs are shown in *light blue* on one dimer and in *light green* on another. ATPase (*yellow-green ribbon* on the left and *light-blue* on the right): the colors are the same as in (a) (Chapter 18, Fig. 7; *see* discussion on p. 439).



Color Plate 6. Valine aminoacyl-tRNA synthetase (PDB entry *1GAX*). The tRNA is shown as a *purple* wireframe structure, SDRs and CERs are *red* balls, and amino acid (valyl-adenylate analog) is in *yellow* wireframe (Chapter 18, Fig. 8; *see* discussion on p. 443).