

Springer Series in Statistics

Eric D. Kolaczyk

Statistical Analysis of Network Data

Methods and Models



Springer

Springer Series in Statistics

Advisors:

P. Bickel, P. Diggle, S. Fienberg, U. Gather, I. Olkin, S. Zeger

Springer Series in Statistics

- Athreya/Lahiri:* Measure Theory and Probability Theory
Bilodeau/Brenner: Theory of Multivariate Statistics
Brockwell/Davis: An Introduction to Time Series and Forecasting
Carmona: Statistical Analysis of Financial Data in S-PLUS
Chow/Teicher: Probability Theory: Independence, Interchangeability, Martingales, 3rd ed.
Christensen: Advanced Linear Modeling: Multivariate, Time Series, and Spatial Data; Nonparametric Regression and Response Surface Maximization, 2nd ed.
Christensen: Log-Linear Models and Logistic Regression, 2nd ed.
Christensen: Plane Answers to Complex Questions: The Theory of Linear Models, 2nd ed.
Cryer/Chan: Time Series Analysis, Second Edition
DasGupta: Asymptotic Theory of Statistics and Probability
Davis: Statistical Methods for the Analysis of Repeated Measurements
Dean/Voss: Design and Analysis of Experiments
Dekking/Kraaikamp/Lopuhaä/Meester: A Modern Introduction to Probability and Statistics
Durrett: Essential of Stochastic Processes
Edwards: Introduction to Graphical Modeling, 2nd ed.
Everitt: An R and S-PLUS Companion to Multivariate Analysis
Gentle: Matrix Algebra: Theory, Computations, and Applications in Statistics
Ghosh/Delampady/Samanta: An Introduction to Bayesian Analysis
Gut: Probability: A Graduate Course
Heiberger/Holland: Statistical Analysis and Data Display; An Intermediate Course with Examples in S-PLUS, R, and SAS
Jobson: Applied Multivariate Data Analysis, Volume I: Regression and Experimental Design
Jobson: Applied Multivariate Data Analysis, Volume II Categorical and Multivariate Methods
Karr: Probability
Kulkarni: Modeling, Analysis, Design, and Control of Stochastic Systems
Kolaczyk: Statistical Analysis of Network Data
Lange: Applied Probability
Lange: Optimization
Lehmann: Elements of Large Sample Theory
Lehmann/Romano: Testing Statistical Hypotheses, 3rd ed.
Lehmann/Casella: Theory of Point Estimation, 2nd ed.
Longford: Studying Human Populations: An Advanced Course in Statistics
Marin/Robert: Bayesian Core: A Practical Approach to Computational Bayesian Statistics
Nolan/Speed: Stat Labs: Mathematical Statistics Through Applications
Pitman: Probability
Rawlings/Pantula/Dickey: Applied Regression Analysis
Robert: The Bayesian Choice: From Decision-Theoretic Foundations to Computational Implementation, 2nd ed.
Robert/Casella: Monte Carlo Statistical Methods, 2nd ed.
Rose/Smith: Mathematical Statistics with Mathematica
Ruppert: Statistics and Finance: An Introduction

(continued after index)

Eric D. Kolaczyk

Statistical Analysis of Network Data

Methods and Models



Eric D. Kolaczyk
Department of Mathematics & Statistics
Boston University
111 Cummington St.
Boston MA 02215
USA

ISBN 978-0-387-88145-4 e-ISBN 978-0-387-88146-1
DOI 10.1007/978-0-387-88146-1

Library of Congress Control Number: 2009921812

© 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 (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.

Printed on acid-free paper

springer.com

To the ‘network’ that is my family . . .
 . . . those both near and far.

In Memorium

Br. Robert C. Ruhl, CSV

March 18, 1947 – Feb 2, 2008

“A teacher affects eternity; he can never tell where his influence stops.”
(Henry Brooks Adams)

Preface

In recent years there has been an explosion of *network data* – that is, measurements that are either of or from a system conceptualized as a network – from seemingly all corners of science. The combination of an increasingly pervasive interest in scientific analysis at a systems level and the ever-growing capabilities for high-throughput data collection in various fields has fueled this trend. Researchers from biology and bioinformatics to physics, from computer science to the information sciences, and from economics to sociology are more and more engaged in the collection and statistical analysis of data from a network-centric perspective.

Accordingly, the contributions to statistical methods and modeling in this area have come from a similarly broad spectrum of areas, often independently of each other. Many books already have been written addressing network data and network problems in specific individual disciplines. However, there is at present no single book that provides a modern treatment of a core body of knowledge for statistical analysis of network data that cuts across the various disciplines and is organized rather according to a statistical taxonomy of tasks and techniques. This book seeks to fill that gap and, as such, it aims to contribute to a growing trend in recent years to facilitate the exchange of knowledge across the pre-existing boundaries between those disciplines that play a role in what is coming to be called ‘network science.’

The book is written for students and researchers with a ‘mature’ knowledge of statistics and hence is intended not only for statisticians but also for people involved with network data in various other areas, like those mentioned above. Background in calculus and linear algebra and some reasonable foundation in statistics and probability are expected. Beyond that, I have attempted to build all necessary material as needed.

In an effort to reach this admittedly diverse audience successfully, I have aimed in each chapter to communicate the material in a manner that strikes an appropriate balance between concepts, on the one hand, and technical depth and rigor, on the other. It is expected that the interested reader will want – and, indeed, is encouraged – to pursue the relevant primary sources for details I may have chosen to omit. The book is in this sense intended to serve as an *entrée* to the larger literature. Copious use of references has been made throughout the book for this very purpose. In addi-

tion, the exercises at the end of each chapter provide further opportunities to explore some of the topics in greater depth. There are both analytical and computational exercises to be found, with the latter frequently designed to be fairly open-ended in nature, so as to encourage exploration. Finally, the methods and models presented herein are illustrated throughout the book with examples from a wide range of disciplines. I have found this overall approach to the pedagogy of the material to work well when I taught classes of precisely the diversity that I envision for the readership of this book.

The book itself would not have been possible without the help, feedback, and support of many. I thank my editor, John Kimmel, and the other folks at Springer, for their help and guidance throughout the publication process. The idea for the writing of this book arose while I was on sabbatical during the 2004-2005 academic year, in the Laboratoire d’Informatique Algorithmique, Fondements, et Applications (LIAFA), at l’Université Paris 7, courtesy of CNRS, and in the Department of Statistics, at Harvard University. My own work in this area has been generously supported by grants from the United States National Institutes of Health (NIH), National Science Foundation (NSF), and Office of Naval Research (ONR). Students and colleagues attending the courses I taught on this material, at Harvard University in Spring 2005 and at Boston University in Fall 2005, are gratefully acknowledged for their interest, questions, and hard work. Many of the datasets used in this book were generously shared by colleagues from around the world, including Ignacio Alvarez-Hamelin, Mark Coates, Mark Crovella, Tim Gardner, Sucharita Gopal, Boris Hayete, Mark Kramer, Emmanuel Lazergas, Naoki Nariai, Robert Nowak, Xiaoyu Jiang, and Fabien Viger. In addition, Ignacio Alvarez-Hamelin, Kevin Boyack, Ulrik Brandes, Rui Castro, Sucharita Gopal, and Mark Kramer were kind enough to produce and share some of the more stunning figures found herein. For graciously responding to my unabashed solicitations for feedback on material in various chapters, I am indebted to Ignacio Alvarez-Hamelin, Alain Barrat, Ulrik Brandes, Tom Britton, Hugh Chipman, Mark Crovella, Tim Gardner, Boris Hayete, Peter Hoff, David Hunter, Xiaoyu Jiang, Simon Kasif, Naoki Nariai, Robert Nowak, Pip Pattison, Mike Rabbat, Garry Robbins, Martin Steffen, Shu Yang, and Ji Zhu. Special thanks in this regard are due to Joe Whittaker, whose comments were extensive and invariably helpful. Notwithstanding all of this feedback, however, any typos, mistakes, and other similar errors are of course my own. Andrej Mrvar, of the Pajek software team, is also to be thanked for his help in responding to my numerous questions. Finally, although surely inadequate in measure, I offer my heartfelt thanks to my wife, Josée Dupuis, not only for her love, patience, and support over the many months that I was submerged in the writing of this book, and for her reading of the final manuscript, but also for serving as my **R** guru, fielding even my many late-night queries when she would surely have rather preferred to continue uninterrupted with her own work!

Eric D. Kolaczyk
Boston, Massachusetts
March, 2009

Contents

1	Introduction and Overview	1
1.1	Why Networks?	1
1.2	Examples of Networks	3
1.2.1	Technological Networks	3
1.2.2	Social Networks	5
1.2.3	Biological Networks	7
1.2.4	Information Networks	9
1.3	About this Book	11
2	Preliminaries	15
2.1	Background on Graphs	15
2.1.1	Basic Definitions and Concepts	16
2.1.2	Families of Graphs	18
2.1.3	Graphs and Matrix Algebra	20
2.1.4	Graph Data Structures and Algorithms	21
2.2	Background in Probability and Statistics	24
2.2.1	Probability	25
2.2.2	Principles of Statistical Inference	31
2.2.3	Methods of Statistical Inference: Tutorials	32
2.3	Statistical Analysis of Network Data: <i>Prelude</i>	42
2.4	Additional Related Topics and Reading	45
	Exercises	45
3	Mapping Networks	49
3.1	Introduction	49
3.2	Collecting Relational Network Data	50
3.2.1	Measurement of System Elements and Interactions	51
3.2.2	Enumerated, Partial, and Sampled Data	54
3.3	Constructing Network Graph Representations	56
3.4	Visualizing Network Graphs	58
3.4.1	Elements of Graph Visualization	58

3.4.2	Methods of Graph Visualization	60
3.5	Case Studies	63
3.5.1	Mapping ‘Science’	65
3.5.2	Mapping the Internet	68
3.6	Mapping Dynamic Networks	74
3.7	Additional Related Topics and Reading	76
	Exercises	77
4	Descriptive Analysis of Network Graph Characteristics	79
4.1	Introduction	79
4.2	Vertex and Edge Characteristics	80
4.2.1	Degree	80
4.2.2	Centrality	88
4.3	Characterizing Network Cohesion	94
4.3.1	Local Density	94
4.3.2	Connectivity	97
4.3.3	Graph Partitioning	102
4.3.4	Assortativity and Mixing	111
4.4	Case Study: Analysis of an Epileptic Seizure	114
4.5	Characterizing Dynamic Network Graphs	116
4.6	Additional Related Topics and Reading	119
	Exercises	120
5	Sampling and Estimation in Network Graphs	123
5.1	Introduction	123
5.2	Background on Statistical Sampling Theory	126
5.2.1	Horvitz-Thompson Estimation for Totals	126
5.2.2	Estimation of Group Size	129
5.3	Common Network Graph Sampling Designs	131
5.3.1	Induced and Incident Subgraph Sampling	131
5.3.2	Star and Snowball Sampling	133
5.3.3	Link Tracing	136
5.4	Estimation of Totals in Network Graphs	137
5.4.1	Vertex Totals	137
5.4.2	Totals on Vertex Pairs	138
5.4.3	Totals of Higher Order	141
5.4.4	Effects of Design, Measurement, and Total	143
5.5	Estimation of Network Group Size	145
5.6	Other Network Graph Estimation Problems	149
5.7	Additional Related Topics and Reading	151
	Exercises	151

6 Models for Network Graphs	153
6.1 Introduction	153
6.2 Random Graph Models	154
6.2.1 Classical Random Graph Models	156
6.2.2 Generalized Random Graph Models	158
6.2.3 Simulating Random Graph Models	159
6.2.4 Statistical Application of Random Graph Models	162
6.3 Small-World Models	169
6.3.1 The Watts-Strogatz Model	169
6.3.2 Other Small-World Network Models	171
6.4 Network Growth Models	172
6.4.1 Preferential Attachment Models	173
6.4.2 Copying Models	176
6.4.3 Fitting Network Growth Models	178
6.5 Exponential Random Graph Models	180
6.5.1 Model Specification	180
6.5.2 Fitting Exponential Random Graph Models	185
6.5.3 Goodness-of-Fit and Model Degeneracy	187
6.5.4 Case Study: Modeling Collaboration Among Lawyers	188
6.6 Challenges in Modeling Network Graphs	191
6.7 Additional Related Topics and Reading	193
Exercises	195
7 Network Topology Inference	197
7.1 Introduction	197
7.2 Link Prediction	199
7.2.1 Informal Scoring Methods	201
7.2.2 Probabilistic Classification Methods	202
7.2.3 Case Study: Predicting Lawyer Collaboration	205
7.3 Inference of Association Networks	207
7.3.1 Correlation Networks	209
7.3.2 Partial Correlation Networks	212
7.3.3 Gaussian Graphical Model Networks	216
7.3.4 Case Study: Inferring Genetic Regulatory Interactions	220
7.4 Tomographic Network Topology Inference	223
7.4.1 Tomographic Inference of Tree Topologies	225
7.4.2 Methods Based on Hierarchical Clustering	228
7.4.3 Likelihood-based Methods	231
7.4.4 Summarizing Collections of Trees	234
7.4.5 Case Study: Computer Network Topology Identification	236
7.5 Additional Related Topics and Reading	241
Exercises	242

8 Modeling and Prediction for Processes on Network Graphs	245
8.1 Introduction	245
8.2 Nearest Neighbor Prediction	246
8.3 Markov Random Fields	249
8.3.1 Markov Random Field Models	249
8.3.2 Inference and Prediction for Markov Random Fields	252
8.3.3 Related Probabilistic Models	256
8.4 Kernel-based Regression	257
8.4.1 Kernel Regression on Graphs	258
8.4.2 Designing Kernels on Graphs	262
8.5 Case Study: Predicting Protein Function	266
8.6 Modeling and Prediction for Dynamic Processes	271
8.6.1 Epidemic Processes: An Illustration	272
8.6.2 Other Dynamic Processes	280
8.7 Additional Related Topics and Reading	281
Exercises	282
9 Analysis of Network Flow Data	285
9.1 Introduction	285
9.2 Gravity Models	287
9.2.1 Model Specification	288
9.2.2 Inference for Gravity Models	292
9.3 Traffic Matrix Estimation	297
9.3.1 Static Methods	298
9.3.2 Dynamic Methods	306
9.3.3 Case Study: Internet Traffic Matrix Estimation	310
9.4 Estimation of Network Flow Costs	316
9.4.1 Link Costs from End-to-end Measurements	317
9.4.2 Path Costs from End-to-end Measurements	321
9.5 Additional Related Topics and Reading	328
Exercises	330
10 Graphical Models	333
10.1 Introduction	333
10.2 Defining Graphical Models	334
10.2.1 Directed Graphical Models	335
10.2.2 Undirected Graphical Models	339
10.3 Inference for Graphical Models	342
10.4 Additional Related Topics and Reading	344
Glossary of Notation	345
References	347
Author Index	373
Subject Index	381

Chapter 1

Introduction and Overview

The systematic collection and analysis of data on networks of one form or another goes back at least to the 1930's in certain select areas of science, and in fact has subtle roots reaching back centuries further. However, during the decade surrounding the turn of the 21st century, network-centric analysis, as a general approach to scientific inquiry, has reached entirely new levels of prevalence and sophistication, with practitioners in fields now ranging from the physical and mathematical sciences to the social sciences and humanities. In this chapter we present a ‘birds-eye’ view of the area that is gradually coming to be known as ‘network science,’ starting with some background, continuing with a mosaic of examples, and finishing with a discussion of the organization and philosophy of this book.

1.1 Why Networks?

The oft-repeated statement that “we live in a connected world” perhaps best captures, in its simplicity, why networks have come to hold such interest in recent years. For example, from the ‘small world’ studies of Harvard sociologist Stanley Milgram [277] and later the play of Guare [188] comes the suggestion that we are each separated from any other person on the planet by at most six other people (i.e., ‘six degrees’). And this concept arose even before the Internet and related inventions like email, chat-rooms, and blogs! Similarly, we see constantly in the popular press examples of the inter-connectedness of various human institutions (e.g., governments) and processes (e.g., economies), and also of humans and natural systems (e.g., in regards to the impact of humans on climate and the environment).

The image of a network – that is, essentially, something resembling a net – is a natural one to use to capture the notion of elements in a system and their interconnectedness. Note, however, that the term ‘network’ seems to be used in a variety of ways, at various levels of formality. The *Oxford English Dictionary*, for example, defines the word *network* in its most general form simply as “a collection of interconnected things.” On the other hand, frequently ‘network’ is used inter-changeably

with the term ‘graph’ since, for mathematical purposes, networks are most commonly represented in a formal manner using graphs of various kinds. In an effort to emphasize the distinction between the general concept and its mathematical formalization, in this book we will use the term ‘network’ in its most general sense above, and – at the risk of the impression of a slight redundancy – we will refer to a graph representing such a network as a ‘network graph.’ The discussion of the remainder of this chapter is at the level of networks, while technical material throughout the rest of the book is generally developed in reference to a specific network graph(s).

The seeds of network-based analysis in the sciences, particularly its mathematical foundation of graph theory, are often placed in the 1735 solution of Euler to the now famous Königsberg bridge problem, in which he proved that it was impossible to walk the seven bridges of that city in such a way as to traverse each only once. Since then, particularly since the mid-1800’s onward, these seeds have grown in a number of key areas. For example, in mathematics the formal underpinnings were systematically laid, with König [235] cited as the first key architect. The theory of electrical circuits has always had a substantial network component, going back to work of Kirchoff, and similarly the study of molecular structure in chemistry, going back to Cayley. As the fields of operations research and computer science grew during the mid-1900’s, networks were incorporated in a major fashion in problems involving transportation, allocation, and the like. And similarly during that time period, a small subset of sociologists, taking a particularly quantitative view towards the topic of social structure, began developing the use of networks in characterizing interactions within social groups.

Presently, examples of network-based analysis may now be found far beyond the traditional areas listed above, involving topics ranging from computer networking and the Internet to biology and gene networks to library science and webs of knowledge. Two important contributing factors to this growth are (i) an increasing tendency towards a systems-level perspective in the sciences, away from the reductionism that characterized much of the previous century, and (ii) an accompanying facility for high-throughput data collection, storage, and management. The quintessential example is perhaps that of the changes in biology over the past 10 to 20 years, during which the complete mapping of the human genome, a triumph of computational biology in and of itself, has now paved the way for fields like systems biology to be pursued aggressively, wherein a detailed understanding is sought of how the components of the human body, at the genetic level and higher, work together.

The focus of this book is on the statistical analysis of *network data*. More specifically, we aim to present a core set of methods and models for the analysis of measurements that are either of or from a system conceptualized as a network. Such data are collected daily in a host of different areas. Each area, naturally, has its own unique questions and problems under study. Nevertheless, from a statistical perspective, there is a methodological foundation emerging, composed of tasks and tools that are each common to some non-trivial subset of research areas involved with network science. It is our goal here to present this foundation.

Much of the challenge in analyzing network data stems from the fact that they involve, either explicitly or implicitly, quantities of a *relational* nature. As such, measurements are typically both high-dimensional and dependent. Additionally, such data are often substantial in quantity, and thus computational tractability is generally an issue not far from the surface when developing and using statistical methods and models in this area. The study of data that are either high-dimensional, dependent, or massive in quantity each is in itself currently an important topic of research in statistical theory and methods. In the analysis of network data, all three are often present in a unique fashion.

1.2 Examples of Networks

In order to better appreciate the nature of the statistical foundation emerging in the analysis of network data, it is useful to have some initial sense of the contexts in which networks arise, the scientific questions being asked, and the measurements being taken. While many such examples are to be found throughout the rest of the book, we present here in this section an initial glimpse, meant to provide somewhat of a mosaic picture. For convenience, and following Newman [296], the presentation is organized loosely into four classes of networks: technological, social, biological, and informational. These divisions are intended to be soft, and not hard, as many networks can be said to fall into more than one category.

1.2.1 Technological Networks

Arguably the networks most familiar to us are those of a technological nature (i.e., human constructions consciously created in a network form). Examples include communication networks (e.g., telephone networks or the Internet), transportation networks (e.g., networks of roads or rails, or networks of airline routes), and energy networks (e.g., networks for delivery of electricity or gas, or electrical circuits).

Some or all of the topology of such networks is often known by some entity. For example, telephone service providers maintain knowledge of the lines they lay and manufacturers of electrical circuits begin with blueprints of circuit designs. Connectivity may be in the form of a literal physical tie, such as a fiber optic cable or a gas line, or in the form of a virtual connection, such as a wireless link between a cellular phone and a nearby tower or the travel of an airplane between the airports of two cities. Interest in these networks is often focused on the flow of some corresponding ‘commodity’ across the network, be it Internet traffic packets, freight carried by trains, or units of electrical energy.

Consider the rather celebrated example of the Internet, which is essentially a network of digital devices communicating over wired and wireless connections via a set of communication protocols. Starting from comparatively modest beginnings, as

a 100-node research network in the mid-1970s, the Internet today is effectively a network of networks, linking on the order of hundreds of millions of devices and responsible for carrying the communication traffic that underlies everything from electronic banking transactions to email correspondence to music, video, and gaming entertainment.

Figure 1.1 shows a visual representation of a portion of the Internet, at a certain level of granularity – the sub-network known as Abilene. Abilene is part of the Internet2 project,¹ a research project devoted to development of the ‘next generation’ Internet. It serves as a so-called ‘backbone’ network for universities and research labs across the United States in a manner analogous to the federal highway system of roads. The 11 large-scale ‘Core’ nodes in this network correspond to regional network aggregation points, connected by systems of optical transportation technologies and routing devices, denoted here as links. In addition, connected to each Core node are additional nodes, such as ‘Connector’ nodes and ‘Exchange Points.’ The Connector nodes are network infrastructures through which local ‘Participant’ networks from universities and research labs access Abilene; the Exchange Points are similar, but instead serve to integrate Abilene with other ‘Peer’ Networks, such as similar networks in other countries. Note the clear hierarchical structure, which also continues down into the Participant networks themselves, and is replicated in the Peer networks, ultimately descending down to the laptops and such sitting in people’s offices.

While most people largely take the Internet for granted, as a part of the infrastructure around which their daily lives are built, there is substantial interest in measuring and studying the Internet, in both the research and commercial communities. Network-oriented questions regarding the Internet tend to focus on those relating to its topology, the traffic it carries, the interaction of the two, and in turn the interaction of those with social and economic factors. For example, in regards to topology we may ask, “What does the Internet look like?” “How big is it?” and “What are its structural characteristics?” In terms of traffic, questions include “How much traffic is flowing across the network?” “How can I distinguish between ‘normal’ and ‘anomalous’ traffic?” and “Does my network have the capacity to meet anticipated demands?” See the book by Crovella and Krishnamurthy [105], for example.

In order to answer questions like these, measurements are taken in the Internet in a variety of active and passive manners. For example, it is possible to actively probe the Internet with small packets of traffic and register the responses that return to the sender as a result of communication protocols, which provides information on the routes the packets traveled and hence some insight into Internet topology. Conversely, it is also possible to ‘sit’ passively on an Internet link and monitor the traffic flowing by to a greater or lesser extent, depending on the granularity of information desired. We will see statistical topics of particular relevance to measurements like these in Chapter 3 (i.e., mapping networks), Chapter 5 (i.e., network sampling bias), Chapter 7 (i.e., Internet topology identification), Chapter 8 (i.e., spread of epidemics in a network), and Chapter 9 (i.e., analysis of network flow data).

¹ <http://www.internet2.edu/>

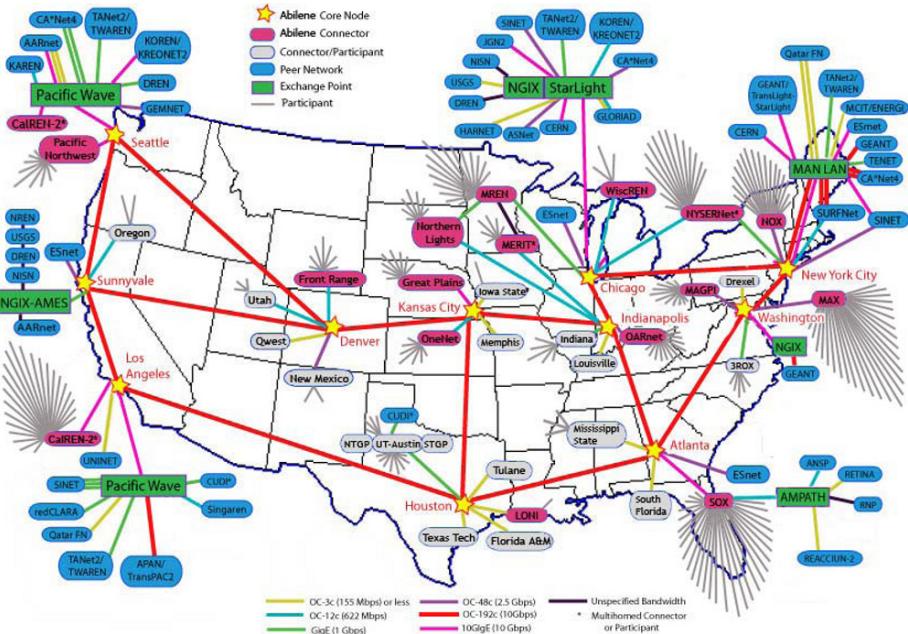


Fig. 1.1 Depiction of the Abilene network in the Internet. Different nodes represent various forms of network ‘entities’, while different colors of links indicate various levels of communication bandwidth. Note that some node names appear more than once, corresponding to the phenomena of ‘multi-homing’, wherein a given network connects to another at more than one location. Figure courtesy of Sucharita Gopal.

1.2.2 Social Networks

A class of networks with one of the longest histories of systematic study, dating back to at least the 1930’s, is that of social networks (i.e., networks representing the interactions among a collection of social entities or ‘actors’). Such entities typically are people or groups of people, but sometimes are non-human, such as animals. The type of interactions considered in this area varies and is constrained in part by the unit and nature of the social entities involved. Examples of social interactions include friendships among people, membership of people in larger social groups (e.g., clubs, companies, etc.), contacts between people (e.g., sexual contacts, meetings between members of terrorist cells, etc.), cooperation on a common endeavor, and the exchange of resources. Specific examples of social networks include networks of friendships among school children, sexual contacts within a community, corporate alliances among businesses, email exchanges between individuals, co-authorship on scientific articles, and trade agreements among nations.

The study of such networks is of particular interest to, and has traditionally been the province of, researchers in social sciences like sociology, anthropology, and psychology, although this interest is increasingly shared now by researchers in a number

of other areas, such as business and public health. The focus in these areas typically is on social structure and the quantitative characterization and analysis of such structure, which they aim to accomplish through the measurement of social interactions and the analysis of the resulting social network topologies. Questions of interest include “Who interacts with whom and what factors influence the tendency to interact?” “Which interactions are mutual?” “Are friends of friends also friends?” “What social groups, if any, exist in the network?”, “Who are the power brokers?” “Who is central to the network and who is peripheral?” and “Which actors are similar in the roles they play?”

Figure 1.2 shows a visual representation of a particular social network, the so-called ‘karate club network’ of Zachary [411]. Nodes represent members of a karate club observed by Zachary for roughly two years during the 1970’s, and links connecting two nodes indicate social interactions between the two members (thickness of the links reflects relative frequency of interactions). This dataset is somewhat unique in that Zachary had the curious fortune (from a scientific perspective) to witness the club split into two different clubs during his period of observation, due to a dispute between the head teacher and an administrator. It was originally published in conjunction with a model for information flow in small groups in the presence of conflict and fission. It has since become a favorite among researchers developing algorithms for detection of social subgroups, since the truth of membership in the two subgroups is a known quantity.

Social network scientists often face unique measurement challenges. Potential difficulties include the identification of social entities of interest, their willingness to be recruited into studies, and possible sources of bias in their response. For example, drug addicts or sex workers, two subpopulations of considerable interest to those studying the impact of social structure on the spread of the AIDS virus, are not necessarily obvious to identify in the general population. Once identified, they may have serious qualms about participating in a study. And once having agreed, in principle, to participate, they may be understandably reluctant to fully disclose information on, say, the sharing of hypodermic needles or their sexual partners. All of these issues, and others like them, can have an important impact on the statistical analysis of such data.

In recent years, the Internet has begun to have a fascinating impact on the field of social network analysis, due both to the potential for large-scale data acquisition and storage and the actual types of social interactions facilitated by the Internet. Examples of networks whose study is impacted in this manner include networks of email exchanges or phone calls, which can be measured by exploiting the existing computer infrastructure in the underlying technological networks, or networks of scientific collaborations, now relatively easily compiled on large scales because of the prevalence of electronic publication and archiving. Many of these networks are significantly larger than the size of typical social networks studied in the past. However, this characteristic can lead to its own issues. For example, the sheer magnitude of emails sent or calls made daily on a service provider’s network quickly leads to concerns about data volume, which can make even seemingly simple tasks of network summary and visualization highly nontrivial.

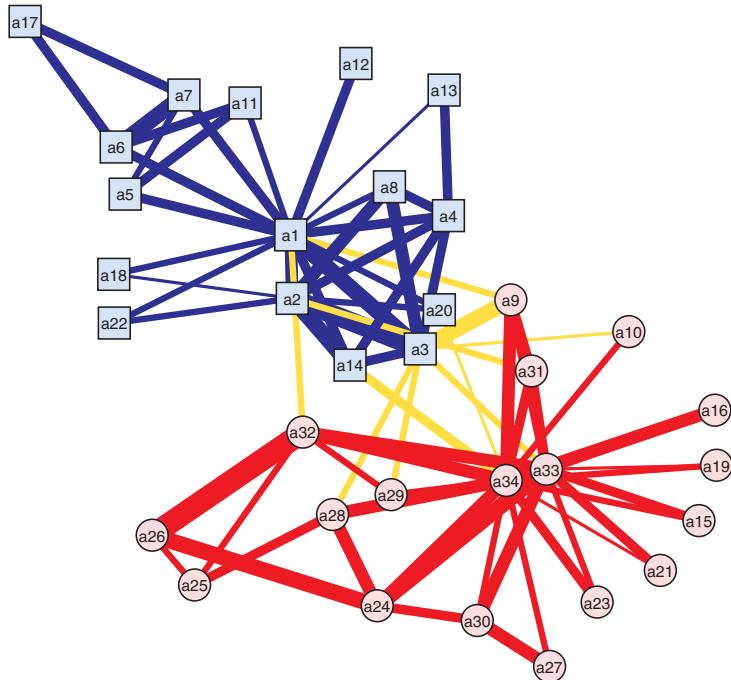


Fig. 1.2 Zachary's 'karate club' network. Subgroups, centered around actors 1 and 34, are indicated by the coloring and shape of their nodes, using blue squares and red circles, respectively. Links between actors within the same subgroup are colored similar to their nodes, while links between actors of different subgroups are shown in yellow.

We will encounter various social network tools and other topics of relevance to social network analysis in Chapter 3 (i.e., mapping networks), Chapter 4 (i.e., descriptive analysis of observed network structure), Chapter 5 (i.e., the sampling of network graphs), and Chapter 6 (i.e., the modeling of network graphs).

1.2.3 Biological Networks

Networks are a natural and commonly used tool for representing the internal workings of biological systems, at all different scales. For example, intra-cellular networks of interest include those describing the regulatory behavior among genes, the physical affinity for binding among proteins, the participation of metabolites together in biochemical processes, and combinations thereof. Similarly, a well-known example of an inter-cellular network is a network of neurons. On the other hand, networks describing interactions among complete organisms include ecological net-

works, such as those describing predator-prey relationships, and epidemiological networks, characterizing the spread of disease in a population.

As an illustration of a biological network, consider the network representation in Figure 1.3, which characterizes the cellular-level biomolecular process underlying the circadian clock mechanism in the organism *Drosophila melanogaster* (fruit fly). The ‘circadian rhythm,’ a 24-hour periodic cycle common to most living beings, is shown here as being driven by a feedback loop that creates a corresponding accumulation and decay in the quantities of two proteins, *Per* and *Tim*. Both the proteins and the genes that code for the proteins are indicated in the diagram, as rectangular and circular nodes, respectively, while the links indicate various steps in the overall process, including DNA translation and transcription, phosphorylation, suppression, and even physical movement across the nuclear membrane (drawn as a vertical dashed line).

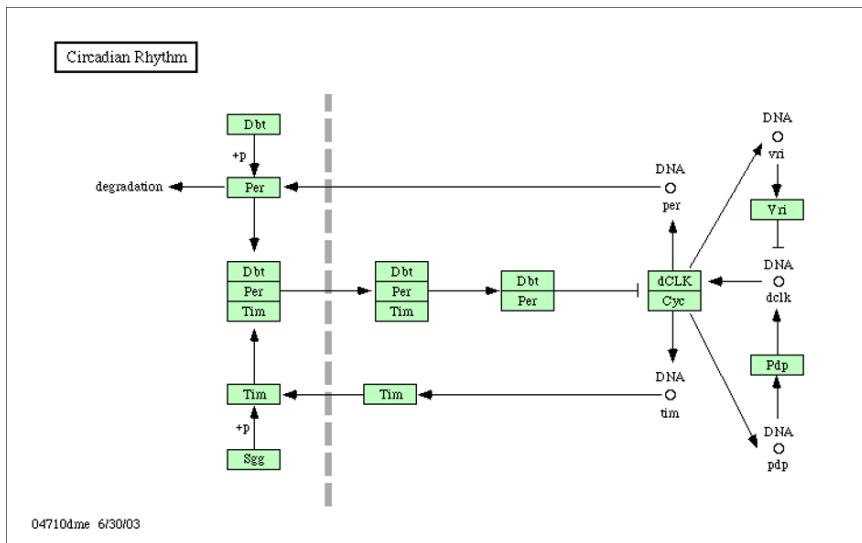


Fig. 1.3 Network representation of the circadian clock mechanism in *Drosophila melanogaster* (fruit fly), as of June 30, 2003, from the Kyoto Encyclopedia of Genes and Genomes (KEGG) [2].

Historically, the construction of such networks has been tremendously time-consuming. However, since roughly the mid-1990's, with the advent of high-throughput measurement techniques in genomics, the entire nature of this task has changed dramatically. The flood of data brings with it the potential to infer relational information like that in Figure 1.3 on a large scale in a semi- to fully-automated fashion. As a result, substantial energy is being focused on creating system-wide network representations of interactions among the different relevant biological elements (e.g., genes, proteins, etc.), and on the use of such networks for discovering higher-level phenomena associated with cellular processes. Examples of such tasks

range from the basic ‘mapping’ of the underlying biological system, to the search for meaningful clusters or sub-networks of system elements (i.e., ‘motifs’), to the inference of roles in cellular function (e.g., protein function prediction) and the prediction of the behavior of the overall system in the face of external influences (e.g., the response to new cancer drug treatments).

Not surprisingly, the nature of the data collected on biological networks and the manner in which they are analyzed and used vary widely with the nature of the underlying biological system being studied and our ability to obtain relevant measurements. We will encounter a variety of examples of biological networks, at various scales, and topics relevant to their study in Chapter 3 (i.e., mapping networks), Chapter 5, (i.e., network sampling), Chapter 6 (i.e., detection of network motifs), Chapter 7 (i.e., inference of networks), and Chapter 8 (i.e., protein function prediction and modeling of epidemiological processes).

1.2.4 Information Networks

Of particular use in this modern ‘information age,’ although by no means new, are information networks (i.e., networks describing relationships among elements of information). Standard examples include networks of citations between academic journals or papers, networks of co-authorship on papers, or networks indicating semantic relationships (e.g., synonym, antonym, etc.) between words or concepts. In addition, the Internet has helped spawn a number of well-known classes of information networks. The pre-eminent example is the World Wide Web (WWW), in which nodes typically are web pages and edges indicate the referencing of one page by another. Another class of Internet-related information networks are peer-to-peer (i.e., ‘P2P’), networks, in which nodes are typically Internet users and links indicate the exchange of content (e.g., music or movies) through an associated network protocol (e.g., Napster, Gnutella, KaZaa, etc.).

The ‘mapping’ of information networks in an informative fashion is usually a non-trivial task of significant interest in itself, particularly given their often massive size. Additionally, there is generally strong interest in questions regarding the structure of such networks, including which nodes are linked to many other nodes (e.g., “Who are the most highly cited authors within the mathematical sciences literature?”), whether certain tightly inter-woven subgraphs may be found (e.g., “How does the content of web pages induce clustering on the WWW?”), and the manner in which network size and structure change over time (e.g., “What are the dynamics of the lifetime of a scientific innovation?”).

As an illustration of an information network, consider the network depicted in Figure 1.4, which is an example of an important class of sub-networks of the WWW called ‘web-logs’ or simply ‘blogs’. Blogs are a form of Web authorship, primarily textual in nature but increasingly more multimedia based. The corresponding web page(s) of a blog consists of a set of entries, often on a particular topic(s), that are archived in reverse chronological order and usually updated frequently. ‘Blogging,’

as the act of maintaining a blog is called, has come to refer to a range of activities that include personal journal keeping, citizen reporting, information dissemination, and social interaction (e.g., support groups, political action, etc.). In fact, blog networks can be viewed not only as information networks, but often also as social networks representing so-called ‘virtual communities.’

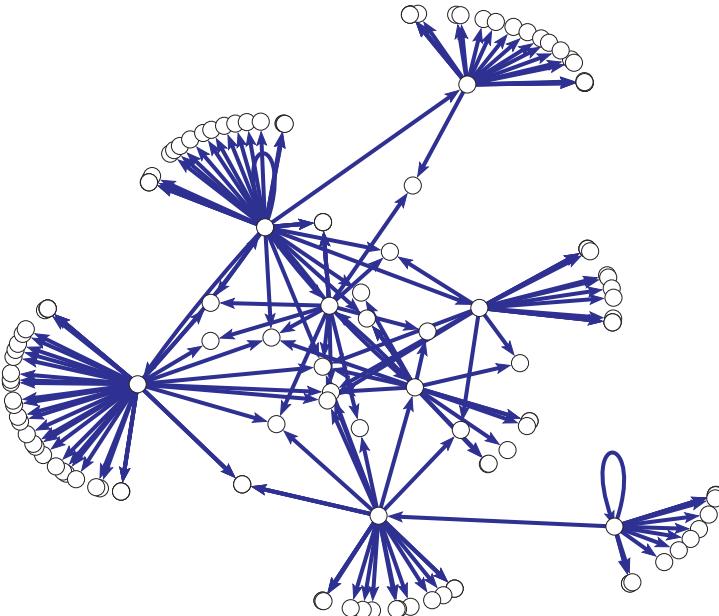


Fig. 1.4 AIDS Blog Network

The network in Figure 1.4 is a snapshot of the pattern of citation among 146 unique blogs related to AIDS, patients, and their support networks, collected by Gopal [184] over a randomly selected three-day period in August 2005. A directed edge from one blog to another indicates that the former has a link to the latter in their web page (more specifically, the former refers to the latter in their so-called ‘blogroll’). Collection of such data is facilitated by the very Internet, protocols, and software that make blogging possible in the first place. The resulting measurements can include not only indications of pairwise blogger interactions, but also information on blog content. Interesting questions relate, for example, to the dynamics of formation of blog communities, the role(s) of individual or subgroups of bloggers in those communities, and the spread of information throughout a blog network.

We will encounter topics relevant to information networks like this in Chapter 3 (i.e., mapping networks), Chapter 4 (i.e., description of structure and patterns), Chapter 5 (i.e., network graph sampling), and Chapter 7 (i.e., link prediction).

1.3 About this Book

Given the vast array of contexts in which networks arise, the multitude of questions asked in regards to those networks, and the variety of measurements taken towards the goal of answering these questions, how can one propose to write a single book on general topics relevant to the statistical analysis of network data? The answer to this question lies in the fact that it is possible – and indeed quite useful – to categorize many of the various tasks faced in the analysis of network data in different domains according to a statistical taxonomy. In so doing, a certain order and structure emerges for presenting some core of the multitude of statistical methods and models proposed in this area by researchers across diverse disciplines. It is along the lines of such a taxonomy, with the goal of presenting such a core, that this book is organized.

Broadly speaking, the material in this book is broken down into topics of (i) descriptive methods, in Chapters 3 - 4, and (ii) modeling and inference, in Chapters 6 - 10, with Chapter 5 playing somewhat of a transition role between the two. Interwoven throughout, we integrate relevant issues of data collection, data management, and computing. Of course, in reality ‘descriptive’ and ‘inferential’ techniques are not always cleanly separated, nor in practice does the task of statistical analysis flow linearly from the former to the latter and simply stop, but rather is generally iterative. Nevertheless, we find this organization convenient.

In more detail, the contents of this book are as follows. Chapter 2 contains preliminary technical material on graphs, probability, and statistical inference necessary for the rest of the book. Following these preliminaries, methods for the descriptive analysis of network data are developed in Chapters 3 and 4. In the former we concentrate on the task of converting network measurements into a network graph representation, while in the latter we focus on the description of structure and the identification of patterns in such network graphs. In Chapter 5 we explore the effects of sampling on the extent to which characteristics of an observed network graph reflect the corresponding characteristics of the underlying network being studied. Then in Chapters 6 and 7 we turn to the task of modeling network graphs. In Chapter 6 we study models for describing an observed network graph, while in Chapter 7 we consider the problem of inferring a network graph based on incomplete or indirect measurements. Next, in Chapters 8 and 9 we turn to problems involving the modeling and inference of processes on a network graph. Chapter 8 concerns network-indexed processes, of both a static and dynamic nature, while Chapter 9 is devoted to the special case of network flow processes.

Finally, in Chapter 10 we briefly discuss the topic of graphical models. The graphical modeling paradigm differs from that associated with most of the models described in this book – wherein graphs serve either as data objects themselves or effectively as indexing for other data objects – in that graphs are used to describe the conceptual structure (or, more formally, collections of conditional independence relations) associated with statistical inference. Nevertheless, the two perspectives are by no means entirely distinct, and in fact we will encounter a number of instances of graphical models earlier in Chapters 6 through 9. The purpose of Chapter 10 is to

make more precise the role that can be played by graphical models in the analysis of network data.

We have aimed in the writing of each chapter to concentrate on what we see as a core set of topics relevant to the theme of the chapter. In doing so, we inevitably exclude certain worthwhile and valuable aspects of the literature. For example, most of the material in the book is developed around the case of static network graphs. Material on dynamic network graphs (i.e., those evolving in time), an increasingly active but less mature area of network research, is included where relevant in certain chapters as only a small subsection near the end. In an attempt to compensate somewhat for this and similar truncation or exclusion of topics, we have included additional references to the literature in the section ‘Additional Related Topics and Reading’ appearing at the end of each chapter. However, admittedly, even through this device it is impossible to be comprehensive.

This book is intended for students and researchers across a wide range of quantitative disciplines, including bioinformatics, computer science, economics, information science, mathematics, physics, sociometrics, and – of course – statistics. At a minimum, a reader will need a solid foundation in calculus and linear algebra, as well as the equivalent of a strong introductory course in probability and in statistical modeling. Readers with additional background will (hopefully!) reap additional benefit accordingly. For those readers outside of statistics who would like to establish a more thorough grounding in statistical modeling and inference beyond the level of an introductory course, we recommend Wasserman’s *All of Statistics: A Concise Course in Statistical Inference* [392], a text written expressly for such readers.

The material in this book is generally presented in a manner that attempts to strike a balance between concepts and mathematical detail. It is expected that readers will want to follow the various threads woven throughout to their origins in the literature, and copious use of references has been made for this purpose. In addition, in order to encourage readers to explore certain topics in greater depth, we have included with each chapter a handful of exercises. These are a combination of analytical and computational exercises. The analytical exercises are often completion problems, picking up loose threads of a more technical nature from the main body of the chapter. The computational exercises are often open-ended, a format we have found useful in teaching this material to students of diverse backgrounds, allowing them to define and attack problems in specific topic areas in a manner most beneficial to whatever network-oriented agendas they bring to the course.

Lastly, we comment on the topic of software. The sheer variety of software available in this area for statistical analysis of network data mirrors the number of communities involved in such work. In this book we have chosen to be ‘software agnostic,’ in that we do not advocate nor solely use any one particular software for the examples presented. There are two reasons for this choice. First, no single software package at this time is best suited for conducting all of the statistical analyses described in this book – and it is not our aim to develop such a package. Second, when we have taught this material, our experience has been that students often come to the course already with a preferred package(s). Most of the network figures in

this book were created using **Pajek** [24, 111], while computing for the numerical illustrations generally was done using either the **R** [1] or **MATLAB®** software environments. We will attempt to maintain on the website for this book

<http://math.bu.edu/people/kolaczyk/SAND.html>

a (surely incomplete!) list of popular packages and software for network analysis. Readers are encouraged to help us maintain this list and to let us know of important omissions. Machine-readable copies of most data used in this book are also available from this website.

Chapter 2

Preliminaries

This chapter contains technical background for the material addressed throughout the rest of the book. We begin in Section 2.1 with an overview of necessary topics from graph theory, which provides us with much of the language and infrastructure for manipulating and describing networks and network data. We then turn in Section 2.2 to a brief review of fundamental elements from probability and statistical inference, which will provide us with most of the language and principles used here for the modeling and analysis of network data. Finally, in Section 2.3, we discuss, through a series of examples, some of the unique challenges inherent in the statistical analysis of network data. Readers sufficiently familiar with both graph theory and statistical inference may wish to skip this chapter and move directly to Chapter 3, after perhaps a quick detour to glance through the examples of Section 2.3 and the issues raised therein.

2.1 Background on Graphs

We have seen that the term ‘network,’ as commonly used, refers simply to a collection of elements and their inter-relations. The sub-field of mathematics known as graph theory lends precision to this notion. In particular, it provides a body of definitions, tools, techniques, and results for discussing graphs and their properties. These will play an especially important role in the material of Chapters 3 and 4. We review here some of the basic terminology and concepts on graphs, discuss common families of graphs, introduce certain important connections between graphs and matrix algebra, and then briefly visit the topic of graph data structures and algorithms.

2.1.1 Basic Definitions and Concepts

Formally, a *graph* $G = (V, E)$ is a mathematical structure consisting of a set V of *vertices* (also commonly called *nodes*) and a set E of *edges* (also commonly called *links*), where elements of E are unordered pairs $\{u, v\}$ of distinct vertices $u, v \in V$. The number of vertices $N_v = |V|$ and the number of edges $N_e = |E|$ are sometimes called the *order* and *size* of the graph G , respectively. Often, and without loss of generality,¹ we will label the vertices simply with the integers $1, \dots, N_v$, and the edges, analogously. A graph $H = (V_H, E_H)$ is a *subgraph* of another graph $G = (V_G, E_G)$ if $V_H \subseteq V_G$ and $E_H \subseteq E_G$. An *induced subgraph* of G is a subgraph $G' = (V', E')$, where $V' \subseteq V$ is a prespecified subset of vertices and $E' \subseteq E$ is the collection of edges to be found in G among that subset of vertices.

As defined, a graph has no edges for which both ends connect to a single vertex (called *loops*) and no pairs of vertices with more than one edge between them (called *multi-edges*). A graph with either of these properties is called a *multi-graph*. For simplicity, and reflecting the bulk of common practice, the presentation in this book will concentrate primarily on graphs, and not multi-graphs, though reference to the latter will be made where appropriate. When it is necessary to indicate explicitly that a graph G is not a multi-graph, we will refer to it as a *simple* graph, and its edges, as *proper* edges.

A graph G for which each edge in E has an ordering to its vertices (i.e., so that $\{u, v\}$ is distinct from $\{v, u\}$, for $u, v \in V$) is called a *directed graph* or *digraph*. Such edges are called *directed edges* or *arcs*, with the direction of an arc $\{u, v\}$ read from left to right, from the *tail* u to the *head* v . Note that there is a natural extension of digraphs to *multi-digraphs*, where multiple arcs (i.e., *multi-arcs*) share the same head and tail. Note too, however, that digraphs may have two arcs between a pair of vertices without their being multi-arcs if the vertices play opposite roles of head and tail for the respective arcs. In this case, the two arcs are said to be *mutual*.

It is necessary to have a language for discussing the connectivity of a graph. One of the most basic notions of connectivity is that of adjacency. Two vertices $u, v \in V$ are said to be *adjacent* if joined by an edge in E . Similarly, two edges $e_1, e_2 \in E$ are adjacent if joined by a common endpoint in V . A vertex $v \in V$ is *incident* on an edge $e \in E$ if v is an endpoint of e . From this follows the notion of the *degree* of a vertex v , say d_v , defined as the number of edges incident on v . The *degree sequence* of a graph G is the sequence formed by arranging the vertex degrees d_v in non-decreasing order. The sum of the elements of the degree sequence is equal to twice the number of edges in the graph (i.e., twice the size of the graph). Note that for digraphs, vertex degree is replaced by *in-degree* (i.e., d_v^{in}) and *out-degree* (i.e., d_v^{out}), which count the number of edges pointing in towards and out from a vertex, respectively. Hence, digraphs have both an in-degree sequence and an out-degree sequence.

¹ Technically, a graph G is unique only up to relabellings of its vertices and edges that leave the structure unchanged. Two graphs that are equivalent in this sense are called *isomorphic*.

It is also useful to be able to discuss the concept of movement about a graph. For example, a walk on a graph G , from v_0 to v_l , is an alternating sequence $\{v_0, e_1, v_1, e_2, \dots, v_{l-1}, e_l, v_l\}$, where the endpoints of e_i are $\{v_{i-1}, v_i\}$. The length of this walk is said to be l . Refinements of a walk include trails, which are walks without repeated edges, and paths, which are trails without repeated vertices. A trail for which the beginning and ending vertices are the same is called a circuit. Similarly, a walk of length at least three, for which the beginning and ending vertices are the same, but for which all other vertices are distinct from each other, is called a cycle. Graphs containing no cycles are called acyclic. In a digraph, these notions generalize naturally. For example, a directed walk from v_0 to v_l proceeds from tail to head along arcs between v_0 and v_l .

A vertex v in a graph G is said to be reachable from another vertex u if there exists a walk from u to v . The graph G is said to be connected if every vertex is reachable from every other. A component of a graph is a maximally connected subgraph. That is, it is a connected subgraph of G for which the addition of any other remaining vertex in V would ruin the property of connectivity. For a digraph, there are two variations of the concept of connectedness. A digraph G is weakly connected if its underlying graph (i.e., the result of stripping away the labels ‘tail’ and ‘head’ from G) is connected. It is called strongly connected if every vertex v is reachable from every u by a directed walk.

A common notion of distance between vertices on a graph is defined as the length of the shortest path(s) between the vertices (which we set equal to infinity if no such path exists). This distance is often referred to as geodesic distance, with ‘geodesic’ being another name for shortest paths. The value of the longest distance in a graph is called the diameter of the graph.

Finally, it is not uncommon to equip (or ‘decorate’) a graph G with auxiliary numerical values on its vertices, edges, or both. For example, edges $e \in E$ are often accompanied by edge weights. In fact, extending the notion of edge weights to all pairs of vertices, the edge set E itself can be represented through a set $\{w_e\}$ of such weights, i.e., $w_e = 1$ if $e \in E$ and 0 if $e \notin E$. When edges are weighted, the corresponding length of a walk (trail, path, etc.) is measured as the sum of the values of the weights along the edges traversed in the walk. The notion of distance generalizes accordingly. These concepts extend naturally to digraphs.

Similarly, graph labellings may be used in representing a multi-graph as a decorated graph. Specifically, given a multi-graph, we can define a graph G having the same vertex set V and having an edge set E such that distinct elements $u, v \in V$ have an edge between them if there is at least one multi-edge between them in the multi-graph. Then, equip each vertex $v \in V$ with a label denoting the number of loops possessed by v in the multi-graph, and similarly, equip each edge with the number of multi-edges it represents.

Of course, in this book a particularly common source of labels for graph vertices and edges will be in the form of measurements of functions or processes on a given graph. However, we postpone introducing the necessary notation for such quantities until Section 2 of this chapter.

2.1.2 Families of Graphs

Graphs come in all ‘shapes and sizes,’ as it were, but there are a number of families of graphs that are commonly encountered in practice. Such prevalence typically is due to some combination of relevance and tractability, with the latter being of an analytical or computational nature, or both. We describe here a handful of the most common examples of graph families.

A *complete* graph is a graph where every vertex is joined to every other vertex by an edge. Figure 2.1 shows, on the left, a representation of a complete graph. This concept is perhaps most useful in practice through its role in defining a *clique*, which is a complete subgraph. A subgraph H of a graph G is said to be a maximal clique if it is complete and no other such subgraph contains it. Cliques are an extreme form of a ‘highly inter-connected’ subgraph, the existence and detection of which is often of interest in the analysis of a network. We will encounter additional, more flexible forms of this idea in Chapter 4.

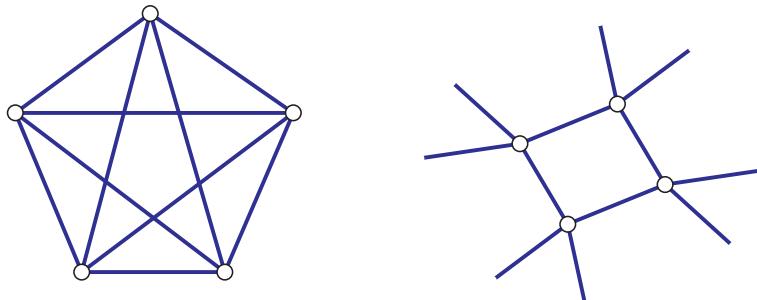


Fig. 2.1 Left: a complete graph. Right: a portion of a 4-regular graph.

A *regular* graph is a graph in which every vertex has the same degree. A regular graph with common degree d is called *d-regular*. Figure 2.1 shows, on the right, a portion of a 4-regular graph, such as is found in the common lattice defining, say, the squares on a chess board.² Regular graphs arise commonly in the study of various quantities in physics and chemistry (e.g., crystal structure), and arise frequently in geo-spatial settings (e.g., as a model of pixel adjacencies in image processing).

A connected graph with no cycles is called a *tree*. The disjoint union of such graphs is called a *forest*. Trees are of fundamental importance in the analysis of networks. They serve, for example, as a key data structure in the efficient design of many computational algorithms. A digraph whose underlying graph is a tree is called a *directed tree*. Often such trees have associated with them a special vertex

² Technically, to render this example truly 4-regular, the board must be made cyclic, in the manner of a circle, by joining the top and bottom edges (thus creating a tube) and then the open ends (thus creating a so-called toroidal lattice).

called a ***root***, which is distinguished by being the only vertex from which there is a directed path to every other vertex in the graph. Such a graph is called a *rooted tree*. An example is shown in Figure 2.2. A vertex preceding another vertex on a path from the root is called an *ancestor*, while a vertex following another vertex is called a *descendant*. Immediate ancestors are called *parents*, and immediate descendants, *children*. A vertex without any children is called a *leaf*. Given a rooted tree of this sort, it is not uncommon to represent it diagrammatically without any indication of its directedness, as this is to be understood from the definition of the root.

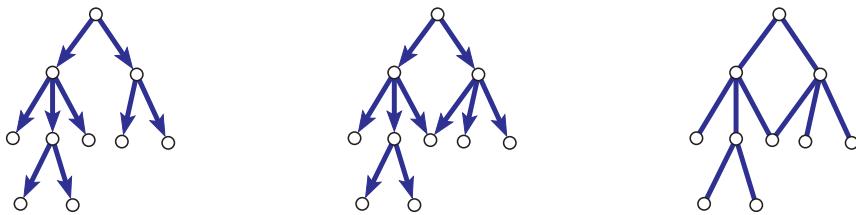


Fig. 2.2 Left: a rooted tree. Center: a DAG. Right: the undirected graph underlying the DAG.

An important generalization of the concept of a tree is the ***directed acyclic graph*** (i.e., the DAG). A DAG, as its name implies, is directed and has no cycles. However, unlike a directed tree, its underlying graph is not a tree, in that replacing the arcs with undirected edges leaves a (simple) graph that contains cycles. Nevertheless, it is often possible to still design efficient computational algorithms on DAGs that take advantage of this near-tree-like structure. An example of a DAG is shown in Figure 2.2, along with its underlying graph.

A **bipartite** graph is a graph $G = (V, E)$ such that the vertex set V may be partitioned into two disjoint sets, say V_1 and V_2 , and each edge in E has one endpoint in V_1 and the other in V_2 . Such graphs are typically used to represent ‘membership’ networks, for example, with ‘members’ (e.g., people) denoted by vertices in V_1 , and the corresponding ‘organizations’ (e.g., clubs), by vertices in V_2 . It is not uncommon to accompany a bipartite graph with at least one of two possible induced graphs. Specifically, a graph $G_1 = (V_1, E_1)$ may be defined on the vertex set V_1 by assigning an edge to any pair of vertices that both have edges in E to at least one common vertex in V_2 . Similarly, a graph G_2 may be defined on V_2 . Figure 2.3 shows an example of a bipartite graph G and its induced graph G_1 .

As a last example, we mention the class of planar graphs. A graph G is said to be ***planar*** if, informally speaking, it may be drawn in the plane, with vertices as dots and edges as lines, in such a way that no pair of edges intersect anywhere other than at vertices to which they are jointly incident. Planar graphs are often an appropriate representation of networks with a spatial component, such as many technological networks, and have a number of special properties induced by the structural requirement that they ‘lie in’ the plane.

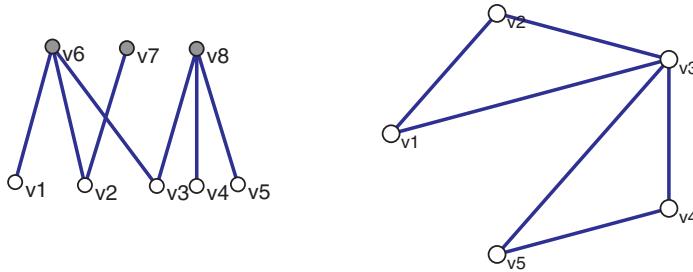


Fig. 2.3 Left: a bipartite graph. Right: a graph induced by the bipartite graph on the ‘white’ vertex set.

2.1.3 Graphs and Matrix Algebra

We shall see that it is frequently useful in the modeling and analysis of network data to be able to characterize a graph G and certain aspects of its structure using matrices and matrix algebra. Our ability to do so in a rigorous manner derives from a formal blending of graph theory with matrix algebra, in a field called algebraic graph theory, the roots of which go back to Kirchoff and his study of electrical networks. We briefly describe a handful of the elements from this area that will be of particular use to us in various parts of this book.

The fundamental connectivity of a graph G may be captured in an $N_v \times N_v$ binary, symmetric matrix \mathbf{A} with entries

$$A_{ij} = \begin{cases} 1, & \text{if } \{i, j\} \in E, \\ 0, & \text{otherwise,} \end{cases} \quad (2.1)$$

where we use the integers $1, \dots, N_v$ generically to denote the elements of V and we represent an edge $e \in E$ explicitly as an unordered pair of vertices $i, j \in V$. In words, \mathbf{A} is non-zero for entries whose row-column indices correspond to vertices in G joined by an edge, and zero, for those that are not.

The matrix \mathbf{A} , called the *adjacency matrix*, is useful not only for storing connectivity information, but also in that certain operations on \mathbf{A} yield additional information concerning G . For example, the row sum $A_{i+} = \sum_j A_{ij}$ is simply equal to the degree d_i of vertex i . Note that, by symmetry, $A_{i+} = A_{+i}$. Furthermore, if we let \mathbf{A}^r denote the r -th power of \mathbf{A} , then the entry A_{ij}^r yields the number of walks of length r between i and j on G . Finally, there are many interesting and useful relations involving the eigenvalues of G . For example, it can be shown that G is a regular graph if and only if the maximum degree d_{max} of G is an eigenvalue of \mathbf{A} .

An adjacency matrix may also be defined for digraphs, by adjusting the definition in (2.1) so that $A_{ij} = 1$ if $\{i, j\} \in E$ represents a directed edge from i to j . Of course, \mathbf{A} is now no longer symmetric. However, it still contains similarly useful additional information. For example, $A_{i+} = d_i^{out}$ and $A_{+j} = d_j^{in}$.

Another useful matrix capturing the fundamental structure in G is the incidence matrix \mathbf{B} , an $N_v \times N_e$ binary matrix with entries

$$B_{ij} = \begin{cases} 1, & \text{if vertex } i \text{ is incident to edge } j , \\ 0, & \text{otherwise .} \end{cases} \quad (2.2)$$

Given their definitions, it is natural to expect that there be a relation between \mathbf{A} and \mathbf{B} . This is indeed true, if we extend \mathbf{B} to a signed incidence matrix, say $\tilde{\mathbf{B}}$, where the entries ‘1’ in (2.2) are given plus or minus signs indicating an arbitrarily assigned ‘orientation’ of the corresponding edge.³ It then can be shown that $\tilde{\mathbf{B}}\tilde{\mathbf{B}}^T = \mathbf{D} - \mathbf{A}$, where $\mathbf{D} = \text{diag}[(d_i)_{i \in V}]$ is a diagonal matrix containing the degree sequence.

The $N_v \times N_v$ matrix $\mathbf{L} = \mathbf{D} - \mathbf{A}$ is in fact quite important in its own right and is called the Laplacian of the graph G . Motivation for the term Laplacian, in analogy to the Laplacian from multivariable calculus (i.e., the sum of second partial derivatives of a function), may be found in the fact that, for a real-valued vector $\mathbf{x} \in \mathbb{R}^{N_v}$, we have

$$\mathbf{x}^T \mathbf{L} \mathbf{x} = \sum_{\{i,j\} \in E} (x_i - x_j)^2 . \quad (2.3)$$

The closer this value is to zero, the more similar are the elements of \mathbf{x} at adjacent vertices in V . Hence, the Laplacian is useful in providing, through (2.3), some sense of the ‘smoothness’ of functions on a graph G , with respect to the connectivity of G . We will see this fact exploited in Chapter 8, for the purpose of regression and classification on network graphs.

The properties of \mathbf{L} , particularly the properties of its eigenvalues and eigenvectors, have much to say about the structure of G . Since \mathbf{L} can be shown to be a positive semi-definite matrix, the eigenvalues are all non-negative. And because $\mathbf{L}\mathbf{1} = \mathbf{0}$, where $\mathbf{1}$ and $\mathbf{0}$ are $n_v \times 1$ vectors of ones and zeroes, respectively, the smallest eigenvalue λ_1 of \mathbf{L} is equal to zero. The second smallest eigenvalue, λ_2 , is typically nontrivial, and arguably the most important of all of the eigenvalues. For example, roughly speaking, the larger λ_2 is, the more ‘connected’ G is, and the more difficult it is to separate G into disconnected subgraphs by selectively eliminating some small number of edges. We will see in Chapter 4 that results of this kind are useful, for instance, in designing tools for clustering the vertices of the graph into meaningful subgraphs.

2.1.4 Graph Data Structures and Algorithms

Facilitating the transition from graphs as purely mathematical objects to graphs as practical tools for use in the analysis of network data are data structures and algorithms for graphs. The study of graph data structures and algorithms is basic to the field of computer science, where much effort has gone into the development of

³ More formally, we arbitrarily assign each vertex in an edge j the role of either head or tail, as in a digraph, and let $\tilde{B}_{ij} = 1$ if vertex i is incident to edge j as a tail, and -1 , if as a head.

efficient methods for the storage and manipulation of a graph, as well as methods for computing various characteristics of and answering different questions about graphs. We briefly highlight some of the most relevant elements of this literature now. Their role in what follows will be most evident in Chapters 3 and 4.

2.1.4.1 Data Structures

There are two common data structures for representing a graph G . The first is the $N_v \times N_v$ adjacency matrix \mathbf{A} defined previously. This choice is often a natural one, given that matrices are fundamental data objects in most programming and software environments. However, if the graph is particularly large, and especially if the graph is sparse (e.g., if $N_e \sim N_v$), then it can be preferable to use a collection of *adjacency lists*. This is because the adjacency matrix in such cases will be both large and filled primarily with zeros, due to the fact that it explicitly represents both present and absent edges, while adjacency lists, on the other hand, store only the information on edges that are present. Specifically, an adjacency-list representation of a graph G is simply an array of size N_v , ordered with respect to the ordering of the vertices in V , each element of which is a list, where the i -th list contains the set of all vertices j for which there is an edge from i to j . A variation on this idea is an *edge list*, a simple two-column list of all vertex pairs that are joined by an edge.

The sum of the lengths of the adjacency lists will be N_e for a directed graph and $2N_e$ for an undirected graph. Therefore, the total amount of computer memory space required for storing a graph in an adjacency list representation is⁴ only $O(N_v + N_e)$. When G is sparse, this memory requirement will be much less than the value $O(N_v^2)$ associated with storage of G through an adjacency matrix. On the other hand, when G is dense (i.e., $N_e \sim N_v^2$), the memory requirements for the two methods will be comparable. In addition, the simplicity of the adjacency matrix representation may sometimes be felt to outweigh any memory disadvantages, especially for smaller graphs.⁵

Note that for either adjacency matrix or adjacency list representations, it is easy to store annotated graphs in a similar fashion. For example, an edge weight w_e , for $e = \{i, j\} \in E$, may be stored in the (i, j) -th location of the adjacency matrix, or in addition to the value j (or i) in the list for i (or j), in an adjacency-list representation.

⁴ For two sequences $(h_n)_{n \geq 0}$ and $(g_n)_{n \geq 0}$, we say $h_n = O(g_n)$, read as ‘ h_n is big-oh of g_n ’, if there exists a constant c such that eventually $|h_n/g_n| \leq c$, for large n .

⁵ Ultimately, a comparison of the two methods of data representation for graphs can be more subtle than we have described. For example, sparse matrix storage methods allow one to effectively store the adjacency matrix in significantly less than $O(N_v^2)$ space. In addition, there can be important interactions between data representation and algorithm design that impact computational efficiency.

2.1.4.2 Algorithms

There are numerous questions we might wish to ask in regards to a given graph. Some are quite simple and their answers may be obtained in a straightforward manner directly from the data structures just discussed. For example, are vertices i and j linked by an edge? What is the degree of vertex i ? For many other questions, however, more work may be required, but it is usually still feasible to obtain an answer in a reasonably efficient manner. For example, what is the shortest path(s) between vertices i and j ? How many connected components does the graph have? And, for a directed graph, does it have cycles or is it acyclic? Finally, for certain questions, there is likely no efficient algorithm. An example of such a question is the one that asks for a maximal clique in a given graph.

Thus algorithmic complexity is frequently an important issue in the analysis of network data. Computational complexity theory distinguishes between ‘tractable’ and ‘intractable’ problems by breaking them into two groups – those that are solvable with an algorithm that runs in polynomial time, and those that are not. A polynomial-time algorithm with n inputs will run in time $\underline{O}(n^p)$ for some $p > 0$. If an algorithm is not polynomial for any choice of p , it is super-polynomial. For example, combinatorially exhaustive algorithms are often exponential, in the sense that they run in $O(a^n)$ time for some $a > 1$. Except for settings with very small n , it is unlikely to be feasible in practice to wait for super-polynomial algorithms to complete; the same is effectively true even for polynomial algorithms having sufficiently large p , with respect to n .

In the case of graph algorithms, the complexity of an algorithm usually is a function of both N_v and N_e . Ideally, one would like the running time of such algorithms to scale linearly in these values (i.e., with $p = 1$), while also keeping memory requirements under control. In practice, for graphs of any reasonably large dimension, a cubic algorithm (i.e., with $p = 3$) is generally pushing the upper limits for what may be computed in an acceptable amount of time. In the case of massive network graphs, however, often nothing less efficient than linear running time is feasible. The design of efficient algorithms is frequently nontrivial, with a more naive, inefficient algorithm being improved only through clever use of storage, indexing, or redundancies in the quantities to be computed.

Consider, for example, the notion of a ‘search’ on a graph, which we can picture as a process of moving outward from a given vertex towards the other vertices in the graph, seeking to fulfill a given criterion. The two most basic search algorithms, breadth-first search (BFS) and depth-first search (DFS), each seek to ‘discover’ the set of all vertices $j \in V$ that are reachable from a given source vertex i_s . Both algorithms run in $\underline{O}(N_v + N_e)$ time, and differ from each other primarily in the manner in which they choose to discover other vertices. The BFS algorithm works outward from i_s , first discovering vertices adjacent to i_s (i.e., one ‘hop’ away), then continuing to vertices two hops away, then three hops away, and so on, until all reachable vertices are discovered. The output of the algorithm is a tree, rooted at i_s and organized so that the path from i_s to a reachable vertex j in the tree corresponds to the shortest path from i_s to j in G . Conversely, the DFS algorithm, as its name suggests,

instead proceeds from i_s by delving as deeply into G as possible from the first adjacent vertex to i_s , after which it iteratively backtracks to the most recently discovered vertex j for which there are undiscovered edges to be explored.

The usefulness of these two search algorithms, and the choice of which to use when, derives from the context of the actual task to be performed. For example, the BFS algorithm is at the core of standard algorithms requiring shortest path information, like Prim's algorithm for producing a minimum spanning tree⁶ and Dijkstra's algorithm for finding all shortest paths from a single source i_s in a directed graph. The latter algorithms can be implemented to run in $O(N_e + N_v \log N_v)$ and $O(N_v^2 \log N_v + N_v N_e)$ time, respectively. The DFS algorithm is often a sub-routine in a larger algorithm, such as the ‘topological sort algorithm,’ which can be used to determine whether a directed graph G is acyclic or not, and algorithms for decomposing G into its strongly connected components, all of which can be implemented in linear time.

Interestingly, there are many problems for which it is unknown whether there even exists a polynomial-time algorithm to solve them. The study of such problems has been a deep and fundamental endeavor of theoretical computer science since the early 1970’s. Within this area there is a formal framework for classifying problems by how difficult they *may* be. Decision problems (i.e., problems requiring a ‘yes’ or ‘no’ answer) that are shown to be *NP-hard* or *NP-complete* are considered quite likely to be intractable.⁷ For optimization problems, wherein we seek to find a solution maximizing or minimizing a given objective function, the problem is referred to by these same terms if the associated decision problem of confirming or rejecting a candidate solution as optimal can be categorized as such. In the context of network graphs, the problem of finding a maximal clique in a graph is an example of an *NP*-complete optimization problem.

2.2 Background in Probability and Statistics

We here provide a brief overview of standard fundamental topics in probability and statistical inference. No attempt is made to be exhaustive in our presentation. Rather, our goal is simply to lay a foundation of terminology, notation, and concepts to be used later in the book, primarily in Chapters 5–9.

⁶ A minimum spanning tree $T \in E$ is a tree that spans the full set of vertices V and for whom the sum of edge weights $W(T) = \sum_{e \in T} w_e$ is minimized.

⁷ The designations *NP-hard* and *NP-complete* refer to two different but related notions of the difficulty of a problem. A precise understanding of the formal distinction between the two will not be necessary in this book. The interested reader is referred to the text by Cormen, Leiserson, Rivest, and Stein [100, Ch. 34].

2.2.1 Probability

2.2.1.1 Basic Definitions and Concepts

Given some sense of an ‘experiment’ (e.g., flipping a coin, determining if two classmates are friends, observing traffic volume on an Internet link, etc.) and an accompanying *sample space* Ω of potential outcomes of that experiment, we will denote the *probability* of an event $A \subseteq \Omega$ by $\mathbb{P}(A)$, where P is a function that assigns a number between zero and one to an event and that satisfies the Kolmogorov axioms. Specifically, we assume (i) $\mathbb{P}(A) \geq 0$ for any event A , (ii) $\mathbb{P}(\Omega) = 1$, and (iii) $\mathbb{P}(A \cup B) = \mathbb{P}(A) + \mathbb{P}(B)$ for any disjoint events A and B (i.e., such that $A \cap B = \emptyset$).

Two events A and B are said to be *independent* if $\mathbb{P}(A \cap B) = \mathbb{P}(A)\mathbb{P}(B)$. For events B such that $\mathbb{P}(B) > 0$, the *conditional probability* of A given B is $\mathbb{P}(A|B) = \mathbb{P}(A \cap B)/\mathbb{P}(B)$. It can be shown that A and B are independent if and only if $\mathbb{P}(A|B) = \mathbb{P}(A)$. From the facts that $\mathbb{P}(A \cap B) = \mathbb{P}(A|B)\mathbb{P}(B) = \mathbb{P}(B|A)\mathbb{P}(A)$ and $\mathbb{P}(B) = \mathbb{P}(B|A)\mathbb{P}(A) + \mathbb{P}(B|A^c)\mathbb{P}(A^c)$, where A^c indicates the complement of A within Ω , we obtain *Bayes rule*,

$$\mathbb{P}(A|B) = \frac{\mathbb{P}(B|A)\mathbb{P}(A)}{\mathbb{P}(B|A)\mathbb{P}(A) + \mathbb{P}(B|A^c)\mathbb{P}(A^c)} . \quad (2.4)$$

For three events A , B , and C , the events A and B are said to be *conditionally independent given C* if

$$\mathbb{P}(A \cap B | C) = \mathbb{P}(A | C)\mathbb{P}(B | C) . \quad (2.5)$$

The concept of a *random variable* as a mapping from a sample space Ω to the real-numbers \mathbb{R} , is used to formalize the notion of a measurement (i.e., data) obtained in association with an experiment. We will use upper case roman characters (e.g., X , Y , and Z) to denote random variables, and lower case (e.g., x , y , and z), to denote specific values taken on by the corresponding random variables. The *cumulative distribution function* (CDF) of a random variable X will be expressed as $F_X(x) = \mathbb{P}(X \leq x)$, and the *complementary CDF* (CCDF), as $\bar{F}_X(x) = 1 - F_X(x)$. The random variable X is said to ‘have distribution’ or ‘be distributed as’ F_X , which is often written as $X \sim F_X$. Similarly, we will use f_X to denote either the *probability mass function* (PMF), if X is a discrete random variable, or the *probability density function* (PDF), if X is a continuous random variable. For convenience, when the distinction is not important, we will typically use ‘PDF’ to refer to such functions. Furthermore, when the underlying random variable X is to be understood from context, we may write the CDF and PDF functions simply as F and f .

We represent the *expectation* of a random variable X as $\mu_X \equiv \mathbb{E}(X) \equiv \mathbb{E}_X(X)$. For discrete random variables this is the sum of the quantities $x f_X(x)$ over the range of values x for which $f_X(x) > 0$, while for continuous random variables, it is the integral of the same.⁸ In the case where the random variable $X = I_A$ is the *indicator*

⁸ It is implicitly assumed in discussing an expectation that the corresponding sum or integral is well defined, in the sense of being finite; if it is infinite, the expectation is said not to exist.

of the event A (i.e., I_A is one if A occurs and zero if not), it may be shown that $\mathbb{E}(I_A) = \mathbb{P}(A)$.

For a function $g(x)$, the expectation of the random variable $Y = g(X)$ is simply $\mathbb{E}_Y(Y) = \mathbb{E}_X(g(X))$. Commonly occurring examples of g include $g(x) = x^k$, for k a positive integer, from which we obtain the k -th moment $\mathbb{E}(X^k)$ of X , and $g(x) = (X - \mu_X)^2$, from which we obtain the variance $\sigma_X^2 \equiv \mathbb{V}(X) \equiv \mathbb{E}(X - \mu_X)^2$ of X . The quantity $\sigma_X = \sqrt{\mathbb{V}(X)}$ is called the *standard deviation* of X . If a random variable X is transformed linearly as $g(X) = cX + b$, then the expectation and variance of the new random variable have the forms $\mathbb{E}(cX + b) = c\mathbb{E}(X) + b$ and $\mathbb{V}(cX + b) = c^2\mathbb{V}(X)$.

These concepts extend easily to several random variables. For example, in the case of two random variables X and Y , their *joint* CDF will be denoted $F_{X,Y}(x,y) = \mathbb{P}(X \leq x, Y \leq y)$, and their joint PDF, similarly as $f_{X,Y}(x,y)$. The *marginal* densities f_X and f_Y may be obtained by integrating (or summing, in the case of discrete random variables) the bivariate density $f_{X,Y}(x,y)$ over all values of y or x , respectively. The variables X and Y are said to be independent if $\mathbb{P}(X \in A, Y \in B) = \mathbb{P}(X \in A)\mathbb{P}(Y \in B)$, for all pairs of events A and B , in which case we have $f_{X,Y}(x,y) = f_X(x)f_Y(y)$. From this last fact, we also have the useful relation $\mathbb{E}(XY) = \mathbb{E}(X)\mathbb{E}(Y)$.

The *conditional* PDF of X , given $Y = y$, is defined as $f_{X|Y}(x|y) = f_{X,Y}(x,y)/f_Y(y)$, for all y such that $f_Y(y) > 0$. Note that, for fixed y , $f_{X|Y}(x|y)$ is itself a well-defined PDF in x , over an appropriate range of values. Hence, for example, there is an accompanying notion of *conditional expectation*, $\mathbb{E}(X|Y = y) = \int xf_{X|Y}(x|y)dx$. It should be noted that this quantity is a function of the value y .

The notion of conditional independence, as defined for events in (2.5), also extends to random variables. Specifically, the random variables X and Y are said to be *conditionally independent given Z* if

$$f_{X,Y|Z}(x,y|z) = f_{X|Z}(x|z)f_{Y|Z}(y|z) , \quad (2.6)$$

for all x, y , and z .

When we have a collection of n random variables X_1, \dots, X_n to which we wish to refer as a unit, in either vector or matrix form, we will typically do so through bold-faced type (e.g., $\mathbf{X} = (X_1, \dots, X_n)^T$). We will denote the corresponding PDF by $f(\mathbf{x}) \equiv f(x_1, \dots, x_n)$, the mean vector, by $\boldsymbol{\mu} \equiv (\mathbb{E}(X_1), \dots, \mathbb{E}(X_n))^T$, and the *variance-covariance* matrix, by $\boldsymbol{\Sigma} \equiv \mathbb{V}(\mathbf{X}) \equiv [\text{Cov}(X_i, X_j)]_{i,j}$, where $\text{Cov}(X_i, X_j) = \mathbb{E}[(X_i - \mu_{X_i})(X_j - \mu_{X_j})]$ and $\text{Cov}(X_i, X_i) = \mathbb{V}(X_i)$. Scaling the covariances by the standard deviations of the corresponding random variables, we obtain the *correlation* $\rho_{X_i, X_j} = \text{corr}(X_i, X_j) = \text{Cov}(X_i, X_j)/\sigma_{X_i}\sigma_{X_j}$.

If an n -length random vector \mathbf{X} is transformed linearly as $\mathbf{CX} + \mathbf{b}$, where \mathbf{C} is an $m \times n$ real-valued matrix and \mathbf{b} is a m -length real-valued vector, then

$$\mathbb{E}(\mathbf{CX} + \mathbf{b}) = \mathbf{C}\mathbb{E}(\mathbf{X}) + \mathbf{b} \quad \text{and} \quad \text{Cov}(\mathbf{CX} + \mathbf{b}) = \mathbf{C}\mathbb{V}(\mathbf{X})\mathbf{C}^T . \quad (2.7)$$

In the special case that the elements of \mathbf{X} are uncorrelated (i.e., $\rho_{X_i, X_j} = 0$ for all $i \neq j$), the expression for the covariance reduces to $\sum_{i=1}^n C_{ii}^2 \mathbb{V}(X_i)$.

When the element X_i 's in a random vector \mathbf{X} are in fact all independent and ‘copies’ of a random variable X , with common distribution F_X , they will be said to be *independent and identically distributed* or ‘iid’. This condition is often expressed in the notation $X_1, \dots, X_n \sim F$. Note that in many cases of interest, the variables may be independent, but from different distributions.

For dependent random variables, it is common to use certain assumptions of conditional independence to simplify analytical and numerical calculations. The notion of a *Markov chain* is arguably the prototypical example. We say that a sequence of integer-valued random variables X_0, X_1, X_2, \dots is a (homogeneous) Markov chain if

$$\begin{aligned} \mathbb{P}(X_{m+1} = j | X_m = i, X_{m-1} = i_{m-1}, \dots, X_0 = i_0) &= \mathbb{P}(X_{m+1} = j | X_m = i) \\ &= P_{ij}, \end{aligned} \quad (2.8)$$

for any choice of $i_0, \dots, i_{m-1}, i_m, j$, called *states*, and any positive integer m , which is typically thought of as a time index. This Markov chain is an example of a discrete-time stochastic process. The first equality in (2.8) states that, given the past history of the process up through time m , the distribution of the process at time $m + 1$ depends only on its state at time m . The second equality indicates that this conditional distribution does not vary with m . The values P_{ij} are called *transition probabilities*.

Define $P_{ij}^{(n)}$ to be the probability that, starting in state i , the Markov chain is found in state j at n units of time (or ‘transitions’) later. The chain is called *irreducible* if any state can be reached from any other in some finite number of transitions (i.e., $P_{ij}^{(n)} > 0$ for some finite n for all states i, j). Importantly, under appropriate conditions,⁹ irreducible chains have a well-defined *stationary distribution*, call it $\{\pi_j\}$, in the sense that the limit

$$\lim_{n \rightarrow \infty} P_{ij}^{(n)} = \pi_j \quad (2.9)$$

exists for each state j , with $\pi_j \geq 0$ and $\sum_j \pi_j = 1$, and it is uninfluenced by the starting state i .

On a final note, in regards to basic definitions and concepts, we point out that in many contexts we will find it is useful to consider a graph G itself as a random object. Formally, this is typically done by thinking of G as having been drawn at random from a collection of possible graphs, say \mathcal{G} , whose definition may be explicit or implicit. In such cases, the value $\mathbb{P}(G)$ will refer to the probability of drawing G from \mathcal{G} under some stated sampling mechanism. Similarly, if $\eta(G)$ represents a real- or vector-valued quantity of characteristics of G (e.g., the number of vertices or

⁹ Specifically, besides irreducibility, the chain must be *ergodic* in that (i) it is not constrained to cycle through states in a periodic manner (i.e., aperiodic), and (ii) from any given state, it is guaranteed to return to that state in a finite number of transitions, with the expected number of transitions also finite (i.e., positive recurrent).

the degree sequence), then $\mathbb{E}[\eta(G)]$ will denote the expected value of that quantity under the sampling on \mathcal{G} .

2.2.1.2 Asymptotic Statements

Often it is possible to produce statements regarding the behavior of a random quantity, as some other quantity, such as sample size, tends toward a limiting value. Such asymptotic statements can be quite useful in looking past the potentially horrendous (if not impossible) details of an exact calculation to concisely summarize the state of affairs when one is close enough to ‘asymptopia.’ The limiting relation in (2.9) is an example of such a statement. Expressions for the n -step transition probabilities $P_{ij}^{(n)}$ are generally not expressable in closed form other than in the simplest of cases. Nevertheless, under the assumed conditions, (2.9) will always hold true, and sometimes informative expressions for the limiting values π_j may be obtained. We quickly review here three canonical types of asymptotic statements that will be encountered in this book.

Let A_1, A_2, \dots be a sequence of events, in association with a suitably defined probability model. We say A_n occurs *with high probability* (WHP) if

$$\lim_{n \rightarrow \infty} \mathbb{P}(A_n) = 1 . \quad (2.10)$$

An example of such a statement is the *weak law of large numbers* (WLLN) for iid random variables X_1, X_2, \dots , with common mean μ , which provides a characterization of the asymptotic behavior of the sample mean $\bar{X}_n = \sum_i X_i/n$. Specifically, it states that (2.10) holds, with A_n defined to be the event that $|\bar{X}_n - \mu| \leq \epsilon$, for $\epsilon > 0$ a fixed constant.

Probably more frequently cited in practice, is the analogous *strong law of large numbers* (SLLN), which states that, under the same conditions,

$$\mathbb{P}\left(\lim_{n \rightarrow \infty} \bar{X}_n = \mu\right) = 1 . \quad (2.11)$$

We note that the strong law implies the weak law, as the names suggest; the converse is not true.

Finally, there is the notion of a *central limit theorem*, the various versions of which state essentially that the probability distribution of suitably normalized sums of random variables tends in the limit of many such random variables to a Gaussian distribution¹⁰ with mean zero and unit variance. For example, in the case of a sequence of iid random variables X_1, X_2, \dots , with common mean μ and finite variance

¹⁰ A random variable X is said to have a *Gaussian* or *normal* distribution, with mean parameter μ and variance parameter σ^2 if

$$f_X(x) = (2\pi\sigma^2)^{-1/2} \exp\{-(x-\mu)^2/(2\sigma^2)\} .$$

If $\mu = 0$ and $\sigma = 1$, Z is said to be a *standard normal* random variable.

$\sigma^2 > 0$, the central limit theorem states that

$$\lim_{n \rightarrow \infty} \mathbb{P} \left(\frac{\bar{X} - \mu}{\sigma/\sqrt{n}} \leq x \right) = \Phi(x) , \quad (2.12)$$

where $\Phi(\cdot)$ is the CDF for a Gaussian random variable with mean zero and unit variance.

2.2.1.3 Simulation of Random Variables

Given a random variable X , with PDF f , it is frequently necessary in applications to compute quantities that can be expressed in integrals of the form

$$\mathbb{E}(g(X)) = \int g(x)f(x)dx , \quad (2.13)$$

for some function g of x (or sums of the analogous form, in the discrete case). The expression in (2.13) includes, for example, the expectation of X , its variance, and the probability that $X \in A$, for sets $A \subset \mathbb{R}$. When computationally tractable, closed-form expressions are not available for this purpose, and numerical integration is infeasible, methods based on numerical simulation – that is, ‘Monte Carlo’ methods – generally may be used instead. Such methods work by generating copies x_1, \dots, x_n of X from f , and computing the stochastic approximation

$$\widehat{\mathbb{E}}(g(X)) = \frac{1}{n} \sum_{i=1}^n g(x_i) . \quad (2.14)$$

An appeal to an appropriate strong law of large numbers then guarantees that the sample average in (2.14) is close to (2.13), for suitably large n .

The challenge for the Monte Carlo approach therefore lies largely in developing suitable methods for simulating draws from the distribution f . In the most common case, we are interested in making draws of iid copies from f . The starting point for doing so is the now-standard ability to generate so-called pseudo-random numbers. That is, to simulate draws of a uniform random variable on $[0, 1]$ (i.e., with PDF $f(x) = 1$ for $x \in [0, 1]$, and zero elsewhere). Methods to simulate draws from more sophisticated distributions f then generally utilize pseudo-random numbers in clever ways. For example, if X is a random variable with continuous CDF F , we can use the fact that the quantity $U = F(X)$, as a random variable, is distributed uniformly on $[0, 1]$ to generate draws from X , through the relation $X \sim F^{-1}(U)$. This is sometimes referred to as the *inverse transform* method.

Sometimes it is too difficult to generate samples from the distribution f that we desire, but we are instead able to sample from another distribution, say h . Rewriting the expression in (2.13) as

$$\mathbb{E}(g(X)) = \frac{\int g(x)w(x)h(x)dx}{\int w(x)h(x)dx} , \quad (2.15)$$

where $w(x) = f(x)/h(x)$, with the method of *importance sampling* we form the quantity

$$\widehat{\mathbb{E}}_h(g(X)) = \frac{(1/n) \sum_{i=1}^n w(x_i)g(x_i)}{(1/n) \sum_{i=1}^n w(x_i)} , \quad (2.16)$$

using draws x_1, \dots, x_n from h . The name ‘importance sampling’ derives from the fact that the method can be used to reduce the error in our approximation of (2.13), by concentrating our sampling in that part of the support of f where it is most needed.

Also commonly used in practice are methods based on Markov chains i.e., *Markov chain Monte Carlo* (MCMC) methods. These methods generate draws from a particular Markov chain, therefore yielding dependent random variables, where the chain is designed so that the limiting stationary distribution (e.g., as in (2.9), for the case of discrete states) is in fact the distribution f of interest to us. MCMC methods can therefore be substantially more complicated than methods like those just described. However, designed appropriately, they allow us to generate samples from quite complex distributions and are used frequently in network modeling.

Perhaps the most widely used type of MCMC algorithm is the Metropolis-Hastings algorithm, which is quite general and requires only that we can evaluate the PDF/PMF.¹¹ It is an example of a so-called *acceptance-rejection method*, wherein possible values are proposed for the next draw, but only some are accepted. For simplicity, consider the case where f is a PMF for a discrete random variable X . Suppose the current value generated from our Markov chain is $X_m = i$. We then propose a candidate state j for X_{m+1} , where j is drawn according to a proposal distribution $\{q_{ij}\}_i$. We accept the proposed state j with probability

$$\alpha_{ij} = \begin{cases} \min \left[\frac{f(j)q_{ji}}{f(i)q_{ij}}, 1 \right], & \text{if } f(i)q_{ij} > 0 , \\ 1, & \text{otherwise ,} \end{cases} \quad (2.17)$$

and reject with probability $1 - \alpha_{ij}$.

The primary challenge here is to design an appropriate proposal distribution $\{q_{ij}\}$, one that not only produces a Markov chain with the appropriate stationary distribution,¹² but which also (effectively) converges to the stationary distribution in a reasonable number of iterations (called the ‘burn-in’ period) and moves around the space of possible values of X quickly enough (called ‘good mixing’). Note that if we wish to simulate independent draws from f , it is necessary to thin the samples we obtain, leaving an appropriate interval between each sample we use, where the length of the interval will depend on the nature of the dependence in the underlying Markov chain.

¹¹ In fact, it is necessary only that we be able to evaluate the PDF/PMF up to a constant of proportionality.

¹² Hence, at a minimum the proposal distribution must produce an irreducible and ergodic Markov chain. In fact, the chain must also be *time-reversible*, in that the relation $f(i)q_{ij}\alpha_{ij} = f(j)q_{ji}\alpha_{ji}$ must hold for all i, j .

2.2.2 Principles of Statistical Inference

Statistical inference refers to the process whereby, given observations x_1, \dots, x_n of $X_1, \dots, X_n \sim F$, we attempt to extract information – or ‘learn’ – about F from the data \mathbf{x} . We may be interested in something as simple as a scalar summary characteristic of F , such as the mean μ_X , or in something as potentially complex as the entire CDF F itself, or its PDF, f . In many cases, our observations are of the form (y_i, x_i) and it is the relation between the random variables Y and X that is of interest. Here the variable Y is called the *response* or the *outcome*, while the variable X is called the *predictor* or the *feature*. Common inferential goals in this setting include learning about the value of $\mathbb{E}(Y|X = x)$, as a function of x , or foretelling a yet-to-be observed value Y_* from the observation $X_* = x_*$.

Statistical inference typically is conducted within the context of a model(s). A *statistical model*, broadly speaking, refers to the specification, either explicitly or implicitly, of a collection \mathcal{F} of CDFs to which F may belong. For example, it is common to specify models of the form $\mathcal{F} = \{f(x; \theta) : \theta \in \Theta\}$. If Θ is finite in dimension and does not grow in size with the sample size n , then \mathcal{F} is said to be a *parametric* model. An example is the model $\mathcal{F} = \{\text{All Gaussian densities with mean } \mu \text{ and variance } \sigma^2 : \mu \in \mathbb{R}, \sigma > 0\}$. On the other hand, if Θ is infinite in dimension or grows in size with n , then \mathcal{F} is called a *nonparametric* model.¹³ An example is the model $\mathcal{F} = \{\text{All CDFs } F \text{ with continuous density } f\}$.

Most introductory (i.e., ‘Stat 101’) treatments of statistics deal almost exclusively with parametric models. However, the use of nonparametric models has become increasingly common across the sciences. This fact has important implications, as nonparametric modeling can bring with it issues not encountered in parametric modeling. Such issues can arise in the analysis of network data, as we discuss below in Section 2.3.

Standard treatments of statistical inference typically characterize problems as falling into either of two broad categories: *estimation* and *hypothesis testing*. In addition, there is the related problem of *prediction*, which may be viewed as a variant of the estimation problem.

Estimation consists of making an ‘educated guess’ about the value of a parameter θ , such as a mean μ or a vector of coefficients β in a regression model $\mathbb{E}(Y|\mathbf{X} = \mathbf{x}) = \mathbf{x}^T \beta$, based on the observed data and any other information built into the model \mathcal{F} . Estimates may be *point estimates* (i.e., estimates taking on a single value $\hat{\theta}$) or *set estimates* (i.e., estimates consisting of a range of possible values over a set), such as an interval $(\hat{\theta}^{low}, \hat{\theta}^{high})$. Sometimes it is not a parameter in a model that is of interest so much as an unobserved random variable, such as the next value X_{n+1} in a sequence of random variables X_1, \dots, X_n or a new response value Y_* corresponding to an observed predictor $\mathbf{X}_* = \mathbf{x}_*$. This form of estimation is typically called prediction. When Y is a discrete random variable (e.g., $Y = \pm 1$), the predic-

¹³ The distinction between parametric and nonparametric models can be made more rigorous than this, but the definitions given here will suffice.

tion problem is called *classification*. Predictions too may take either point or set forms.

Hypothesis testing involves weighing the evidence contained in observations, say x_1, \dots, x_n , in support of a hypothesized theory. This theory is said to constitute the *null hypothesis* and is typically written as H_0 . Often one explicitly specifies an alternative to the null as well, the *alternative hypothesis*, which we will write as H_1 . For example, a canonical case is that of testing whether a parameter θ is equal to a particular value θ_0 or not (i.e., $H_0 : \theta = \theta_0$ is posited against $H_1 : \theta \neq \theta_0$). Performing a hypothesis test generally amounts to computing a suitable *test statistic* $t_{obs} = T(\mathbf{x})$ from the data and comparing it to a *reference distribution* F_T that describes the (random) behavior of the variable T in the case that H_0 is true. If t_{obs} is found to lie in the ‘extreme’ regions of the support of F_T e.g., $F_T(t_{obs})$ or $\bar{F}_T(t_{obs})$ are sufficiently ‘small,’ then the null hypothesis is rejected. Otherwise, the null is retained. In practice, a reference distribution may derive from previous measurements, analytical results (such as asymptotic theory), or computationally intensive re-use of the original sample.

For most inference problems that have been studied to any reasonable extent, there are likely a variety of methods that have been proposed. Differences among methods can be attributed to aspects ranging from slight variations in model specification (e.g., a variance is assumed known, versus unknown), to alternate technical ‘machinery’ (e.g., estimation through method of moments, versus maximum likelihood), to preferences in overall paradigm (e.g., frequentist versus Bayesian¹⁴). One’s choice of methodology is generally influenced by some combination of empirical performance, computational tractability, theoretical properties (if such have been derived), and, of course, personal preference.

2.2.3 Methods of Statistical Inference: Tutorials

Rather than attempting a general overview of various methods of statistical inference, which is clearly beyond the scope of this chapter, we will instead review be-

¹⁴ Traditionally, the two primary ‘schools of thought’ on inference in the statistics literature have been *frequentist* and *Bayesian*. In essence, they differ in (i) how they interpret probability, and (ii) how they view parameters. The frequentist interpretation of the probability $\mathbb{P}(A)$ is as the relative frequency at which the event A occurs in the limiting case of an infinite number of repetitions of an experiment and, as such, as an objective property of nature. A parameter θ is treated as a fixed but unknown constant. In particular, parameters are not treated as random. In contrast, formally, the Bayesian interpretation of the probability $\mathbb{P}(A)$ is as something that reflects a subjective level of belief in the occurrence of A . Accordingly, a parameter θ is viewed as a fixed quantity about which one nevertheless can make probability statements. Given an unknown parameter θ , with distribution F_θ (called the prior distribution), and data \mathbf{x} , with conditional distribution $F_{\mathbf{X}|\theta}$ (called the likelihood), inference on θ is conducted through statements involving the distribution $F_{\theta|\mathbf{X}}$ (called the posterior), which is defined through the use of Bayes rule. Informally, the Bayesian approach is often used as a convenient framework within which to combine models for both measurement (i.e., likelihood) and parameters (i.e., prior).

low just a handful of methods that will be most relevant to future chapters in this book, through the use of two tutorials. These tutorials serve to illustrate the chosen methods in the context of two classical settings: (i) inference of a mean based on a single sample, and (ii) regression and prediction with linear models. These two settings were chosen particularly for their likely familiarity, and will be useful as early as Section 2.3, when we contrast them with some of the settings to be seen later in the book in the analysis of network data.

2.2.3.1 Inference of a Mean

Suppose we have n iid observations x_1, \dots, x_n of a random variable $X \sim F$. We will consider the problem of performing statistical inference on the mean μ , arguably the most classical and canonical problem of its kind. While this problem is clearly important from a practical perspective, it also turns out that principles and techniques for this problem often lie at the core of methods of inference for more complicated parameters θ .

A natural estimator of μ is, of course, the sample mean $\bar{x} = \sum_i x_i/n$. While its usage can be motivated on intuitive grounds, we note that it is also a simple example of a *method of moments* estimator. That is, suppose our aim is to learn an unknown parameter θ in a model $\mathcal{F} = \{f_X(x; \theta) : \theta \in \Theta\}$. Denote the k -th moment of X as $\alpha_k(\theta) = \int x^k f(x; \theta) dx$. The method of moments dictates that we equate the first K moments, for a given choice of K , with their sample versions $\hat{\alpha}_k = \sum_i x_i^k / n$, and solve the resulting system of K equations for θ . Specifically, we form the system

$$\begin{aligned}\alpha_1(\theta) &= \hat{\alpha}_1 \\ \alpha_2(\theta) &= \hat{\alpha}_2 \\ &\vdots = \vdots \\ \alpha_K(\theta) &= \hat{\alpha}_K\end{aligned}\tag{2.18}$$

and set the estimator $\hat{\theta}$ to be that value of θ solving (2.18). For the problem of estimating a scalar mean μ , we have $\theta = \mu = \alpha_1$, and thus need only $K = 1$ equation (i.e., $\alpha_1(\theta) = \hat{\alpha}_1$). Solving this trivially yields the estimator $\hat{\mu}^{MM} = \bar{x}$.

The sample mean also is the estimator that results from applying the *maximum likelihood* principle, in the case that our model class \mathcal{F} assumes, say, a Gaussian distribution with mean μ and (known) variance $\sigma^2 = 1$. In maximum likelihood estimation, we estimate a parameter of interest θ by the value $\hat{\theta}^{ML}$ that maximizes the likelihood $\mathcal{L}(\theta)$, where

$$\mathcal{L}(\theta) = \prod_{i=1}^n f_X(x_i; \theta)\tag{2.19}$$

is the joint density of the X_i 's interpreted as a function of θ . This approach is equivalent to maximizing the (natural) logarithm $\ell(\theta) = \log \mathcal{L}(\theta) = \sum_i \log f(x_i; \theta)$, called

the *log-likelihood*, which is sometimes an easier object to deal with, both analytically and numerically. In the case that $\theta = \mu$ and the density f is the Gaussian density described above, we find that

$$\ell(\mu) = c - \frac{1}{2} \sum_{i=1}^n (x_i - \mu)^2 , \quad (2.20)$$

where c is a constant with respect to the maximization in μ . A small calculation confirms that (2.20) is maximized by $\hat{\mu}^{ML} = \bar{x}$.

In the same way that the data x_1, \dots, x_n are realizations of random variables X_1, \dots, X_n , the sample mean \bar{x} may itself be associated with a random variable, say \bar{X} . The distribution of this random variable is called the *sampling distribution*. The accuracy of \bar{X} can be characterized by its *mean squared error*,

$$\text{MSE}(\bar{X}) = \mathbb{E}(\bar{X} - \mu)^2 . \quad (2.21)$$

This quantity can be decomposed as $\text{MSE}(\bar{X}) = \text{bias}^2(\bar{X}) + \mathbb{V}(\bar{X})$, where $\text{bias}(\bar{X}) = \mathbb{E}(\bar{X}) - \mu$. Note that since $\mathbb{E}(\bar{X}) = \mu$, the sample mean has zero bias. That is, it is an *unbiased* estimator, which may be interpreted as saying that on average \bar{X} is ‘on target.’ The quantity $\mathbb{V}(\bar{X})$ captures the variability of \bar{X} and typically accompanies the sample mean through its role in defining the *standard error*, $\text{se}(\bar{X}) = \sqrt{\mathbb{V}(\bar{X})} = \sigma / \sqrt{n}$, where $\sigma^2 = \mathbb{V}(X)$. If σ is not known, an estimated standard error $\hat{\text{se}}$ can be formed by replacing σ by the standard deviation of the data.

Often the standard error is used in a more formal fashion to create, with the sample mean, an interval estimate of the form

$$C(\mathbf{x}) = (\bar{x} - z_{\alpha/2} \hat{\text{se}}, \bar{x} + z_{\alpha/2} \hat{\text{se}}) . \quad (2.22)$$

Here $z_{\alpha/2}$ is that value for which a standard normal random variable Z has $\mathbb{P}(-z_{\alpha/2} \leq Z \leq z_{\alpha/2}) = 1 - \alpha$. Under fairly general conditions, it may be asserted that $\mathbb{P}(\mu_X \in C(\mathbf{X})) \approx 1 - \alpha$, with the approximation improving with increasing sample size.¹⁵ The interval $C(\mathbf{x})$ in (2.22) is called a $100(1 - \alpha)\%$ *confidence interval*.

Now suppose that, rather than estimate μ , we wish to test the null hypothesis $H_0 : \mu = \mu_0$ versus the alternative, $H_1 : \mu \neq \mu_0$. A familiar test statistic is the Wald statistic

$$T(\mathbf{x}) = \frac{\bar{x} - \mu_0}{\hat{\text{se}}(\bar{x})} , \quad (2.23)$$

commonly called the *t-statistic* in this setting, which is just the observed \bar{x} shifted by its mean (or ‘centered’) under the null hypothesis and scaled by its estimated standard error. For moderately large n , the value $T(\mathbf{X})$ has a distribution approximately equal to that of a standard normal distribution.

¹⁵ Note that confidence intervals are a frequentist inference tool, and hence μ_X is considered fixed. The randomness here is inherent in the interval $C(\mathbf{X})$, and hence the corresponding probability statement is in regards to how likely the event is that the random C contains the unknown, fixed μ_X .

Standard practice is to compare the observed value $T_{obs} = T(\mathbf{x})$, for $\mathbf{X} = \mathbf{x}$, to the distribution of a standard normal variable Z in either of two manners. One may specify a value α , and report whether or not $|T_{obs}| \geq z_{\alpha/2}$ holds, rejecting H_0 if it does, and retaining H_0 , if not. Alternatively, one may simply report the probability $p = \mathbb{P}_{\mu_0}(|T(\mathbf{X})| \geq |T_{obs}|)$, where \mathbb{P}_{μ_0} indicates that the probability is calculated under the assumption that $\mu = \mu_0$. In the first case, the emphasis is on the test as a control procedure, and the probability of incorrectly rejecting H_0 is constrained to be (approximately) no more than α in value. The value α is called the *significance level* of the test. In the second case, the emphasis is on presenting some measure of the ‘weight of evidence’ against H_0 , since a simple conclusion of ‘reject’ or ‘retain’ is not very informative. The probability p is called the *p-value* and is the smallest significance level α at which H_0 would be rejected. Smaller values of p (e.g., p less than 0.05, 0.01, 0.001, etc.) indicates increasingly less support for the null hypothesis.

When there are multiple tests to be performed, it can become necessary to modify the manner in which we discuss error rates and control. This phenomenon is the so-called *multiple testing problem* (also referred to as the multiple comparisons problem). Suppose there are m independent samples $\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(m)}$, and m corresponding test statistics $T_j = T(\mathbf{x}^{(j)})$, one for each sample $j = 1, \dots, m$. While we know how to control each individual test so as to have an individual significance level α , in some settings it may be desirable to instead control some analogous notion of error for the family of m tests as a whole. Note that if each test T_j is controlled at the α significance level, then the probability of making at least one false rejection over the set of m tests is $1 - (1 - \alpha)^m$. For successively larger m , it is therefore necessary to use correspondingly smaller α for each individual test. The well-known *Bonferroni correction* procedure advocates using significance levels of α/m for each test, in an effort to crudely bound the overall error at a rate of α . Unfortunately, for large m this approach can lead to each individual test having very poor statistical power.¹⁶

A common approach to addressing this issue is to focus not on the family-wise error rate, but rather on the rate of false discoveries. The *false discovery rate* (FDR) is defined to be

$$\text{FDR} = \mathbb{E} \left(\frac{R_{false}}{R} \mid R > 0 \right) \mathbb{P}(R > 0) , \quad (2.24)$$

where R is the number of rejections among our m tests and R_{false} is the number of false rejections. Benjamini and Hochberg [33] provide a method for controlling the FDR at a user-specified level γ by rejecting the null hypothesis for all tests associated with p -values $p_{(j)} \leq (j/m)\gamma$, where $p_{(j)}$ is the j -th smallest p -value among the m tests. A popular variation on this method is that of Storey [370], in which the rate of *positive* false discoveries (i.e., pFDR) is the focus, where the pFDR is just the quantity in (2.24) without the probability $\mathbb{P}(R > 0)$. In this context, Storey proposes an analogue of the p -value for multiple testing, called the *q*-value, as well as a method for estimating q -values in practice. With q -values one rejects all null

¹⁶ The term *power* formally refers to the probability of rejecting the null hypothesis H_0 when the alternative H_1 is in fact true (i.e., the probability of detecting a deviation from the null).

hypotheses in a set of m tests for which the corresponding q -values are less than or equal to some user-specified q^* , with the expectation that approximately $q^* \times m$ of those rejections will be false discoveries.

Returning to the problem of estimating μ , let us examine a Bayesian approach.¹⁷ From the Bayesian perspective the parameter μ itself is equipped with a prior distribution, F_μ , and inference on μ is based on the posterior distribution $F_{\mu|\mathbf{X}}$. Suppose that, conditional on μ , the observations x_1, \dots, x_n are modeled as iid from a Gaussian distribution with (unknown) mean μ and known variance σ^2 , similar to the frequentist model above. Additionally, however, suppose that we model F_μ as Gaussian too, with mean a and variance b^2 . Our choice of a indicates our *a priori* belief about ‘typical’ values of μ , while the choice of b^2 reflects the strength of that belief.

The choice of Gaussian distribution for the prior is typically made for convenience, as it may then be shown that $F_{\mu|\mathbf{X}}$ (i.e., the conditional distribution of μ , given $\mathbf{X} = \mathbf{x}$) is itself Gaussian as well, with expectation $\bar{\mu}$ and variance τ^2 , where

$$\bar{\mu} = \mathbb{E}(\mu|\mathbf{X} = \mathbf{x}) = w\bar{x} + (1 - w)a , \quad (2.25)$$

for $w = (\text{se}^{-2})/(\text{se}^{-2} + b^{-2})$, and

$$\frac{1}{\tau^2} = \mathbb{V}^{-1}(\mu|\mathbf{X} = \mathbf{x}) = \frac{1}{\text{se}^2} + \frac{1}{b^2} . \quad (2.26)$$

Here se is short-hand for $\text{se}(\bar{X}) = \sigma/\sqrt{n}$.

The posterior mean $\bar{\mu}$ in (2.25) is a natural point estimate for μ , as it corresponds to the most likely value under the posterior¹⁸ (i.e., the mode). It takes the form of a weighted average of the sample mean \bar{x} and the prior mean a . The weights w and $1 - w$ reflect how the variance b^2 in the prior (corresponding to how certain we are in asserting that $\mu = a$) and the variance se^2 (summarizing the uncertainty in the data-based estimator \bar{X}) compare. Note that for n large enough, we will have $w \approx 0$, in which case $\bar{\mu} \approx \bar{x}$. That is, with enough observations, the information in the data will overwhelm that in the prior; for lesser amounts of data, however, the prior information plays an increasingly greater role.

Producing a Bayesian interval estimate of μ under our model, in analogy to a frequentist confidence interval, amounts to finding a set C such that $\mathbb{P}(\mu \in C|\mathbf{X} = \mathbf{x}) = 1 - \alpha$. The probability underlying the definition of C here is with respect to the posterior distribution,¹⁹ and that, as we have already remarked, is Gaussian. Therefore, a natural choice is the interval $C = (\bar{\mu} - z_{\alpha/2} \tau, \bar{\mu} + z_{\alpha/2} \tau)$. For large

¹⁷ There is also a Bayesian analogue of hypothesis testing, but we will not address that here, as it is somewhat more complicated to present and will play little to no role in the remainder of the book.

¹⁸ Additionally, it is the estimator $\bar{\mu}$ that minimizes the quantity $\mathbb{E}(\bar{\mu}(\mathbf{X}) - \mu)^2$, where the expectation is with respect to the joint distribution of μ and \mathbf{X} . As such, it is an example of a so-called *Bayes-optimal* estimator.

¹⁹ Note that under the Bayesian paradigm, μ is modeled as a random variable and hence, unlike with a frequentist confidence interval, the probability statement is in regard to how likely the event is that the random μ be in the fixed (conditional on $\mathbf{X} = \mathbf{x}$) interval C .

n , this interval will be very close to the $100(1 - \alpha)\%$ confidence interval given by $(\bar{x} - z_{\alpha/2} \text{se}, \bar{x} + z_{\alpha/2} \text{se})$.

On a final note, we point out that our illustration of doing statistical inference on a mean μ , and the use of the sample mean \bar{x} to do so, can be deceptive in its simplicity. Much of what has been described above generalizes in a fairly explicit manner to the case of estimating an arbitrary parameter θ , based on iid observations x_1, \dots, x_n of a random variable $X \sim F$. For example, given an estimator $\hat{\theta}$, we may still define the mean squared error $\text{MSE}(\hat{\theta}) = \mathbb{E}(\hat{\theta} - \theta)^2$, and the decomposition into bias and variance components is identical to that of the sample mean. Although not all estimates $\hat{\theta}$ will necessarily be unbiased, in many settings the bias is relatively small compared to the variance. However, $\text{se}(\hat{\theta}) = \sqrt{\mathbb{V}(\hat{\theta})}$ may not always be amenable to estimation by a simple ‘plug-in’ method. In such settings, an estimator based on ‘reuse’ of the sample, such as a bootstrap estimator, may be preferable. See the text by Efron and Tibshirani [131]. Similarly, under fairly general conditions, the quantity $(\hat{\theta} - \theta)/\text{se}(\hat{\theta})$ has a distribution that is approximately standard normal, for moderately large n , in the case of both method of moments and maximum likelihood estimators. This, therefore, means, for example, that confidence intervals of the form $(\hat{\theta} - z_{\alpha/2} \text{se}(\hat{\theta}), \hat{\theta} + z_{\alpha/2} \text{se}(\hat{\theta}))$ have approximate probability $1 - \alpha$ of containing θ , and test statistics of the form $T(\mathbf{x}) = (\hat{\theta} - \theta_0)/\text{se}(\hat{\theta})$ can be compared to $z_{\alpha/2}$ to test $H_0 : \theta = \theta_0$ versus $H_1 : \theta \neq \theta_0$ at the α significance level. Elements of the Bayesian methodology outlined above also generalize to a reasonable extent, but we will not go into detail here.

2.2.3.2 Linear Regression Inference

Consider now the situation where our data are of the form $(y_1, x_1), \dots, (y_n, x_n)$, being n iid observations of random variables (Y, X) with joint distribution $F_{Y,X}$. Suppose we wish to study the relationship between Y and X . A common approach to doing so is *regression*, in which the focus is on the *regression function*

$$r(x) = \mathbb{E}(Y|X = x) = \int y f_{Y|X}(y|x) dy . \quad (2.27)$$

We will review here various aspects of the *linear regression* framework, in which $r(x)$ is assumed to have a linear form. Not only is this framework classical and a still-standard part of the toolbox of most practitioners, but its underlying principles often remain at the core of more sophisticated methods of regression.

To begin, consider the *simple linear regression* model, which specifies that, given $X_i = x_i$, for each $i = 1, \dots, n$,

$$y_i = \beta_0 + \beta_1 x_i + \varepsilon_i , \quad (2.28)$$

where the ε_i are iid random variables for which $\mathbb{E}(\varepsilon_i|X_i = x_i) = 0$ and $\mathbb{V}(\varepsilon_i|X_i = x_i) = \sigma^2$. In other words, the y_i 's are modeled as ‘noisy’ measurements of the line

$r(x) = \mathbb{E}(Y|X = x) = \beta_0 + \beta_1 x$, where the noise terms are, conditional on the X_i 's, independent, with zero mean and constant variance.

Specification of a linear regression function relating Y_i and X_i reduces the problem of learning the function $r(x)$ to one of learning the coefficients (β_0, β_1) . Intuitively, a good choice of coefficients is that pair (β_0, β_1) for whom the distances from the values y_i to the corresponding line are minimized. That is, we seek to minimize the size of the residuals $y_i - (\beta_0 + \beta_1 x_i)$. Traditionally this is done through minimization of the *residual sum of squares*

$$\text{RSS} = \sum_{i=1}^n (y_i - (\beta_0 + \beta_1 x_i))^2 . \quad (2.29)$$

The corresponding values, say $\hat{\beta}_0$ and $\hat{\beta}_1$, that achieve this minimum are called the *least squares estimates* of β_0 and β_1 .

More generally, suppose that we observe iid data $(y_1, \mathbf{x}_1), \dots, (y_n, \mathbf{x}_n)$, where $\mathbf{x}_i = (x_{i1}, \dots, x_{ip})^T$ is now a $p \times 1$ vector. The *multiple linear regression* model specifies that, in analogy to (2.28),

$$y_i = \sum_{j=1}^p x_{ij} \beta_j + \varepsilon_i , \quad (2.30)$$

where $\mathbb{E}(\varepsilon_i | \mathbf{X}_i = \mathbf{x}_i) = 0$ and $\mathbb{V}(\varepsilon_i | \mathbf{X}_i = \mathbf{x}_i) = \sigma^2$. Typically the first element of the vectors \mathbf{x} is fixed at 1, which provides an intercept term for the model, the role played by β_0 in (2.28). Letting $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_n]^T$ be the $n \times p$ matrix obtained by stacking the n vectors \mathbf{x}_i in rows, and writing $\beta = (\beta_1, \dots, \beta_p)^T$, $\varepsilon = (\varepsilon_1, \dots, \varepsilon_n)^T$, and $\mathbf{y} = (y_1, \dots, y_n)^T$, the model in (2.30) can be re-expressed as $\mathbf{y} = \mathbf{X}\beta + \varepsilon$.

The model in (2.30) specifies the y_i to be ‘noisy’ measurements of the hyperplane defined by $r(\mathbf{x}) = \mathbf{x}^T \beta$, where the argument \mathbf{x} is now a $p \times 1$ vector. Therefore, we again seek a value for β that minimizes the size of the residuals. In this case, employing our matrix shorthand, the residuals may be expressed as $y_i - \mathbf{x}_i^T \beta$, and the residual sum of squares takes the form $\text{RSS} = (\mathbf{y} - \mathbf{X}\beta)^T (\mathbf{y} - \mathbf{X}\beta)$. A calculus argument shows that minimizing $\text{RSS} = \text{RSS}(\beta)$ in β yields the least squares estimate

$$\hat{\beta} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y} . \quad (2.31)$$

Note that $\hat{\beta}$ is only well defined if the $p \times p$ matrix $\mathbf{X}^T \mathbf{X}$ is invertible, which is true if and only if the $n \times p$ matrix \mathbf{X} is of full column rank p .

The estimate $\hat{\beta}$ is a linear combination of the elements of \mathbf{y} , which are realizations of a random vector \mathbf{Y} . Accordingly, it can be shown using the identities in (2.7), that $\mathbb{E}(\hat{\beta} | \mathbf{X}) = \beta$ and $\mathbb{V}(\hat{\beta} | \mathbf{X}) = \sigma^2 (\mathbf{X}^T \mathbf{X})^{-1}$. So the least squares procedure yields, under the model in (2.30), unbiased estimates²⁰ of the elements β_j in β , with standard errors $\text{se}(\hat{\beta}_j) = \sigma \sqrt{v_j}$, where v_j is the j -th element of the diagonal of $(\mathbf{X}^T \mathbf{X})^{-1}$. If

²⁰ In fact, the least squares estimator can be shown to have the smallest variance among all unbiased estimates of β .

σ is unknown, one may estimate the standard error $\text{se}(\hat{\beta}_j)$ by $\hat{\text{se}}(\hat{\beta}_j) = \hat{\sigma} \sqrt{v_j}$, where $\hat{\sigma}^2 = \text{RSS}(\hat{\beta})/(n-p)$ is an unbiased estimate of σ^2 .

Under fairly general conditions, for moderately large n , the sampling distribution of $\hat{\beta}$ is approximately that of a multivariate Gaussian²¹ with mean vector β and covariance matrix $\sigma^2(\mathbf{X}^T \mathbf{X})^{-1}$. This approximation is exact in the case that the errors ε_i in (2.30) themselves have (univariate) Gaussian distributions, conditional on their \mathbf{x}_i 's. In either case, the availability of a sampling distribution allows us to accompany our point estimate $\hat{\beta}$ with confidence intervals and hypothesis tests, if desired, in a manner similar to that we saw for making inferences on a mean μ through the sample mean \bar{x} . For example, a $100(1 - \alpha)\%$ confidence interval for β_j is given by

$$(\hat{\beta}_j - z_{\alpha/2} \hat{\text{se}}(\hat{\beta}_j), \hat{\beta}_j + z_{\alpha/2} \hat{\text{se}}(\hat{\beta}_j)) . \quad (2.32)$$

Similarly, a test of $H_0 : \beta_j = \beta_j^{(0)}$ versus $H_1 : \beta_j \neq \beta_j^{(0)}$, at significance level α may be performed by comparing the test statistic

$$T_j = \frac{\hat{\beta}_j - \beta_j^{(0)}}{\hat{\text{se}}(\hat{\beta}_j)} \quad (2.33)$$

to the value $z_{\alpha/2}$.

When prediction is of interest, we may obtain a prediction of an unobserved value $Y_* = y_*$ from a given \mathbf{x}_* in a straightforward manner using a plug-in approach. Specifically, we predict y_* with $\hat{y}_* = \mathbf{x}_*^T \hat{\beta}$. One may also define a notion of standard error for this prediction, and predictive intervals. The calculations are similar, but not identical to, those underlying the standard error and interval estimates for a parameter. They differ in that, to describe the variability in predicting the random variable $y_* = \mathbf{x}_*^T \beta + \varepsilon_*$, one must account for both the variability in estimating β and the variability due to not knowing ε_* . See Wasserman [392, Ch. 13.4] for details.

Least squares estimation, while generally motivated from a linear algebraic or geometric perspective, can also be arrived at through application of the maximum likelihood method. Specifically, assume that the conditional distributions $F_{\varepsilon_i | \mathbf{X}_i}$ of the ε_i , given $\mathbf{X}_i = \mathbf{x}_i$, are independent Gaussians, with mean zero and variance σ^2 . Due to (2.30), this implies that the conditional distributions $F_{Y_i | \mathbf{X}_i}$ of the random variables Y_i , given their respective $\mathbf{X}_i = \mathbf{x}_i$, are also independent Gaussians, but with mean $\mathbf{x}_i^T \beta$ and variance σ^2 . That is, of the form

$$f_{Y_i | \mathbf{X}_i}(y | \mathbf{x}) = (2\pi\sigma^2)^{-1/2} \exp \left\{ -(y - \mathbf{x}^T \beta)^2 / (2\sigma^2) \right\} . \quad (2.34)$$

²¹ A $p \times 1$ random vector $(X_1, \dots, X_p)^T$ is said to have a multivariate Gaussian density, with $p \times 1$ mean vector μ and $p \times p$ covariance matrix Σ if

$$f(\mathbf{x}) = ((2\pi)^p |\Sigma|)^{-1/2} \exp \left\{ -(1/2)(\mathbf{x} - \mu)^T \Sigma^{-1} (\mathbf{x} - \mu) \right\} .$$

In principle, application of the maximum likelihood method here requires us to maximize the joint density²² $f_{\mathbf{Y}|\mathbf{X}}$ as a function of the unknown parameters (β, σ) . However, this density may be written as the product of the conditional density $f_{\mathbf{Y}|\mathbf{X}}$ and the marginal density $f_{\mathbf{X}}$, and only the conditional density is a function of these parameters. So it is sufficient to consider only the conditional density, as a *conditional likelihood*,

$$\mathcal{L}(\beta, \sigma) = \prod_{i=1}^n f_{Y_i|\mathbf{X}_i}(y_i|\mathbf{x}_i) . \quad (2.35)$$

Writing out (2.35), using (2.34), and taking logarithms, we find that the conditional log-likelihood of (β, σ) is given by

$$\ell(\beta, \sigma) = -\frac{n}{2} \log(2\pi) - n \log(\sigma) - \frac{1}{2\sigma^2} \sum_{i=1}^n (y_i - \mathbf{x}_i^T \beta)^2 . \quad (2.36)$$

Maximizing (2.36) in β involves only the last term, and can be seen to be equivalent to minimizing the quantity $\text{RSS}(\beta)$, thus resulting in the least squares estimate $\hat{\beta}$ in (2.31). Additionally, maximizing (2.36) in σ^2 can be shown to yield the estimate $\text{RSS}(\hat{\beta})/n$, which for large n is nearly the same as the unbiased estimate $\text{RSS}(\hat{\beta})/(n-p)$.

This likelihood-based perspective on linear regression is also useful in facilitating a Bayesian approach to the problem. Suppose, for example, that in addition to our Gaussian assumption on the conditional distribution of the errors ε_i , we also model the elements β_j of β as independent Gaussian random variables with common mean zero and variance τ^2 . Then it may be shown that the posterior distribution $F_{\beta|\mathbf{Y}, \mathbf{X}}$ of β , given the data, is multivariate Gaussian with a log-density of the form

$$\log f(\beta|\mathbf{Y}=\mathbf{y}, \mathbf{X}) \sim - \sum_{i=1}^n (y_i - \mathbf{x}_i^T \beta)^2 - \lambda \sum_{j=1}^p \beta_j^2 , \quad (2.37)$$

where $\lambda = \sigma^2/\tau^2$, and the symbol ‘~’ indicates here that terms not involving β have been omitted.

The expression in (2.37) is maximized, as a function of β , by the value

$$\hat{\beta}^{ridge} = (\mathbf{X}^T \mathbf{X} + \lambda \mathbf{I})^{-1} \mathbf{X}^T \mathbf{y} . \quad (2.38)$$

The value in (2.38) is the mode of the posterior and, since the posterior is Gaussian, the mean. It has been proposed from a non-Bayesian perspective under the name of *ridge regression*, where it is called a *ridge estimator*. Note that in the event that $\lambda = 0$, we simply recover the least squares (and maximum likelihood) estimator $\hat{\beta}$ in (2.31). This corresponds to the case where $\tau^2 \rightarrow \infty$, indicating a lack of any real information from the prior.

²² Here as a sub-script, and similarly below, in a slight abuse of notation, \mathbf{X} refers to the random vector $(X_1, \dots, X_p)^T$.

For $\lambda > 0$, the estimate $\hat{\beta}^{ridge}$ will differ from $\hat{\beta}$. In particular, it will be biased, with the bias increasing as λ increases. On the other hand, note for example that the ridge estimator is well-defined even when \mathbf{X} is not of full rank, unlike the estimator in (2.31). In exchange for this bias, there is the potential to decrease the variance below that of the least squares estimator. This is often possible, for example, in cases where \mathbf{X} is nearly rank-deficient, since then the variances in $\mathbb{V}(\hat{\beta})$ can be quite large, due to instability in the inversion of the matrix $\mathbf{X}^T \mathbf{X}$. If the trade off between bias and variance is managed fruitfully, the result can be an estimator with an overall reduction in mean-square error (i.e., with $MSE(\hat{\beta}^{ridge}) < MSE(\hat{\beta})$). The nature of the trade off depends on the value of λ , which may be set either subjectively (in a purely Bayesian manner) or through data-adaptive methods.

More generally, the optimization of (2.37) to produce (2.38) can be viewed as just one instance of a larger class of procedures falling under the name of *complexity penalized least squares*, where estimators are defined through

$$\hat{\beta}^J = \arg \max_{\beta} \text{RSS}(\beta) + \lambda J(\beta) . \quad (2.39)$$

That is, we estimate β by that value which optimizes a penalized version of the least squares criterion $\text{RSS}(\beta)$, where $J(\cdot)$ is an arbitrary penalty function and $\lambda > 0$ is a tuning parameter. Adjusting the magnitude of λ controls the balance between the terms $\text{RSS}(\beta)$ and $J(\beta)$ in selecting $\hat{\beta}$.

There are a variety of choices for J . Like ridge regression, they effectively introduce bias into the estimation of β , in exchange for a reduction in variance. However, unlike ridge regression, some choices of J are designed to actively eliminate elements of the vectors \mathbf{x}_i from the model, through assignment of the values $\hat{\beta}_j^J = 0$ to ‘unimportant’ variables X_j . That is, such methods perform *model selection* in addition to estimation. These choices are particularly useful in settings where p is large, compared to n . Note too that, while some choices of J allow for analytic solutions to $\hat{\beta}^J$, such as the choice $J(\beta) = \beta^T \beta$ underlying ridge regression, in most cases it is necessary to solve (2.39) using numerical optimization software.

The topic of linear regression modeling has been developed in extraordinary detail and is still a mainstay of statistical modeling. On the other hand, there are entire classes of problems for which linear regression models can be at least unsatisfactory, if not just plain inappropriate. Such problems typically involve nonlinearity in the variables X , and non-Gaussianity, or even non-additivity, of the noise variables ε . Nevertheless, various aspects of linear regression models are often present to a nontrivial extent in the regression models that have been developed to deal with issues like these, which can be useful for understanding and interpreting such models. For example, generalized linear models essentially extend the concept of maximum likelihood estimation in linear models to other likelihoods, which includes the class of logistic regression models used in classification of binary response variables Y . Similarly, many techniques in the area of kernel-based modeling, including those methods based on support vector machines, arguably consist of mapping original

variables X to a new, larger space, applying established linear statistical methods, and then representing the result of those methods in the space of the original variables. On the other hand, the extent to which the inferential machinery for such other model classes is developed can often lag behind that of the class of linear models, sometimes decidedly so. Methods of estimation and prediction generally accompany the initial proposal of a model class, but practical methods for quantifying the uncertainty in such inferences, for example in the form of confidence and prediction intervals, may be lacking, due to analytical or computational hurdles, or both.

2.3 Statistical Analysis of Network Data: *Prelude*

The unique relational nature of network data means that one often can encounter certain challenges in statistical analysis that are not present in ‘standard’ statistics problems, like those covered in introductory statistics courses. Two major sources of these challenges are (i) the dependency of the quantities involved and (ii) their dimensionality. The effects of these two are often further compounded by the sheer volume of data obtained in many network analysis studies. We close this chapter with a set of three short examples illustrating our point, each of which is a vignette of a larger set of topics to be covered later in the book.

Example 2.1 (Inference of Gene Association Networks). Recall from Section 1.2.3 that the study of patterns of regulatory association among genes in organisms is a topic of fundamental interest to scientists in genetics. At an elementary level, such associations may be thought of as corresponding to the extent to which one gene plays a role in ‘activating’ or ‘suppressing’ the activity of another gene (indirectly through the actions of the protein product(s) produced in accordance with its genetic code), across all pairs of genes. Current technology allows us to obtain, fairly cheaply and efficiently, an indication of the relative level of activity of genes (i.e., their level of ‘expression’), through the use of DNA micro-array chips. With micro-arrays, one can simultaneously measure the expression levels of all genes in an organism, under various experimental circumstances. If a pair of genes has expression levels that are sufficiently similar across experiments, that may be suggestive of a regulatory association between the two. However, just what criteria should be used to judge similarity?

Formally, we may suppose we have a set of N_v genes $V = \{1, \dots, N_v\}$, and for each gene $i \in V$, we measure a vector of expression levels $\mathbf{x}_i = (x_{i1}, \dots, x_{in})^T$ under n separate experimental conditions. A simple but commonly used measure of similarity between a pair of genes $i, j \in V$ is the correlation, $\hat{\rho}_{ij}$, between the vectors \mathbf{x}_i and \mathbf{x}_j . Genes for which $\hat{\rho}_{ij}$ is judged to be ‘large’ (either positively or negatively) are judged to be associated with each other, and this is indicated by assigning the pair $\{i, j\}$ to an edge set E . The resulting graph $G = (V, E)$ then represents an inferred gene regulatory network.

In order to judge when $|\hat{\rho}_{ij}|$ is ‘large’, it is compared to a threshold t . In so doing, one is implicitly conducting a test of the hypothesis $H_0 : \rho_{ij} = 0$ against the

alternative $H_1 : \rho_{ij} \neq 0$, where ρ_{ij} can be interpreted as the correlation of expression levels between i and j across the universe of relevant experimental conditions. Such a test in fact does not differ much in principle from the test for the value of a mean μ based on the Wald statistic in (2.23). However, we are not conducting just one test here, but rather approximately N_v^2 tests simultaneously! In other words, we find ourselves faced with a multiple testing problem, with the number of tests scaling with the dimensionality of the network. In addition, note too that there is a built-in dependency among our test statistics, by virtue of correlating the vector \mathbf{x}_i of expression levels for each gene with that of every other gene.

As we will see in Chapter 7, the importance of the multiple testing problem here is in the critical impact it has on the threshold t used and, more fundamentally, on the manner in which one seeks to control errors in inferring the presence or absence of edges in the underlying network graph. Methods based on FDR and similar principles are one approach to this problem. \square

Example 2.2 (Modeling Social Ties). Sociologists have long been interested in understanding what factors influence the development of social ties among actors. Social theory posits many such potential factors. A central tenet of social network analysis is that it is possible to assess the relevance of a given factor(s) on the social dynamics of a population of interest through the statistical modeling and analysis of quantitative measurements of the underlying social network. A well-known example is *homophily* – the tendency for entities to associate with others like them, popularized in the common parlance through the phrase, “Birds of a feather flock together.” The presence of homophily has been confirmed in numerous social network contexts, often along the lines of the types of characteristics one might expect, such as age, gender, and social class or role.

A common technique for assessing the presence of social factors like homophily in a social network is to construct network regression models, with social ties as response variables Y and measurements relevant to the social factors as predictors X . More precisely, suppose that $G = (V, E)$ is a graph corresponding to an observed social network among individuals $i \in V$, with a social tie (e.g., friendship) between individuals $i, j \in V$ indicated by an edge $\{i, j\} \in E$. Let $Y_{ij} = Y_{ji} = 1$ if $\{i, j\} \in E$, and zero if not. That is, $\mathbf{Y} = [Y_{ij}]$ is just the adjacency matrix for G , treated as a random matrix. A regression model for the Y_{ij} ’s might then specify a form for $\mathbb{P}(Y_{ij} = 1 | \mathbf{X}_i = \mathbf{x}_i, \mathbf{X}_j = \mathbf{x}_j)$, where \mathbf{X}_i contains measurements on the i -th individual, such as age or gender.

We note, however, that often social theory hypothesizes factors that can be interpreted directly in terms of the presence or absence of a social tie as a function of other social ties. Homophily, for example, implies that ‘friends of my friends are my friends’. Therefore, it can be expected that there is information on the event $\{Y_{ij} = 1\}$ in the values of the other random variables

$$\mathbf{Y}_{(-ij)} = \{Y_{kl} : \{k, l\} \neq \{i, j\}\} . \quad (2.40)$$

Thus, in building a social network regression model one may want to also include the variables $\mathbf{Y}_{(-ij)}$ as predictors for the Y_{ij} , and specify models for probabilities of the form $\mathbb{P}(Y_{ij} = 1 | \mathbf{Y}_{(-ij)} = \mathbf{y}_{(-ij)}, \mathbf{X}_i = \mathbf{x}_i, \mathbf{X}_j = \mathbf{x}_j)$.

Regression models of this form fall under the class of *auto-regression* models, in that – unlike standard regression models – the response variable Y also plays a role as a predictor variable. This type of model, analogous to those encountered in time series analysis and the analysis of spatial data, explicitly seeks to capture the type of dependency often inherent in relational data. In addition, such models will potentially have a high number p of parameters, unless simplifications are made (e.g., assumptions of homogeneity of effect sizes across pairs $\{i, j\}$). We will examine models like these and others in more detail in Chapters 6 and 8. \square

Example 2.3 (Estimation of an Internet ‘Traffic Matrix’). It was mentioned in Section 1.2.1 that Internet service providers (ISPs) have a vested interest in monitoring the traffic on the Internet networks they maintain. One fundamental quantity that providers wish to monitor is the volume of data sent from each origin node in the network to each destination node. An example would be the number of bytes flowing between an origin and a destination in, say, a five-minute interval. If one organizes such traffic flows in a matrix form, say with origins for rows and destinations for columns, the result is what the Internet measurement community calls a *traffic matrix*.

If we represent a given network by a graph $G = (V, E)$, and assume for simplicity that all vertices can serve as both origins and destinations, then we see that it is necessary to monitor N_v^2 flows to obtain the traffic matrix of this network. It has traditionally been viewed as prohibitive to deploy throughout the network the technology infrastructure (i.e., measurement and communication devices) necessary to measure these flows directly. Easier to measure is the traffic flowing over the N_e links in the network. But such measurements are less useful for monitoring, as the flow over each link essentially consists of a super-position of traffic flows for all origin-destination pairs utilizing that link.

Interestingly, this observation can actually be exploited in a useful manner. Let \mathbf{x} be a $N_v^2 \times 1$ vector of traffic flows observed in a given time period for all pairs of origins and destinations in G . Similarly, let \mathbf{y} be a $N_e \times 1$ vector of the flows observed over all links during the same time period. Additionally, let \mathbf{B} be a $N_e \times N_v^2$ binary matrix, with the element in the e -th row and ij -th column taking the value $B_{e;ij} = 1$ if edge e is traversed in going from the i -th origin to the j -th destination, and 0 if not. This is called the *routing matrix*, as it describes the routing of traffic over G . The super-position of origin-destination flows over links suggests the relation $\mathbb{E}(\mathbf{Y}|\mathbf{X} = \mathbf{x}) = \mathbf{Bx}$. This in turn seems to indicate that one might predict \mathbf{x} from the measurements $\mathbf{Y} = \mathbf{y}$ by setting up a linear regression model $\mathbf{y} = \mathbf{Bx} + \varepsilon$, and solving for \mathbf{X} .

Unfortunately, it is easy to see that this problem will typically be ill-posed, since there are N_v^2 values to be learned from only N_e measurements, and generally N_e is substantially smaller than N_v^2 . In other words, we have what looks like a standard regression problem, but with $p \gg n$, in which case there is no unique least squares

solution. A good deal of success has been had in solving this problem, however, when the least squares problem is replaced by an appropriately specified complexity penalized least squares problem, a problem of the form

$$\hat{\mathbf{x}} = \arg \max_{\mathbf{x}} [(\mathbf{y} - \mathbf{Bx})^T (\mathbf{y} - \mathbf{Bx}) + \lambda J(\mathbf{x})] , \quad (2.41)$$

for some penalty $J(\cdot)$. We will see additional details on this and related problems in Chapter 9. \square

2.4 Additional Related Topics and Reading

A more thorough introduction to the topic of graph theory may be found in any of a number of introductory textbooks, such as those by Bollobás [42], Diestel [118], or Gross and Yellen [186]. Details on graph data structures and algorithms are in many computer science algorithms texts. See the text by Cormen, Leiserson, Rivest, and Stein [100], for example, from which most of the material of Section 2.1.4 was taken. For statistics and probability background, the text by Wasserman [392] is useful for its combination of coverage and conciseness, and we have drawn heavily on it for the presentation of material in Sections 2.2.1 and 2.2.2. In addition, Wasserman cites many other standard texts on specific topics. A more careful and complete exposition on the topic of probability alone, including Markov chains, may be found in standard introductory texts on the subject, such as that by Ross [333]. For more complete developments of the methods on simulation of random variables we described, and other related methods, see the texts by Liu [264] or Robert and Casella [325].

Exercises

2.1. For the graph in Figure 2.4, complete the following.

- a. Determine a walk from vertex 2 to vertex 7 that is not a trail.
- b. Determine a trail from vertex 2 to vertex 7 that is not a path.
- c. Determine a path from vertex 2 to vertex 7.
- d. How many paths are there from vertex 1 to vertex 7? What is the geodesic distance between these two vertices? Is there a unique shortest path between them?
- e. How many connected subgraphs of order four can be found? How many of these subgraphs contain a cycle of length at least three? Of length four?

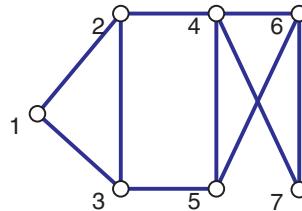


Fig. 2.4 Graph corresponding to Exercise 2.1.

2.2. For a network software package of your choice,²³ familiarize yourself with the basic aspects of its data structures and algorithms. For example, using the graph in Figure 2.4, create an appropriate input file(s), according to the format(s) available (e.g., adjacency, lists, etc.). Additionally, using the same graph as input, explore some of the basic algorithmic capabilities of your software by, for instance, calculating all shortest paths among all pairs of vertices, a minimal spanning tree from a given vertex, etc.

2.3. Recall the definition of the Laplacian $\mathbf{L} = \mathbf{D} - \mathbf{A}$ of a graph G , in Section 2.1.3, where \mathbf{A} is the adjacency matrix and $\mathbf{D} = \text{diag}[(d_i)_{i \in V}]$ is a diagonal matrix defined in terms of the degree sequence $\{d_i\}_{i \in V}$. An alternative form of the Laplacian – typically called the *normalized Laplacian* – is the matrix defined as

$$\tilde{\mathbf{L}} = \mathbf{D}^{-1/2} \mathbf{L} \mathbf{D}^{-1/2} = \mathbf{I} - \mathbf{D}^{-1/2} \mathbf{A} \mathbf{D}^{-1/2} .$$

- a. Both types of graph Laplacian can be shown to have non-negative eigenvalues, and minimal eigenvalue equal to zero, with corresponding eigenvector $\mathbf{1}$. In addition, the eigenvalues of $\tilde{\mathbf{L}}$ can be shown to all fall in the interval $[0, 2]$. For the graph shown in Figure 2.4, compute \mathbf{L} and $\tilde{\mathbf{L}}$ and verify numerically that these properties hold.
- b. For a connected graph G , the second smallest eigenvalue of \mathbf{L} and $\tilde{\mathbf{L}}$ will be non-zero, and its corresponding eigenvector is often found to be useful for partitioning G into two cohesive subgraphs.²⁴ A simple approach is to annotate each vertex in G with the sign of the corresponding entry in this eigenvector. Do this for the graph shown in Figure 2.4. Is the graph partitioned in a sensible manner?

²³ For an (incomplete) list of relevant network-related software tools and packages, consult the website for this book at <http://math.bu.edu/people/kolaczyk/SAND.html>.

²⁴ We will examine this usage of the Laplacian more formally later, in Chapter 4.3.3.

- c. Try modifying the graph in Figure 2.4, by adding and deleting edges and vertices, and repeating parts (a) and (b). How do the eigenvalues and the eigenvector corresponding to the first non-zero eigenvalue behave under these changes? What happens if your graph becomes disconnected?

2.4. Suppose that $\mathbf{X} = (X_1, \dots, X_n)^T$ is an n -length random vector, with expectation $\mathbb{E}(\mathbf{X}) = \mu$ and covariance matrix $\mathbb{V}(\mathbf{X}) = \Sigma$. Let $\bar{X} = \sum_{i=1}^n X_i/n$ be the average of the elements of \mathbf{X} . For this problem, you may wish to use the expressions in (2.7).

- a. If the X_i all have a common mean, say $\mu_i = \alpha$, show that $\mathbb{E}(\bar{X}) = \alpha$.
 b. Show that the variance of \bar{X} is given by the expression

$$\mathbb{V}(\bar{X}) = n^{-2} \left[\sum_{i=1}^n \Sigma_{ii} + 2 \sum_{i < j} \Sigma_{ij} \right] .$$

- c. In the case where the elements of \mathbf{X} are uncorrelated and have common variance, and hence $\Sigma = \text{diag}[(\sigma^2, \dots, \sigma^2)]$, for some $\sigma^2 > 0$, show that the expression in part (b) for the variance of \bar{X} reduces to σ^2/n .

2.5. Recall the ridge regression estimator

$$\hat{\beta}^{ridge} = (\mathbf{X}^T \mathbf{X} + \lambda \mathbf{I})^{-1} \mathbf{X}^T \mathbf{Y}$$

in equation (2.38).

- a. Show that $\hat{\beta}^{ridge}$ can be obtained by maximizing (2.37) as a function of β , which is equivalent to solving

$$\frac{\partial}{\partial \beta} \log f(\beta | \mathbf{Y} = \mathbf{y}, \mathbf{X}) = 0 .$$

- b. The data in Table 2.1 were generated according to the model

$$y_i = 1 + 1 \times x_{i1} + 1 \times x_{i2} + 0 \times x_{i3} + \varepsilon_i ,$$

for $i = 1, \dots, 10$, where the x_{i1} are independent draws from a Gaussian distribution, with mean zero and unit variance, the x_{i2} are generated in the same manner, and independently of the x_{i1} , and the x_{i3} are formed by adding a small amount of noise to the averages $(x_{i1} + x_{i2})/2$. The errors ε_i are independent, among themselves and of the x variables, and follow a Gaussian distribution with mean zero and variance $(0.01)^2$.

Fit a linear regression to these data for Y as a function of X_1 and X_2 only. Then fit a linear regression for the full model (i.e., with X_3 included as well). The estimates you obtain for the coefficients of X_1 and X_2 and their corresponding standard errors should be quite different in the two cases, due to the fact that when X_3

Y	X_1	X_2	X_3
2.292	1.209	-0.472	0.3588
2.068	-1.577	2.291	0.3377
0.926	-0.605	1.231	0.3030
-1.419	-0.689	-1.047	-0.8682
4.451	0.814	0.585	0.7039
0.679	-1.681	0.617	-0.5338
3.226	0.304	1.441	0.8917
2.220	-0.202	0.360	0.0736
-0.649	-1.510	-1.262	-1.3820
-2.008	-1.081	-0.323	-0.7049

Table 2.1 Data for Exercise 2.5.

is in the model the three predictor variables are nearly linearly dependent, and therefore $\mathbf{X}^T \mathbf{X}$ is nearly singular.

The effect of the term $\lambda \mathbf{I}$ in $\hat{\beta}^{ridge}$ is to stabilize the operation of matrix inversion required by the least squares solution. Calculate the ridge regression estimates $\hat{\beta}^{ridge}$, in the model with X_1, X_2 , and X_3 , for various choices of λ , and compare your estimates to those obtained by least squares. How close can you get your estimates to the true value $\beta = (1, 1, 1, 0)^T$ as you change λ ?

Chapter 3

Mapping Networks

Given a complex system under study, one natural goal is to create a network-based representation of the system and to present that representation in the form of a visual image. The process by which such images are produced involves a number of distinct steps, with important decisions to be made along the way, and in fact is not unlike the production of cartographic maps. In this chapter, we address the topic of ‘mapping’ networks.

3.1 Introduction

As in many other areas of the quantitative sciences, visual imagery plays a fundamental role in network analysis. Such has been the case since even before the advent of computers and computer graphics. Freeman [158] has described, for example, how the initial impetus for the use of visualization in social network analysis can be traced to the seminal work of Moreno and colleagues in the 1930’s. These and other early efforts at network visualization were in the form of hand-drawn, annotated graphs and related diagrams of relational data. The computer brought with it the potential to automate, and the motivation to standardize, the drawing portion of this process. Nevertheless, network visualization remains a nontrivial task, for at least two main reasons.

First, in most contexts there is not necessarily one, single network graph representation for a given system of interest – where by ‘network graph representation’ we refer here to a network graph G and any accompanying weights, annotations, and labels. Phrased another way, we may say that to speak vaguely of “*the network*” for a system is often misleading. There may in fact be many potentially relevant network representations, not unlike the way that a given geographical region may be represented in many different ways (e.g., with respect to its topography, its population patterns, its geology, its land use by humans, etc.). Second, even given a particular network graph representation of a complex system, visualization of that representation is still rendered nontrivial by the substantial challenges inherent in effectively

communicating the information in a network graph using just the two-dimensional surface of a printed page or a computer display.

A graph G is just a pairing of two particular sets – a vertex set and an edge set – which itself is lacking in any inherent geometry. A visualization of G can be thought of informally, therefore, as a mapping from the space of such pairs to two- or three-dimensional Euclidean space. Hence, there is both an enormous amount of flexibility in defining such mappings and a correspondingly diverse range of information that may be communicated, or lost, as the case may be. In this sense, the challenges faced here are related to many of those faced in the visualization of high-dimensional datasets – or indeed in the visualization of information quite generally, as eloquently discussed and illustrated by Tufte [382], for example. However, this area is still arguably quite young and research is quite active, with aspects of mathematics, algorithms, aesthetics, and the workings of the human visual system all playing roles.

There is a strong analogy to be made between the creation of a network-based visualization for a complex system, on the one hand, and a cartographic map of a geospatial region on the Earth, on the other. As a result, the former process is more and more frequently being referred to through phrases such as ‘network mapping.’ It is convenient to identify, broadly speaking, three key stages in the production of network maps:

- (i) collection of data from a system of interest;
- (ii) creation of a network graph representation from the data; and
- (iii) rendering of the representation in the form of a visual image.

These stages are illustrated schematically in Figure 3.1. In addition, there is a temptation to define a critical but typically difficult, and often ignored, fourth stage, that of assessing the representative-ness of a network map, a process referred to as ‘validation’ in geography.

In this chapter, we will consider each of the three stages listed above in turn, in Sections 3.2 through 3.4. We will then examine a pair of case studies in Section 3.5, illustrating the overall mapping process in two different contexts – mapping ‘Science,’ in the field of scientometrics, and mapping the Internet. Lastly, the topic of mapping dynamic network graphs will be discussed in Section 3.6. The issue of validation will be revisited again briefly at the end of this chapter; a related discussion, in the context of network inference, is the topic of Chapter 7.

3.2 Collecting Relational Network Data

Network mapping effectively begins with the collection of data from a system that is to be mapped. Whether we collect the data ourselves, or someone else does, an understanding of the data collection protocol is imperative. Of course, what data are to be collected as well as the tools and techniques for their collection vary widely from discipline to discipline. Furthermore, the quality and quantity of data that we

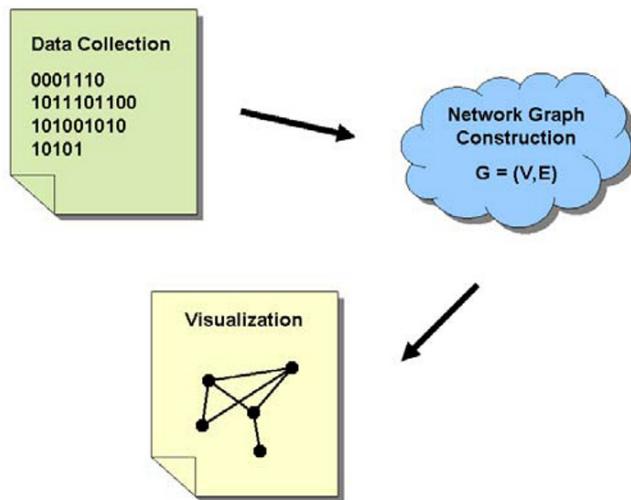


Fig. 3.1 Schematic illustration of the process of mapping a network.

can realistically expect to collect differ among disciplines as well. Our aim here in this section will be to survey some of the common aspects associated with, and typical issues faced in, network data collection across disciplines.

3.2.1 Measurement of System Elements and Interactions

In order to produce a network visualization, it is necessary to have a network graph representation. But it is not generally the case that network graphs are obtained explicitly themselves. Rather, more often than not they are constructed from a set of more basic measurements on some elements and their interactions in an underlying system of interest. Therefore, the choice of what is meant by ‘elements’ and ‘interactions,’ as well as what measurements to take for each, is important, since ultimately this choice influences not only the network graph(s) that might be constructed but also *a fortiori* the analyses of such graphs and hence the conclusions that may be drawn thereof. We illustrate this point through the following example.

Example 3.1 (Two Alternative Perspectives on Drosophila’s Circadian Clock Mechanism). Recall from Chapter 1 the network shown in Figure 1.3 for the circadian clock mechanism in the organism *Drosophila*. As mentioned in the text accompanying this figure, there are actually a number of relevant units and sub-processes depicted. The green rectangles (e.g., *Tim*, *Per*, *dCLK*, etc.) represent specific proteins, the basic biological molecules underlying this process. Through the various

lines and arrows drawn, the proteins are shown to be serving at least two important purposes. First, some of them physically interact with each other, such as to form more sophisticated complexes of proteins. Complexes like these are shown with the corresponding green rectangles stacked upon each other, as with *Dbt* and *Per* in the middle of the figure. Second, some of the proteins, in attaching to specific locations on the organism's DNA, near particular genes, serve to regulate (i.e., activate or repress) the 'expression' of these genes; that is, they regulate the production of the specific proteins for which these genes code. This activity is shown in the figure by short paths of two lines, from the original protein, to a circle with 'DNA' next to it, to another protein.

Hence, there are at least two sets of 'elements' and 'interactions' of potential interest here: (i) proteins, and their affinity for interacting with each other, and (ii) genes, and their regulation among themselves. Each may be represented in the form of a small network graph, as shown in Figure 3.2. Note, however, that the two graphs, although depicting two highly inter-related aspects of the overall circadian clock mechanism, are nonetheless different in many ways. For example, while the protein *Per* interacts with no fewer than four other proteins, the gene coding for *Per* regulates none of the other genes directly.

Current high-throughput methods in genomics allow scientists to obtain measurements that provide – separately – evidence of protein interaction (e.g., protein affinity experiments) and gene regulation (e.g., DNA micro-array experiments). In conjunction with various methods of network inference (some of which we will see in Chapter 7), these measurements allow for the construction of network graphs of protein interaction and gene regulation on large scales. However, as we have seen here, each only provides a partial view of what transpires in the underlying biological system, and hence biologists are necessarily cautious in their interpretation and usage of such networks as they seek to disentangle the biological workings of different organisms. □

So different notions of 'elements' in a system, even if closely related, can produce quite different network graphs. However, even with a single, well-defined notion of what 'elements' will mean in a given study, there may be choices left to be made, such as when there exist different 'scales' at which the elements can be labeled. In studying the topology of the Internet, for example, such as that underlying the Abilene map in Figure 1.1, we might consider the elements to be individual users, or machines and devices (which, depending on their placement in the Internet, can serve varying numbers of users), or groups of such users/machines, such as those falling under the purview of a given administrative unit (e.g., a so-called autonomous system, or AS). Alternatively, in a study of citation patterns in the scientific literature, we might consider individual authors, or the papers they write, or the journals in which the papers are published, or even the disciplines that are represented by the journals. (We will return to each of these examples through the case studies in Section 3.5.)

With the elements of interest chosen, the relevant 'interactions' are sometimes relatively pre-determined, but not necessarily. In the case of Internet topology, for example, as mentioned previously, we might be interested in the topology of the

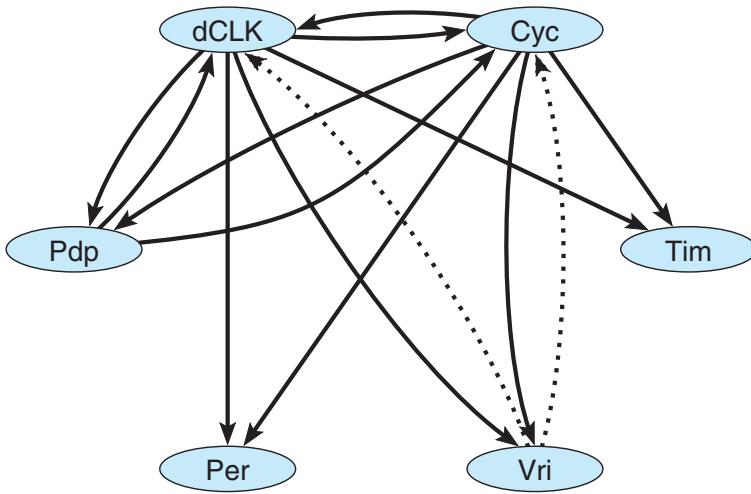
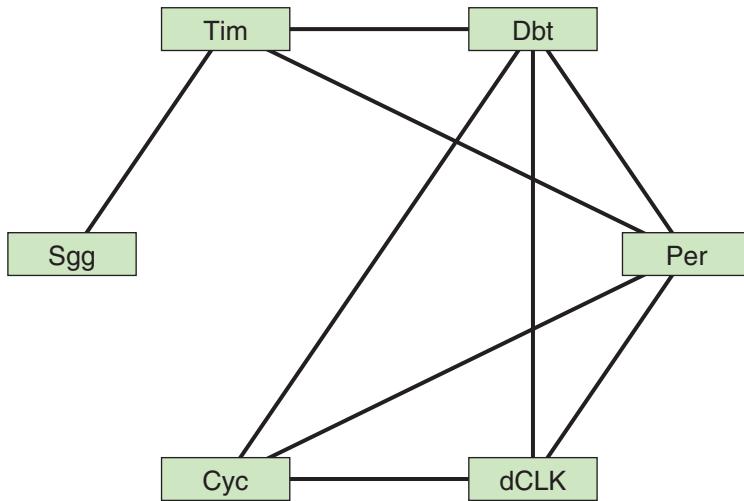


Fig. 3.2 Network graph representations of protein binding (top) and gene regulation (bottom) associated with the circadian clock mechanism in *Drosophila melanogaster* (fruit fly), as extracted from the composite representation in Figure 1.3. Proteins are shown as vertices in the form of green rectangles, and their interactions, by undirected edges. Genes, coding for their respective proteins, are shown as vertices in the form of blue ellipses. The regulation of one gene by another is depicted by a directed edge from the latter to the former (solid indicates activation, and dotted, repression).

literal, physical Internet, or rather, of the effective, logical Internet. Similarly, in studying citation patterns, say, among journals in a given literature, we might use some measure of the frequency of citations of articles in one journal by those in another (termed ‘inter-citation’), or rather some measure of the frequency with which two journals cite common papers (termed ‘co-citation’).

Measurements of interactions can take many forms (e.g., binary, counts, or continuous values), and can be either direct or indirect in nature, in the sense that they may capture the intended notion of ‘interaction’ to a greater or lesser extent. For example, in the context of friendship networks in social network analysis, we may have the opportunity to interview the actors in a group and directly ask them whether or not they feel they are friends with each other actor (i.e., binary response), or instead we may (perhaps by choice) observe the frequency of particular types of interactions (i.e., count response). Note that even in the case of the seemingly direct measurement (i.e., asking “Who are your friends?”) there can be issues, in that the veracity of the response is not guaranteed, the notion of ‘friendship’ held by actors may not be the same, or it may not coincide with that of the scientist.

Ultimately, it is worth keeping in mind with network studies – as with any study involving assessment of association – that not only is what we choose (or are capable of) to measure in a system important, but also, potentially, so too is what remains unmeasured in the system.

3.2.2 *Enumerated, Partial, and Sampled Data*

So far in this chapter we have talked as if we could collect whatever data we set our eyes upon. This is an overly optimistic perspective. Instead, it is important to distinguish between network data that are enumerated, partial, or sampled. To facilitate our discussion, and borrowing terminology from statistical sampling theory, we will picture the collection of all possible system elements that we would like to measure as a finite set and refer to that as a ‘population,’ with the elements themselves referred to as ‘units.’

Enumerated data are data collected in an exhaustive fashion from the full population. In some network analysis studies it is possible to collect data in this manner. For instance, in some social network studies, such as those that might involve the dynamics among members of small groups, with full compliance from the group members, we could view the data collected as exhaustive. Network data on friendships among children in a given school are a possible example, as is the network of interactions among members of the karate club in Zachary’s study, described in Section 1.2.2. Similarly, in the analysis of citations in scientific journals of a given discipline (e.g., medicine, physics, etc.), thanks to database resources like the Scientific Citation Index, PubMed, and others, it is possible to collect relevant citation information exhaustively as well, at least within certain windows of time.

Partial data are data that derive from a full enumeration of only a subset of the population. Note that, depending on the context and the perspective of those con-

ducting a given study, many datasets that are apparently enumerated may be better considered, for some purposes, as partial. Consider the Abilene network shown in Figure 1.1. The Internet is a global technological network, but in this figure we have only shown a sub-network pertaining to the United States. And in fact, that sub-network in turn is a network that supports the transit of Internet data primarily for users only in U.S. educational and research institutions. Hence the presence of other networks, such as analogous commercial U.S. networks or networks in other countries, is largely absent from this map. To the extent that Abilene is a coherent sub-network under the watchful eye of a single administrative unit, such a perspective may be quite acceptable for many purposes. On the other hand, given that part of the traffic on this network is from Abilene users to and from locations throughout the rest of the world, for some questions ignoring the partial-ness of the network can be misleading. For instance, an observed disruption in traffic patterns in New York may be found to have little to do with any of its immediate Abilene neighbors, but rather to be due to the propagation of disruptions from, say, its European counterpart, Géant.

The concern posed by partial network data is closely related to that posed by finite regions in the analysis of spatial data and the danger of potential ‘boundary effects’ (e.g., see Cressie [102]). In the Abilene example above this connection is of course particularly evident, since the network itself has a strong geo-spatial aspect to it.

Sampled data are data on units selected from the population through some random mechanism (either implicit or explicit). Like partial data, sampled data are only some subset of the possible data to be collected, but unlike partial data, sampled data will not *a priori* yield an exhaustive view of a well-defined sub-population. In many areas of network analysis, sampling is the rule rather than the exception. An example of sampled data is the AIDS blog data in Chapter 1.2.4, where the random choice of a three-day period of observation induces a random selection of bloggers. There are many types of network sampling designs, and it is generally important to understand the nature of the design underlying the collection of network data, as sampling can affect analyses at later stages in ways that are potentially both nontrivial and subtle. The topic of network sampling is substantial enough that we will wait to develop it more fully on its own in Chapter 5.

The three categories of data we have used above – enumerated, partial, and sampled – are convenient, but the reality of data collection can be more nuanced, combining aspects of more than one of these. Alternatively, it can be useful instead to think in terms of *observed data* and *missing data*, a perspective that has been found to be quite powerful in many contexts in statistical analysis generally (e.g., see Little and Rubin [263]), particularly over the past few decades. The missing data paradigm has evolved into an organized approach to categorizing mechanisms for missing-ness of data, studying their effects on a given analysis, and, when possible, adjusting for such effects. However, unfortunately, it seems to have received much less attention so far in the field of network analysis, with most efforts to date found in the context of social networks and relevant primarily to the types of structural

analysis methods we will see in Chapter 4. See Kossinets [237], for example, and citations therein.

The discussion above is included here at this point in the book primarily as a cautionary note, with the basic message being that the manner in which the data are collected can be important for their proper analysis and the interpretation of such analyses. At a minimum this suggests that a certain sense of *caveat emptor* (i.e., ‘buyer beware’) be maintained, by both producers and users of networks and network analyses – even visual analyses like those to be considered next in this chapter. At a more involved level, we may want to incorporate the mechanism(s) of data collection into the analysis, using models like those discussed in Chapters 5, 6, and 7. Modeling can also be useful, as we will see, for attempting to account for the effects of *measurement error*, a phenomenon distinct from those discussed so far. An obvious measurement error of interest in network analysis is the false declaration of the presence or absence of interactions among measured units. The mechanisms for such errors typically are inherent in the process of data collection.

3.3 Constructing Network Graph Representations

We turn now to the task of constructing a network graph representation from a set of network measurements. At its most basic, the representation will consist of just a graph $G = (V, E)$, with vertex set V and edge set E . However, more generally, the representation may also include additional auxiliary information, such as a set of edge weights $\{w_e\}_{e \in E}$, providing some indication of the strength of association between vertices, vectors $\{x_v\}_{v \in V}$ of variables describing vertex attributes, and labels. Attribute variables may be discrete in nature, such as those indicating class membership (e.g., gender in social networks), or continuous (e.g., size of a population in the geographical region serviced by an airport in an airline network). It will be this sort of information specifically that we seek to incorporate into a map, using methods like those to be described in the next section.

Having collected a set of network measurements, in principle the construction of a network graph G consists ‘simply’ of assigning vertices v to measured elements (or units) and edges $e \in E$ to measured interactions between elements.¹ However, even at this stage there may be nontrivial decisions to be made.

Consider just the issue of specifying vertices. In rendering a map on a printed page or a computer screen, both the space and the resolution available will be finite. For network graphs with more than a few hundred vertices, maps can quickly become cluttered. So the unit of analysis may still become an issue. As we shall discuss more in Section 3.4, we can choose to deal with this problem by displaying only a relevant sub-graph(s). On the other hand, if it is not clear *a priori* just which

¹ If it is desired to assign a direction to the measured interactions, then it will be a digraph that is constructed, with the set E consisting of arcs from one vertex to another. As the treatment of digraphs in this section is analogous to that of undirected graphs, we will confine our discussion here to undirected graphs.

sub-graph(s) to display, or if it is felt that there is value to be had in displaying the entire graph in some form or another, then some reduction of the vertex set may be necessary. Aggregation of vertices is one approach, particularly when there is some relevant sense of scale inherent in the possible choice of vertex definitions, such as in the Abilene map in Figure 1.1.

Now consider the related issue of specifying edges. As mentioned earlier in Chapter 2, the number of edges N_e in measured network graphs frequently has been found to scale linearly with the number of vertices N_v . So if the number of vertices to be displayed is problematic, the task of displaying a similar number of edges will likely be no less so. In fact, since the number of edges can, of course, scale as much as quadratically (i.e., $N_e \leq \binom{N_v}{2}$), we can find ourselves faced with the problem of having too many edges to cleanly display even if the number of vertices does not seem unreasonable. Graph visualizations suffering from excessive density of edges often are justifiably described as looking like a ball of yarn.

The ball-of-yarn phenomenon may be addressed through various means. For example, again, when there is some relevant sense of scale underlying the measurements, choice of scale may be used effectively. In the Abilene network, or more generally in many networks with a connection to the Earth's geography, such as various transportation networks, vertices tend both to be spatially localized and to have few if any interactions with vertices outside the immediate locale. As a result, spatial aggregation will lead to both aggregation of vertices and thinning of edges. This effect is apparent in the map of Abilene and its blow-out, in Figure 1.1. Viewed more formally, rooted sub-trees or DAGs at a given depth in an original candidate graph, say G_0 , are trimmed at their root, effectively replacing the entire sub-structures (both vertices and edges) with a single vertex in G .

Alternatively, vertex annotations may sometimes be used to separate an initial, dense candidate network $G_0 = (V_0, E_0)$ into a handful of less dense networks, say G_i , for $i = 1, \dots, I$. For example, if the annotations are indicators of category or group membership, it may be more visually informative to display the sub-networks of those vertices and their interactions associated with each category or group. In biology, for instance, it can be useful to associate genes or proteins with their biological function(s), when these functions are known. The resulting graphs then describe the gene regulatory relationships or protein interactions among just those genes or proteins involved with a given function. Larger networks are then sometimes built up from these smaller networks by incorporating related functions together. More formally, this process of thinning corresponds to selecting only a certain subset of vertices of V_0 , say V_i , as those associated with characteristic or group i , and defining G_i to be the corresponding induced subgraph.

As a final example, when the edges in a candidate graph G_0 are defined according to some measure of similarity among vertices, the edges in G_0 may be thinned by re-assessing the strength of measured interaction judged necessary to merit an edge. For example, it is common to assign edges according to whether or not the correlation between attribute vectors of a given pair of vertices exceeds some threshold in absolute value. Adjustment of the threshold therefore clearly moderates the density of edges.

Note that in all three of the techniques described above, and in any other similar techniques, fundamentally a choice is being made in summarizing the extent of ‘interaction’ or ‘relation’ contained in the original measurements that is to be communicated in the corresponding visualization. More sophisticated approaches to reducing an initial candidate graph G_0 to one better suited for visualization involve the use of techniques that summarize graph structure. An illustration, based on the use of clustering, will be presented in Section 3.5.1. These techniques themselves will be discussed in detail in Chapter 4.

3.4 Visualizing Network Graphs

Up until this point in this chapter we have spoken only loosely of displaying network graphs, the final component of the process illustrated in Figure 3.1. Here in this section we consider the problem of display in its own right, which turns out to be far more than a simple matter of plotting the data. Techniques for displaying network graphs fall under the field of *graph drawing* or *graph visualization* and incorporate a combination of elements from mathematics, human aesthetics, and algorithms. Our discussion below is broken into two parts, beginning with an overview of the elements of graph visualization and followed by a brief description of some of the more common graph visualization problems and their solutions.

3.4.1 Elements of Graph Visualization

Suppose we have a set of network measurements that have been encoded in a network graph representation $G = (V, E)$, and we now wish to summarize G in a visual manner. At the heart of the graph visualization problem is the challenge of creating “geometric representations of … combinatorial structures,” [117] using symbols (e.g., points, circles, squares, etc.) for vertices $v \in V$ and smooth curves for edges $e \in E$. For human consumption it is most convenient, of course, if a graph is drawn² in two-dimensional space, as opposed to three-dimensional space or on some more abstract surface. Hence, we will restrict our attention to this setting.

Intuitively, it is not hard to see that there are uncountably many ways that we could lay down a candidate set of points and curves on paper to represent a graph G . The important question for any such candidate, however, is whether or not it adequately communicates the desired relational information in G . While in principle this might suggest the drawing of graphs by hand, in practice hand drawings are only realistic for very small graphs. Generally graphs of nontrivial size must be drawn, at least in part, using automated methods.

² Here and throughout we use terms like ‘draw’ only in the colloquial sense, although more formal mathematical treatments of this topic area exist (e.g., see Chapter 8 of Gross and Yellen [186]) which attach more specialized understandings to these terms.

Of course, one approach, which in fact is used frequently, is to just lay out all vertices in G as equi-spaced points around a circle, and to draw straight lines between points for vertices connected by an edge. Unfortunately, although such drawings are simple enough to create, they tend to accentuate any tendency towards the ball-of-yarn phenomenon mentioned earlier. In order to facilitate improved automatic drawings of graphs, various specifications or requirements have evolved – some firm, and some flexible – which have been formally categorized as drawing conventions, aesthetics, and constraints. See di Battista, Eades, Tamassia, and Tollis [117].

Drawing conventions are basic requirements that a given drawing must satisfy. For example, we might specify that only straight line segments be used to draw edges, rather than polylines or smooth curves, that vertices and edges be positioned only along an underlying lattice, or that no edges intersect (i.e., that G be drawn planar). For trees or, more generally, for directed acyclic graphs, it is common to use an ‘upward’ (or ‘downward’) convention, in that all edges are drawn as a curve increasing (or decreasing) along the vertical axis.

While drawing conventions are hard requirements, in the sense that they must be satisfied, *aesthetics* are soft, in that they are specified with the intention that they be satisfied to the extent possible, all else being equal. For example, by definition a planar graph G may be drawn using edges that do not intersect anywhere other than at vertices, but such graphs are comparatively rare. Therefore, if edge crossings must be tolerated, it is natural to wish for them to occur as infrequently as possible. Similarly, we might wish to minimize the area used for the drawing, or the aspect ratio, or the total or maximum edge lengths, or the total or maximum number of bends edges may take, and so on.

Finally, *constraints* are requirements that pertain to subgraphs $H \subset G$, rather than the overall graph G itself. These might specify the relative placement of a vertex in the drawing, or a cluster of vertices, the direction of a path, etc.

Together, drawing conventions, aesthetics, and constraints effectively serve as parameters for automatic graph drawing methods, and the determination of a graph drawing becomes a formal optimization problem. Such optimizations frequently are difficult to solve exactly in real time for graphs that are nontrivial in size. Therefore, it is common to develop computationally efficient algorithms that seek an approximate solution, often through the use of heuristics and the imposition of priorities among aesthetics. Such algorithms are complicated enough, and software (both commercial and non-commercial) plentiful enough, that for most users it makes little sense to code the algorithms oneself. All of the network visualizations presented in this book, for example, were constructed using pre-existing software implementations. See Appendix A of the volume edited by Kaufmann and Wagner [221] for a fairly extensive list of packages recommended by those authors.

On a final practical note, we point out that graph drawing – like statistics itself, perhaps – involves not only ‘science’ but also some ‘art.’ Few automatically generated graph drawings cannot be improved upon through post-processing ‘by hand’.

3.4.2 Methods of Graph Visualization

There are far too many methods of graph visualization for us to present a full survey here. But there are two broad themes under which a large portion of such methods can be argued to fall, namely (i) methods for drawing graphs with special structure, and (ii) methods for graph drawing based on physical analogies. We will focus on describing and illustrating a handful of examples falling under each. We will also discuss extensions relevant to visualizing large graphs and directed graphs. References to more complete treatments of this topic area may be found in Section 3.7.

3.4.2.1 Drawing Graphs with Special Structure

Two special structures whose drawing has received a large amount of attention are planar graphs and trees. Graphs that are planar are especially relevant traditionally to certain important network areas, such as the design of circuits. Networks with a strong connection to the Earth's geography, such as some energy or transportation networks, also are often easily associated with planar graphs. The Abilene network and its representation in Figure 1.1 is another example.

However, planar graphs are also a natural starting point conceptually in the study of graph drawing, being by definition free of any edge crossings. In fact, many graph drawing algorithms for non-planar graphs build upon algorithms for drawing planar graphs, by first drawing a planar subgraph of the non-planar graph, and then augmenting that drawing to display the full graph. Two common techniques for the layout of planar graphs are to use orthogonal paths for edges (i.e., changing direction only by 90-degree turns in the manner of a canonical circuit drawing) or to use k -sided convex polygons for each cycle of length k in the graph. In other words, blocks and polygons, respectively, become the geometric structures upon which the visualization of the graph, as a combinatorial object, is built.

Layout algorithms for planar graphs are generally quite computationally efficient, running in as little as linear time in the number of vertices (i.e., $O(N_v)$ time). Note, however, that underlying the algorithms for drawing planar graphs can also be algorithms for determining whether a candidate graph G has various required properties, starting of course with the property of planarity itself. Algorithms exist for testing whether or not a graph is planar in linear time, but other related problems, such as certain constructions of planar subgraphs, are prohibitively expensive to solve exactly (i.e., NP -complete) and their solution is generally approximated.

Trees, although themselves planar graphs, are both distinct enough and encountered commonly enough that there is a separate body of methods for their drawing. Trees are used, for example, in the representation of various hierarchical structures, such as organizational charts and genealogies. In fact, it is this very notion of hierarchy that is the special aspect of trees that we generally wish to communicate effectively in their display. Figure 3.3 shows examples of three common layouts for rooted trees using layering of edges, horizontal and vertical edges (i.e., so-called hv -

layout), and edges that radiate outward on concentric circles (i.e., so-called ‘radial layout’), respectively. Algorithms for all three layouts run in linear time.

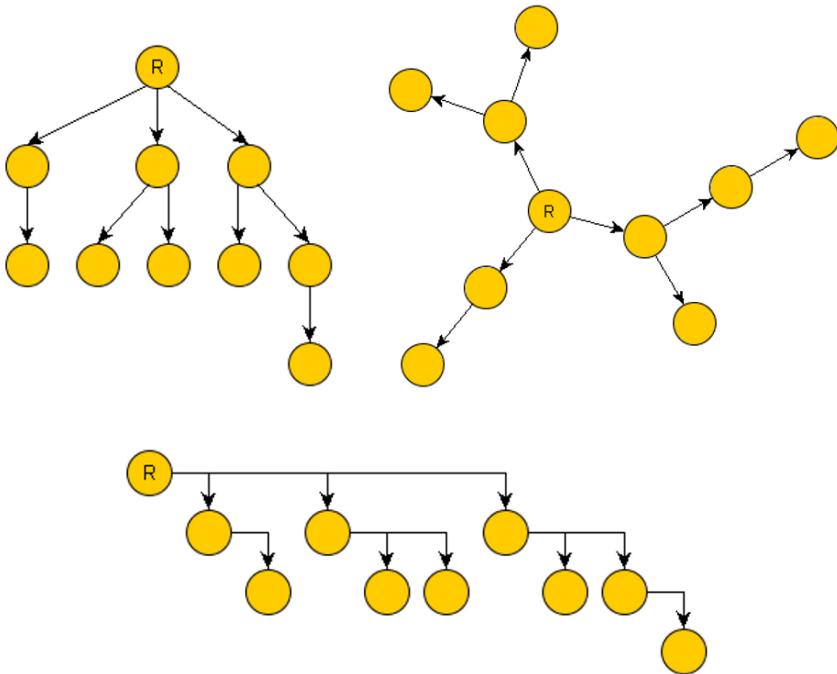


Fig. 3.3 Three displays of the same tree, rooted at the vertex labeled ‘R’: layered (top left), circular (top right), and horizontal-vertical (bottom).

3.4.2.2 Drawing Graphs Using Analogies to Physical Systems

If a graph G does not have a special structure, a popular and effective set of approaches to creating useful drawings is based on exploiting analogies between the relational structure in graphs and the forces among elements in physical systems. The expectation is that attraction and repulsion between ‘likes’ and ‘not likes,’ respectively, inherent in the organization of much of the natural world around us, if similarly used to represent a graph, will provide a familiar context within which to interpret the underlying relational information.

One approach in this area, and the earliest proposed, is to introduce attractive and repulsive forces by associating vertices with balls and edges with springs. If a literal system of balls connected by springs is disrupted, thereby stretching some of the springs and compressing others, upon being let go it will return to its natural state. So-called *spring-embedder* methods of graph drawing define a notion of force

for each vertex in the graph depending, at the very least, on the positions of pairs of vertices and the distances between them, and seek to iteratively update the placement of vertices until a vector of net forces across vertices converges. The method of Fruchterman and Reingold [164] is a commonly used example of this type, and underlies the network visualization in Figure 1.4. It runs nominally in $O(N_v^2)$ time per iteration, but typically various devices are employed to both speed up the run time and decrease the number of iterations until convergence.

Alternatively, motivated by the fact that it is possible to associate the collection of forces in spring systems with an overall system energy, a second approach in this area is that of *energy-placement* methods. An energy, as a function of vertex positions, ostensibly is defined using expressions motivated by those found in physics. That vertex placement is chosen which minimizes the total system energy. A physical system with minimum energy is typically in its most relaxed state, and hence the assertion here is that a graph drawn according to similar principles should be visually appealing. Methods based on multidimensional scaling (MDS), which have a long history in the social network literature, are of this type. The method of Kamada and Kawai [217] is a popular variant of MDS methods and was used to draw the network visualization of the karate club data in Figure 1.2.

An advantage of energy-placement methods, in specifying vertex placement as a formal optimization, is that standard methods of numerical optimization may be brought to bear. Typically (modified) Newton-Raphson algorithms are used, or eigensystems are solved, depending on the choice of energy function, and hence run in $O(N_v^2)$ time. Alternatively, a number of algorithms have been introduced to compute approximations of these graph layouts in essentially linear time (e.g., Brandes and Pich [56], Ganser, Koren, and North [165]). Another advantage to the energy-placement perspective is that it lends itself well to the formal inclusion of additional requirements (e.g., aesthetics and constraints), through the definition of more complex energy functions. There is generally a trade off between the complexity of the energy functional, as various requirements are incorporated, and the ease with which the corresponding optimization may be performed.

3.4.2.3 Drawing Large Graphs

Despite their sophistication, for all of the methods described so far the graph drawings still tend to look increasingly cluttered as the number of vertices N_v nears 100 or so, due to the finiteness of the available space and resolution. As was discussed above, in Section 3.4.1, if faced with cluttering, we may wish to go back and modify the network graph G itself. Alternatively, specific information to be communicated by the network visualization might suggest that only a relevant subgraph(s) be shown. For example, sometimes it is useful to highlight the structure local to a given vertex, such as in the so-called *egocentric* network visualizations commonly used in the social network literature, which show the vertex, its immediate neighbors, and all edges among them. An illustration is shown in Figure 3.4, based on the karate club network of Figure 1.2. A clever compromise between global and local network

visualizations is the use of ‘fish-eye’ views, similar to that achieved by wide-angle lenses in photography, wherein the network local to a vertex is shown in detail, but with the rest of the network shown as well, in successively less and less detail farther away from that vertex.

3.4.2.4 Drawing Directed Graphs

All of the discussion in this section up to this point has focused only on the drawing of undirected graphs. The drawing of directed graphs tends to be more involved. Standard techniques employ layering (as mentioned earlier in the context of trees), with the basic framework going back to Sugiyama, Tagawa, and Toda [375]. To draw a digraph in a layered fashion with, say, up-to-down orientation, the following requirements are generally incorporated: (i) the avoidance of arcs pointing upward; (ii) a relatively even placement of vertices and uniform length of arcs; (iii) as few crossings of arcs as possible; and (iv) reasonably vertical arcs. Common algorithms approach the enforcement of these requirements in a stagewise fashion and frequently employ heuristics in place of attempting to solve some of the demonstrably difficult (i.e., *NP*-hard) computational problems that arise in this context as a result of the directed-ness of the graph. The reader is referred to the resources described in Section 3.7 for details.

Before closing our discussion of graph drawing, it cannot hurt to again point out the importance of the human hand in this process. The production of sophisticated network maps is most often an iterative and interactive process. In fact, the better software packages for network visualization allow for at least some degree of interactive mapping. In addition, automatic methods of graph display typically are not set up to make decisions on the incorporation of additional attributes and labels of vertices and edges with a basic network representation G , when available. User choice of symbols, colors, and line thicknesses and types, for example, can all be used productively to better communicate such information in the original network data.

3.5 Case Studies

In order to better illustrate network mapping as a coherent process, and the challenges faced at the different stages described above, we present in this section two cases studies, both involving the mapping of very large-scale systems. The first case study presents an example of the mapping of ‘Science,’ while the second describes the construction of a particular map of the Internet.

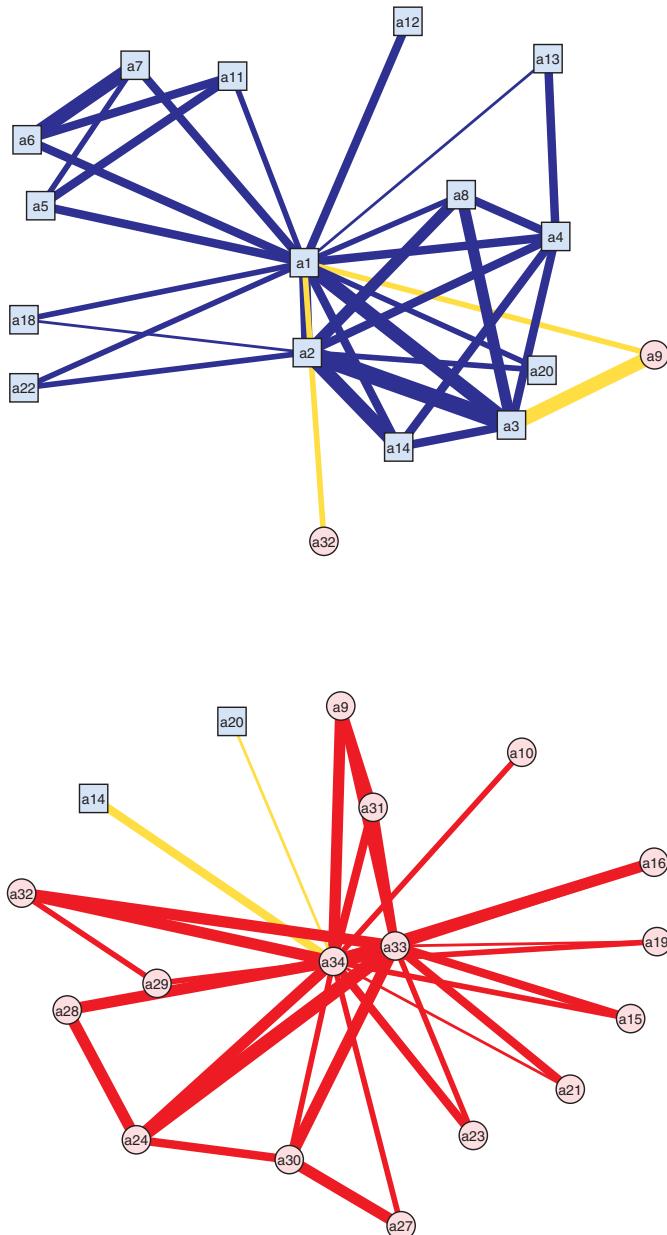


Fig. 3.4 Two ego-centric views of the karate club network of Figure 1.2, from the perspectives of vertices 1 (top) and 34 (bottom), the authority figures about which the club ultimately split.

3.5.1 Mapping ‘Science’

The human enterprise of Science and Technology (which we shall simply call ‘Science’ throughout) is both vast and multi-faceted, and is therefore of significant interest to members of many communities – such as groups and agencies within government, industry, and the sciences themselves. In particular, it is useful to be able to identify and understand patterns and associations in the growth and development of the various branches of Science. The field of *scientometrics*, whose foundations go back at least 40 years, to the work of Price [112] and others, aims to do so by applying the very techniques of scientific analysis to Science itself.

In this case study, we summarize the work of Boyack, Klavans, and Börner [50], whose goal was to produce a map of the ‘backbone’ of Science. Mirroring the development of material in the earlier part of this chapter, we organize our presentation in three parts: measurement, network graph construction, and visualization.

3.5.1.1 Measurement

It is common in scientometrics to study collections of written documents, in the form of articles published in archival journals, as a concrete representation of the knowledge produced by scientific endeavors. If we view such collections of articles as the ‘system’ to be measured, there are a number of natural choices of units. One choice would be the articles themselves, while another would be the journals within which the articles are published. The latter represents a coarser scale of unit than the former, resulting in a noticeably smaller (but still not at all small!) network, and is what we will use here. Ultimately, in the maps produced below, journals will be clustered and clusters of journals will be replaced by units of individual scientific disciplines.

We note that it is also not uncommon in scientometric studies to use the authors of articles as units. But this choice would be more relevant to the goal of mapping the patterns of co-authorship among individuals in a given scientific discipline, perhaps from a social network perspective, rather than that of mapping the relationships among the disciplines themselves.

Having chosen journals as our units, consider the task of defining a notion of interaction among journals that will allow us to meaningfully quantify the pattern of underlying scientific relations we wish to map. Measures based on citations are the standard choice. The citation of one paper by the authors of another is an explicit and universally accepted manner of acknowledging a connection between the two. In addition, detailed citation information for a large fraction of the journals across disciplines is now available in reasonably accessible form in various citation databases. To measure the strength of the citation patterns between two journals, we will use inter-citation frequencies. That is, for each journal pair (i, j) , the strength of their relationship will be quantified by the number of times C_{ij} that journal i in a given time period (e.g., one year) cites journal j in any time period.

Note that in choosing to use citation databases, there is an aspect of partial sampling that will enter into our measurement process, in that such databases do not necessarily include all journals published in all disciplines. For most disciplines, we can imagine that the impact of partial sampling on our goal of mapping the ‘backbone’ of Science will likely be minimal, because presumably human experts have chosen what are felt to be the most visible and important journals. Yet for some disciplines, such as Computer Science, where much of the ‘high-impact’ publishing is done in the form of refereed conference proceedings papers, which are excluded from many of the main citation databases, this partial sampling is potentially a concern.

Boyack, Klavans, and Börner chose to use databases provided by the Institute of Scientific Information (ISI). Specifically, a complete set of records on 1.058 million articles were obtained from 7,349 journals in these databases for the year 2000. Of the 7,349 journals, only 7,121 of them appeared at least once as both citing and cited, and of the 23.08 million citations within the 1.058 million articles, only roughly 16.24 million of them (or 70%) were to journals similarly within this database. From these 16.24 million citations among these 7,121 journals, the corresponding inter-citation frequencies C_{ij} were calculated. The resulting matrix of frequencies was in fact extremely sparse, with 98.6% of the entries equal to zero.

In addition to the citation information, Boyack, Klavans, and Börner also obtained the ISI journal category assignments, which were used to, among other things, produce disciplinary labellings of clusters on the final maps shown later in Figures 3.5 and 3.6.

3.5.1.2 Network Graph Construction

A network graph G can be defined directly from the inter-citation frequency matrix, where journals i and j , serving as vertices, have an edge (i, j) between them if and only if the sum of their inter-citation frequencies (i.e., $C_{ij} + C_{ji}$) is non-zero.³

However, Boyack, Klavans, and Börner found that this measure of similarity did not yield graphs that, according to a validation study, clustered journals in a manner consistent enough with human expectations, as summarized through the ISI categories assigned to each journal. Therefore, a number of other similarity measures were considered, each transforming the inter-citation frequency matrix in a different manner. The so-called Jaccard coefficient (or Jaccard measure),

$$JAC_{ij} = JAC_{ji} = \frac{C_{ij} + C_{ji}}{\sum_{k \neq j} C_{ik} + \sum_{k \neq i} C_{jk}}, \quad (3.1)$$

a normalized version of the symmetrized inter-citation frequencies that attempts to adjust for the overall citation frequency of individual journals, was judged to be preferable.

³ Note that, as the inter-citation frequency matrix is asymmetric, we could also choose to construct a directed graph.

In addition, these authors found it useful to apply some thinning of edges prior to mapping, and therefore chose to trim edges to include, for each journal, only the 15 most similar other journals. That is, the degree of vertices in the graph G was limited to be at most 15.

3.5.1.3 Visualization of the ‘Backbone’ of Science

The maps presented here were produced through a combined use of methods of automatic display, on the one hand, and human interpretation and annotation, on the other. VxOrd [109], an in-house visualization package produced by Sandia Labs, was used for the automatic display. The underlying drawing algorithm is based on an enhanced version of the spring-embedder methodology discussed in Section 3.4.2, involving the use of sophisticated optimization methods to search the space of possible graph drawings and a grid that helps reduce computation time from the typical $O(N_v^2)$ down to $O(N_v)$. In addition, it employs edge-cutting criteria, designed towards producing drawings that balance the detail with which both local and global structure are shown.

The human interpretation and annotation entered the mapping process at the next stage. The immediate output obtained from VxOrd is characterized by numerous small clusters of highly inter-connected vertices, with a sparse collection of ties between clusters. Given the nature of Science and scientific work, such structure is not unexpected. Boyack, Klavans, and Börner therefore took the VxOrd output and, through examination of the journals within clusters and their corresponding ISI categories, assigned labels to the clusters. The result of the overall two-stage process is shown in Figure 3.5, where the edges have been removed to improve readability.

Lastly, to produce our final map of the ‘backbone’ of Science, at the level of scientific disciplines, the clusters in Figure 3.5 were each replaced by a single vertex, the size of which is proportional to the number of journals in the cluster. Then, directed edges were placed from one vertex to another if more than 7.5% of the citations by journals corresponding to the first were to journals corresponding to the second, with darker edges reflecting larger percentages. Furthermore, the vertices were colored, on a scale ranging from dark to light, in a manner proportional to the relative frequency of self-citations among journals within a discipline, so that darker vertices may be viewed as more independent and lighter vertices as less independent. The results are shown in Figure 3.6.

Various features are apparent from this map, and we note just a few here. As is not uncommon with output from many automatic drawing methods, more highly linked vertices are located closer to center, while vertices with fewer links lie along the perimeter. Among those near the center, biochemistry and medicine are the two most highly connected and also quite large. The two disciplines differ, however, in their dependency on other disciplines, in that biochemistry is comparatively independent, and medicine, similarly dependent. Among the more mathematically-oriented sciences we note that statistics is both fairly close to the center of the map and comparatively independent, matching well with its reputation as an originating

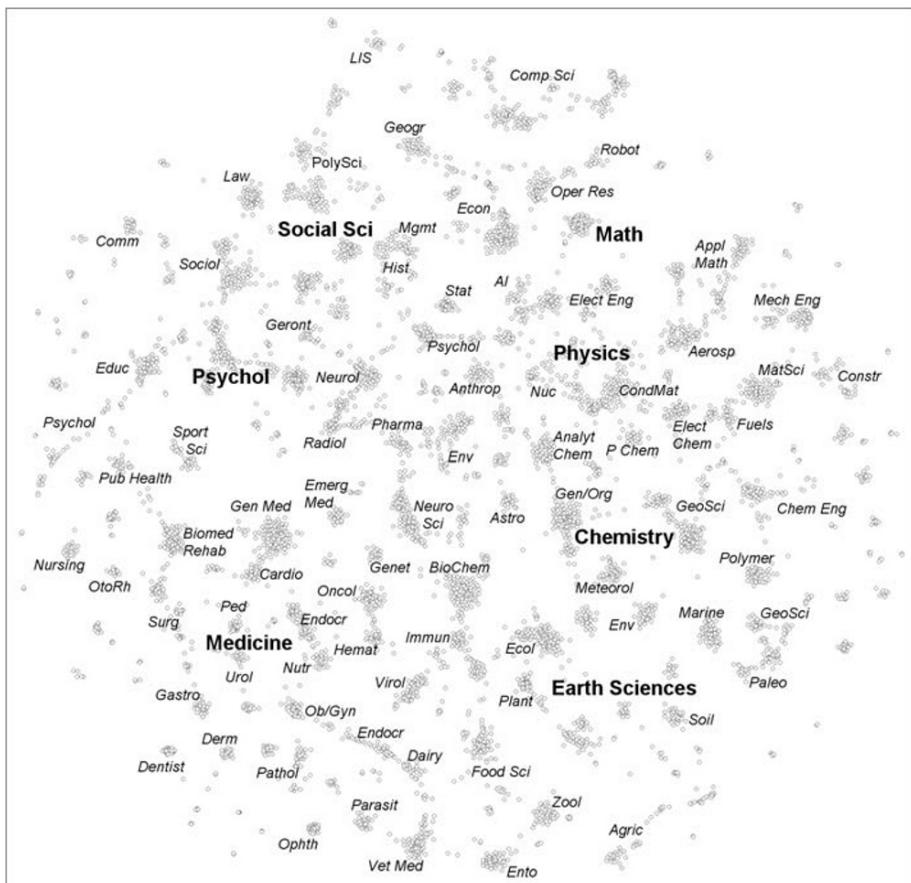


Fig. 3.5 Preliminary map of Science, resulting from a combination of automated drawing software and human interpretation and annotation. All edges have been removed, to improve readability. Figure courtesy of Kevin Boyack.

source of important quantitative concepts and tools that are then used broadly across other fields.

3.5.2 Mapping the Internet

While the Internet arguably pervades nearly every aspect of modern society, in ways both obvious and not so obvious, it may nevertheless come as a surprise to many people that a single, comprehensive map of this purely human construction is largely lacking. The lack of such a map is due to a variety of factors, including its dynamic and self-organizing nature, proprietary and security concerns among its



Fig. 3.6 Map of the ‘backbone’ of Science. Figure courtesy of Kevin Boyack.

various service providers (which own and maintain the separate, interacting pieces of the Internet), and simply its sheer size. Because of the dependency of so many interests on the Internet, the construction and maintenance of accurate maps is an important and ongoing activity.

Realistically, there are numerous inter-related aspects of the system colloquially referred to as ‘The Internet,’ including the physical infrastructure, the logical paths by which information flows over that infrastructure, the content underlying that information, the usage patterns of those creating, disseminating, and consuming that content, and the traffic created by such usage. Accordingly, there are many maps that we might think to produce. In this section, we focus on the task of mapping the logical structure of the Internet, drawing on the treatment in Crovella and Krishnamurthy [105] for background on measurement and network graph construction, and on Alvarez-Hamelin, Dall’Asta, Barrat, and Vespignani [8] for visualization.

3.5.2.1 Measurement

Recall that the physical Internet is a network of digital devices communicating over wired⁴ connections via a set of communication protocols. Information is transferred over the physical Internet in the form of *packets*, which may be thought of as ‘pieces’ of a larger information object (e.g., an email, photo, web-page content, etc.), equipped with information identifying the source and the destination. Devices can be categorized as either *end systems*, such as desktop and laptop computers, and *routers*, which are responsible for receiving packets from one location and sending them onward towards their destination (much like a system of post offices). The connections between devices are simply referred to as *links*.

The logical Internet, on the other hand, which will be our focus, refers to the set of paths over which packets are routed through the physical Internet. Common units used in mapping the logical Internet are either the routers themselves or aggregations of routers. Common units of aggregation include administrative units (e.g., grouping according to so-called autonomous systems or ASs) and geographical concentrations (e.g., as in the Abilene map in Figure 1.1). Logical links may then be representative of either the connections between routers or the effective connections between aggregations of routers. Note, however, that as the logical Internet refers to the *used* portion of the Internet, physical nodes and links that are unused at a given point in time in the routing of traffic are essentially invisible from the logical perspective. In this section, we will use the routers themselves as units.

The distinction between physical and logical is important to us because, while there are no databases from which to extract comprehensive information on either, unlike the case when mapping Science in Section 3.5.1, it is the logical Internet that is most amenable to large-scale measurement, albeit by indirect means. Common techniques utilize information pertaining to or generated by the very communication protocols underlying the flow of Internet traffic. For example, the data we use here was generated by use of a certain probing mechanism, called `traceroute`, in a manner not unlike the sending of acoustic signals into the Earth in geological surveys of oil reserves.

Figure 3.7 shows an illustration of how `traceroute`-based measurement works. Part of the usual information with which each Internet packet is equipped, prior to being sent from its source to a destination, is information on its ‘time to live’ (TTL), an integer value that is decreased by one at each router through which the packet passes and is intended to ensure that misrouted packets do not continue to circulate through the Internet for eternity. If the TTL of a packet reaches zero at a router on its way to a destination, that router ‘drops’ the packet (i.e., it is sent no further) and, importantly, can generate a message to notify the original sender that includes the router’s unique identifying address (i.e., its Internet protocol, or IP, address). The `traceroute` tool exploits this protocol characteristic by sending a sequence of packets towards a destination of interest with successively greater TTLs, and records the corresponding IP address returned by each intermediate router re-

⁴ We will ignore the issue of wireless connections in this section since, due to their nature, such connections essentially are found only at the periphery of the Internet.

sponding that it has encountered a packet with a TTL of zero. In Figure 3.7, for example, a sequence of three traceroute probes (the communication patterns for which are shown in blue, green, and red) are used in discovering the two routers along the path from the given source to the intended destination.

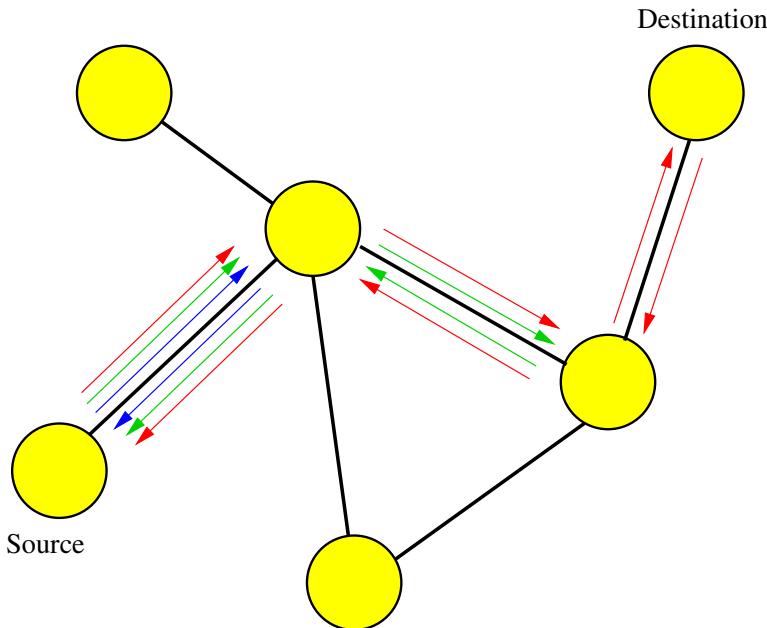


Fig. 3.7 Illustration of the use of traceroute to sample the logical topology of the Internet.

By probing in this manner from a number of sources to a number of destinations, it is possible to obtain raw information by which some sense of the logical topology of the Internet may be derived. CAIDA⁵ (the Cooperative Association for Internet Data Analysis) is an independent, non-profit group, combining resources from government, industry, and academia, that conducts a number of large-scale Internet measurement projects, including the *Skitter* topology project. For Skitter, CAIDA has established roughly 20 measurement centers around the world, from which it sends traceroute-like probes to approximately 800,000 destinations, chosen in an attempt to provide as uniform a coverage of ‘destination space’ as possible. The maps shown later in Figure 3.8 are based on CAIDA data collected between April 21 and May 8, 2003.

Note that traceroute-like studies such as Skitter result in sampled network data, and so it is important to consider the effects of such sampling. An obvious sampling effect is the selection of sources and destinations, which will clearly dictate the paths for which routing information is gathered. We will consider this issue further

⁵ <http://www.caida.org/>

in Chapter 5. In addition, there are more subtle effects, such as the role played by certain security measures in the Internet, like the use of firewalls (from behind which topology information cannot be obtained) and the increasing tendency for owners of routers to disable the reply mechanism upon which `traceroute` is based (so as to reduce the overall communication burden on their router). Discussion of effects such as these are beyond the scope of this book.

3.5.2.2 Network Graph Construction

In principle, from each sequence of `traceroute` probes we can infer a logical path in the Internet from a source to a destination, and given a collection of such sequences, from the source to multiple destinations, an entire tree (or, more generally, a DAG) can be inferred. Finally, a collection of such trees, obtained from multiple sources, can be merged to create a network graph G representing the measured topology.

In reality, however, there are a number of practical difficulties that must be dealt with. For example, there is some asymmetry in the manner in which traffic is routed in the Internet, so that, unlike the simple illustration in Figure 3.7, the path from a node A to a node B may not be the same as that from B to A . Hence, such studies realistically produce only information on directed paths. Second, there is the issue of time sensitivity, in the sense that the paths routed on the Internet are dictated by a set of protocols reacting to a combination of human and machine feedback, and hence are dynamic. In merging the results of probes sent over a period of days or weeks, for example, false links can be generated, induced purely by routing changes. Third, routers do not typically have a single interface with the network by which traffic is received and sent, but rather multiple such interfaces, and hence multiple IP addresses. In other words, each router can be encountered by various probes under a number of different ‘aliases.’ It is therefore necessary to resolve these possibly multiple aliases so as to avoid mapping multiple nodes where only one router is present. The reader is referred to Crovella and Krishnamurthy [105] for additional details on such problems and methods for their solution.

3.5.2.3 Visualization of the Router-Level Internet

After the appropriate post-processing, the CAIDA data yield a network graph G representing 192,244 discovered routers and 609,066 discovered edges. As discussed in earlier sections of this chapter, it is a challenge to create informative visualizations of such large graphs. CAIDA itself has produced some visualizations⁶ based on aggregation of this sort of router-level data to the scale of autonomous systems. Here we will present another form of visualization, one designed to highlight the

⁶ See <http://www.caida.org/tools/measurement/skitter/visualizations.xml>.

known hierarchical structure of the Internet (i.e., as illustrated in the context of the Abilene network in Figure 1.1).

Specifically, we will use the concept of a *k*-core decomposition, which has been at the heart of a number of proposals for the representation and visualization of large network graphs (e.g., Batagelj, Mrvar, and Zaversnik [25], Baur, Brandes, Gaertler, and Wagner [26], and Alvarez-Hamelin, Dall'Asta, Barrat, and Vespignani [8]). A *k*-core of a graph G is a subgraph H of G for which all vertex degrees are at least k , and such that no other subgraph obeying the same condition contains it (i.e., it is maximal in this property). *k*-cores are one natural way⁷ to look at group structure across a graph G . In a *k*-core decomposition, the vertex set V is partitioned into ‘shells’ of vertices, say C_c , such that all vertices in C_c belong to the *c*-core but not the $(c + 1)$ -core. Vertices in C_c are said to have a ‘shell index’ of c . Appropriate visualization of this collection of shells allows us to view the graph G at successive levels of ‘core-ness’.

Figure 3.8 shows such a visualization, as well as two close-up views, generated using the *k*-core visualization tool *LaNet-vi* described in Alvarez-Hamelin, Dall'Asta, Barrat, and Vespignani [8]. *LaNet-vi* is a freely available software tool⁸ that uses a combination of different layout concepts, including some from radial and spring-embedder methods introduced earlier, and runs in $O(N_v + N_e)$ time. Vertices, represented as dots, are placed within their respective shells C_c , which in turn are organized in a series of concentric rings. Shells with higher index (i.e., more at the ‘core’ of the network) are located on rings closer to the center, and those with lower index, farther from the center. The rings are distinguished by their color, ranging across the natural spectrum, from red (high shell index) to purple (low shell index). Within each ring, connected clusters of vertices are kept together and ordered around the circumference. Ring thickness is a function of how connected vertices within a given shell are to those in shells higher or lower than them. Individual vertices with comparatively more neighbors of higher shell index are located closer to the internal part of their ring, and vertices with more neighbors of lower shell index, closer to the external part of their ring. Hence, the more variable this higher/lower connectedness among vertices of a given shell index, the thicker the corresponding ring (relative to a user-defined maximum thickness). The size of individual vertices scales proportional to their degree and is displayed on a logarithmic scale.

Various structural characteristics of the router-level Internet are suggested by the maps in Figure 3.8. For example, the hierarchical structure is quite evident, as links appear to run primarily between vertices of different shell indices, with noticeably fewer links among vertices of the same shell index. Furthermore, differences in shell thickness are apparent, with a number of the shells of lower shell index evidently much thicker than those of higher shell index. This feature indicates a larger range of variability in the shell index of neighbors of vertices in lower-index shells compared to higher-index shells. Note too that, although a large fraction of the high-degree vertices are located near the center, we can see (particularly in the cross-section)

⁷ We will encounter *k*-cores again, as well as other similar measures, in Chapter 4.

⁸ Available at <http://xavier.informatics.indiana.edu/lanet-vi>.

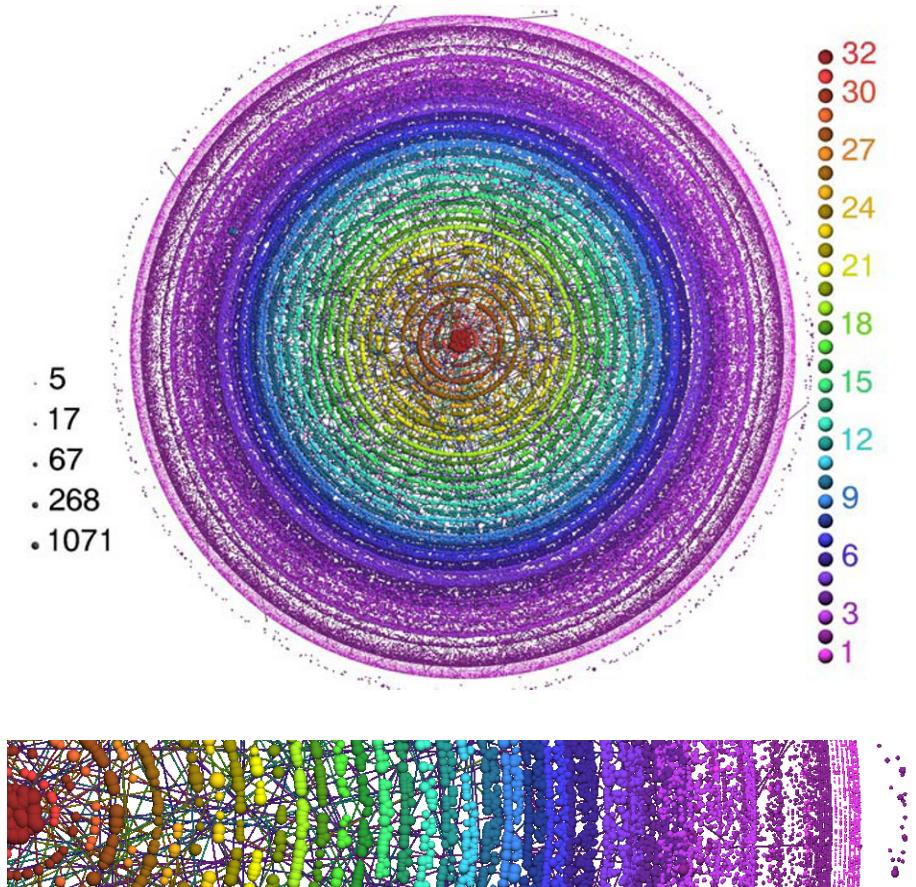


Fig. 3.8 Visualization of a k -core decomposition of the CAIDA router-level Internet data. Figure courtesy of Ignacio Alvarez-Hamelin.

that there are nevertheless some high-degree vertices quite far out on the external rings as well, indicating that highly connected routers can in fact be quite peripheral.

3.6 Mapping Dynamic Networks

The material in this chapter has focused almost exclusively so far on the case where the goal is to produce a single map of a static network graph G . But clearly many of the systems studied from a network-based perspective are intrinsically dynamic in nature. Depending on the goals of the researcher, it may therefore be of interest to produce, instead of a single static map, a dynamic network map(s) from a collection

$\{G^{(t)}\}$ of network graphs $G^{(t)}$ indexed over times t in some range \mathbb{T} . We close this chapter with a brief look at this topic.

Obviously many of the basic principles of measurement, graph construction, and display discussed for static network mapping continue to hold true in the dynamic case. And in fact the most straightforward way to create a dynamic mapping of a network is as a sequence of separate static maps. More ambitiously, if the data are rich enough temporally, instead of displaying all of the individual static maps, one might animate them to produce a ‘movie,’ so as to watch the network graphs change over time.

From the point of view of the graph display problem, the basic difference between the static and dynamic cases is the need to respect, in the case of the latter, what is referred to as the user’s *mental map*. This term is used to describe the result of the process by which, upon studying a given static network map, a user becomes familiar with it, interprets it, and navigates about it. Dynamic changes in a sequence of network graphs, say from $G^{(t)}$ to $G^{(t+1)}$, involve the addition and deletion of vertices and links. If these changes are small, from one time period to the next, intuitively we would like to see only correspondingly small changes in each of their respective visualizations. That is, we would expect a certain amount of ‘stability’ across visualizations. However, unless this requirement is incorporated as an additional parameter, automatic drawing methods can produce very different looking displays in the face of even small changes. The result is that the user is then forced to form a new mental map for each graph $G^{(t)}$ and to struggle with the effort of reconciling the sequence of maps.

Although the area of dynamic graph drawing is still arguably in its infancy, there are already a number of approaches that have been developed for automated display of dynamic network graphs with constraints to reinforce a consistent user mental map. Most stringently, we may refuse to allow vertices and edges to move with the addition of new elements to the graph. But this strategy can quickly lead to cluttered maps if the graph changes sufficiently over time. More subtly, we may attempt to allow only ‘local’ elements in the vicinity of a newly added graph element to change. Use of distance metrics between graphs, in an effort to choose layouts for which consecutive graphs are not ‘far’ from each other, are also popular. More generally, Brandes and Wagner [58, 59] have suggested a Bayesian framework (based on the use of Gibbs distributions for random fields) within which many of these ideas, and ideas from static network graph display, can be subsumed and dealt with together in a consistent manner. Overall, however, there does not yet appear to be a consensus on a preferred method(s). See Chapter 9 of the edited volume of Kaufmann and Wagner [221] for a recent overview.

Of course, all of the above presupposes that the addition of a temporal index does not adversely affect basic issues of data acquisition and management, nor the construction of effective graph representations. But in some dynamic network contexts this is most certainly not the case, and attention must be given to all of these issues right from the start. Volinsky and colleagues (e.g, Hill, Agarwal, Bell, and Volinsky [198]) have suggested an approximation scheme for dynamic network graphs based on the concept of exponential smoothing, common in statistical time series

analysis, wherein the network graph representation at time t (i.e., $G^{(t)}$) is written as a weighted sum of the network graph at the previous time point, $G^{(t-1)}$, and the set of vertices and edges new at time t , say $g^{(t)}$. That is, they propose a recurrence of the form $G^{(t)} = \theta G^{(t-1)} \oplus (1 - \theta)g^{(t)}$, where the operation \oplus represents union of vertex and edge sets and weighted arithmetic sum of edge weights. Combined with an edge-thresholding mechanism and a strategy for estimation of the relevant parameters, the methodology is illustrated in the context of massive telephone call graphs.

3.7 Additional Related Topics and Reading

Network mapping is about effective visual communication of relational information. Here we have presented network mapping as a three-stage process, consisting of data collection, network graph representation, and visualization. Brandes and colleagues (e.g., Brandes et al. [60], Brandes [53]) have offered a similar characterization of network visualization. Their characterization is broken down according to the three elements of system, design, and algorithm, which arguably overlap primarily our second and third stages, thereby providing in part a more refined breakdown of those two.

Most methodical expositions of issues associated with the collection of relational data appear to be found in the context of social networks – for example, see the books by Scott [344] and Wasserman and Faust [393], and the review by Marsden [269]. An extensive overview of principles and techniques for network graph visualization may be found in the edited volume of Kaufmann and Wagner [221]. In addition, Chapter 2 of the Ph.D. thesis of Brandes [53] is quite useful in this regard. For detailed development of the corresponding algorithms, see the text of di Battista, Eades, Tamassia, and Tollis [117]. It should be noted that, as Brandes [53] has pointed out, compared to the more established body of work on the visualization of more standard statistical data (e.g., as summarized in the books by Tukey [383], Tufte [382], and Cleveland [93, 94]), the development of formal methods for the visualization of relational data is still in its relative infancy.

Lastly, as was mentioned in Section 3.1, the validation of network maps is an interesting and important topic that merits greater attention. In the automatic construction of geographical maps, such as from measurements obtained by remote sensing instruments, *validation* refers to the process by which one verifies the extent to which the map matches the reality. But in such contexts the comparison is to an appropriate ‘ground truth,’ such as can be obtained by having an ‘expert’ literally visit and assess the spatial content of areas of the Earth. In the network context, it is less clear what constitutes ‘ground truth’ in many cases, and in other cases it is not even clear that ‘ground truth’ is in fact the most useful representation of the information that one wishes to convey. Brandes [53, Ch. 1] provides a nice illustration of this latter issue, through the comparison of the traditional schematic map of a city subway system versus a geographically accurate map of the same. The geographi-

cal map, while arguably a more faithful physical replicate of the actual subway, is nevertheless in fact not as useful to the average rider as the schematic map. On the other hand, when some sense of the reality to be captured is available, such as in the form of an alternative set of measurements of the same system of interest, it is tempting to want to ask to what extent the network map formed from the original measurements matches the information captured in the alternative measurements. Some initial efforts in this direction in the context of mapping ‘Science’ may be found in Klavans and Boyack [228].

Exercises

3.1. For a network graph of interest to you, reflect on the nature of the underlying measurements and the corresponding construction of the graph. Trace out in detail (to the extent you can, if you yourself did not create the network graph) the manner in which vertices and edges were defined from the measurements. To what extent are your data enumerated, sampled, or partial? Is it of interest to be able to ‘validate’ your network graph and, if so, what information would need to be available to perform such a validation?

3.2. In Figure 3.9 is shown a graph displayed with a circular layout. This graph is in fact a tree. If we assign the vertex 1 to be the root, and we assign an order to the other vertices with respect to (i) their distance from the root and (ii) the order of their labels among vertices of equal distance, then we have an *ordered rooted tree*.

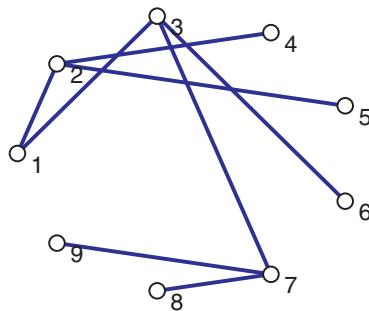


Fig. 3.9 Graph corresponding to Exercise 3.2.

A *preorder traversal* of this tree is a walk that visits the root first, and then visits the vertices of the first sub-tree in preorder, and then the second, in a recursive fashion, yielding the preorder listing $\{1, 2, 4, 5, 3, 6, 7, 8, 9\}$. A *postorder traversal* of this tree is a walk that traverses in postorder the vertices in the first sub-tree of

the root and then the second, in a recursive fashion, yielding the postorder listing $\{4, 5, 2, 6, 8, 9, 7, 3, 1\}$.

Consider the following tree drawing algorithm. First, we assign a Cartesian coordinate (x_v, y_v) to each vertex v , where x_v is the rank of v in the preorder traversal list, and y_v , the rank in the postorder traversal list. Second, points are connected by a straight line if their corresponding vertices are adjacent in the original underlying graph. Implement this algorithm for the tree shown in Figure 3.9. Is your drawing planar? What is its orientation? What is its effective area? How might you modify the assignment of coordinates to vertices to reduce this area?

3.3. Recall that basic spring-embedder methods of graph drawing assign a measure of ‘force’ to each vertex $v \in V$ of a graph G and iteratively adjust the position $p_v = (x_v, y_v)^T$ of each vertex until the forces on each have converged to an equilibrium value. A standard setup defines the force $F = (F_x, F_y)^T$ on v in the x direction to be

$$F_x(v) = \sum_{\{u,v\} \in E} k_{uv}^{(1)} (||p_u - p_v|| - \text{len}_{uv}) \frac{x_v - x_u}{||p_u - p_v||} + \sum_{(u,v) \in V^2} \frac{k_{uv}^{(2)} (x_v - x_u)}{||p_u - p_v||^3}, \quad (3.2)$$

and similarly in the y direction, with y_u, y_v substituted for x_u, x_v . Here $||\cdot||$ indicates Euclidean distance, while len_{uv} , $k_{uv}^{(1)}$, and $k_{uv}^{(2)}$ are user-defined parameters, where len_{uv} is the desired length of the edge to be drawn between u and v , $k_{uv}^{(1)}$ is a ‘spring stiffness’ (i.e., larger values force $||p_u - p_v||$ closer to len_{uv}), and $k_{uv}^{(2)}$ is a strength of ‘electric repulsion’ counter-balancing the spring stiffness.

A simple algorithm utilizing these equations is as follows. First, initialize the algorithm by randomly assigning each vertex to a location. Then, for each iteration of the algorithm and each vertex $v \in V$, compute $F(v)$, using equation (3.2) above and your own specified constants len_{uv} , $k_{uv}^{(1)}$, and $k_{uv}^{(2)}$, and move v in the direction of $F(v)$ by an amount $\varepsilon ||F(v)||$, for some small $\varepsilon > 0$.

Implement the above algorithm for a small network graph (i.e., a toy graph of your own construction will do). For an appropriate choice of parameters, visualize a succession of iterations of the algorithm (i.e., as in a movie), so as to observe the manner in which the method produces its final graph drawing from the random initialization. Explore the behavior of the method as a function of your parameter settings.

3.4. Using a network graph of interest to you, familiarize yourself with a graph drawing software. While doing so, pay particular attention to issues of aesthetics and the information being conveyed as a function of different layouts and variations on any parameters associated with these layouts. Is it useful or necessary to modify the number of vertices or the density of edges in your graph? Are there interesting visualizations of subgraphs in your network? After a thorough exploration, produce a concise report summarizing your findings.

Chapter 4

Descriptive Analysis of Network Graph Characteristics

Given a network graph representation of a system – possibly constructed from an appropriate set of measurements, in the manner of the previous chapter – it is common to explore the characteristics and structural properties of the network. Tasks range from the calculation of simple metrics summarizing topological structure, both local and global, to the unsupervised extraction of complex relational patterns. In this chapter we present a core set of tools and techniques for such purposes.

4.1 Introduction

In the study of a given complex system, questions of interest can often be re-phrased in a useful manner as questions regarding some aspect of the structure or characteristics of a corresponding network graph. For example, various types of basic social dynamics can be represented by triplets of vertices with a particular pattern of ties among them (i.e., triads); questions involving the movement of information or commodities usually can be posed in terms of paths on the network graph and flows along those paths; certain notions of the ‘importance’ of individual system elements may be captured by measures of how ‘central’ the corresponding vertex is in the network; and the search for ‘communities’ and analogous types of unspecified ‘groups’ within a system frequently may be addressed as a graph partitioning problem.

The structural analysis of network graphs has traditionally been treated primarily as a descriptive task, as opposed to an inferential task, and the tools commonly used for such purposes derive largely from areas outside of ‘mainstream’ statistics. For example, an overwhelming proportion of these tools are naturally graph-theoretic in nature, and thus have their origins in mathematics and computer science.¹ Similarly, the field of social network analysis has been another key source, contributing tools usually aimed – at least originally – at capturing basic aspects of social structure and

¹ The concepts and terminology reviewed in Chapter 2.1 will be particularly relevant here in the present chapter.

dynamics. More recently, the field of physics has also been an important contributor, with the proposed tools often motivated by analogues in statistical mechanics.

For convenience of exposition, we have organized the various concepts and tools for structural analysis to be covered in this chapter as primarily falling into two broad categories: (i) characterization of individual vertices and edges, and (ii) characterization of network cohesion (i.e., involving more than just individual vertices and edges). Note that we have expressly *not* used the related terms ‘local’ and ‘global’ in our categorization, as some measures of vertex-level characteristics in fact summarize information from across the graph, and conversely there can be cohesion among vertices only in some small portion of the graph.

By way of organization, this chapter is laid out as follows. Section 4.2 covers vertex and edge characteristics, while Section 4.3 covers the characterization of network cohesion. Then, in Section 4.4, we illustrate the combined usage of the various tools and techniques described, through a case study in the context of networks of cortical-level probes in the brain and epileptic seizures. Lastly, the analysis of structure in dynamic networks is touched upon in Section 4.5. Some additional topics, as well as references for reading, are described in Section 4.6.

4.2 Vertex and Edge Characteristics

As the fundamental elements of network graphs are their vertices and edges, there are a number of network characterizations centered upon these. We discuss several such characterizations here in this section. Our presentation is broken down according to (i) those characterizations based upon vertex degrees, and (ii) those seeking to capture some more general notion of the ‘importance’ of a vertex – typically referred to as vertex centrality measures. We also discuss how measures of vertex centrality often extend in a straightforward manner to describe the importance of edges.

4.2.1 Degree

Recall that the degree d_v of a vertex v , in a network graph $G = (V, E)$, counts the number of edges in E incident upon v . That is, it provides a basic quantification of the extent to which v is connected to other vertices within the graph. When the vertex degrees are considered in aggregate, through the degree sequence $\{d_1, \dots, d_{N_v}\}$, various useful measures of the nature of the overall connectivity in the graph can be defined. We discuss a few such measures here. Note that in the case of directed graphs, such measures are typically applied separately to the sequences $\{d_v^{in}\}_{v \in V}$ and $\{d_v^{out}\}_{v \in V}$ of in- and out-degrees.

4.2.1.1 Degree Distributions

Given a network graph G , define f_d to be the fraction of vertices $v \in V$ with degree $d_v = d$. The collection $\{f_d\}_{d \geq 0}$ is called the *degree distribution* of G , and is simply the histogram formed from the degree sequence, with bins of size one, centered on the non-negative integers. For example, in the protein-protein interaction network of Figure 3.2, corresponding to *Drosophila*'s circadian clock mechanism, the degree sequence $\{1, 3, 4, 4, 3, 3\}$, associated with the vertices *Sgg* through *Cyc*, moving clockwise, yields the degree distribution $\{f_1 = 1/6, f_3 = 1/2, f_4 = 1/3\}$.

For directed graphs, degree distributions may be defined analogously for in- and out-degrees. For example, in the gene regulatory network also shown in Figure 3.2, the in-degree distribution is $\{f_2^{in} = 2/3, f_3^{in} = 1/3\}$, while the out-degree distribution is $\{f_0^{out} = 1/3, f_2^{out} = 1/3, f_4^{out} = 1/3\}$.

The degree distribution provides a natural summary of the connectivity in the graph. The examples above are, of course, fairly trivial. In practice, degree distributions are arguably more interesting as descriptors for large graphs.

Example 4.1 (Degree Structure for Internet and Protein Interaction Networks). Figure 4.1 shows plots of the degree distributions for two large network graphs. The first graph corresponds to the router-level Internet representation derived from the CAIDA data described in Section 3.5.2. Recall that this graph derived from the union of measured subgraphs, resulting from probes sent from roughly 20 measurement centers to approximately 800,000 Internet destinations, resulting in 192,244 discovered vertices and 609,066 discovered edges.

The second graph corresponds to a network of measured protein interactions in the organism *Saccharomyces cerevisiae* – better known as brewers' or bakers' yeast. The data were culled from the January 2007 version of the BioGRID database,² a general repository for biological interaction datasets, and consist of an aggregation of published experimental results from various labs across the world on the affinity for protein pairs to interact with each other. This network graph, made up of 5,151 vertices (proteins) and 31,201 edges (interactions), is similar to the protein-interaction network for *Drosophila* shown in Figure 3.2. However, it differs not only in magnitude but also, importantly, in the nature of the declared interactions. Specifically, in the yeast database, the interactions were determined largely by the strength of evidence provided by various high-throughput experimental techniques, rather than cumulative biological judgment. In addition, the proteins and their interactions are not specific to any one biological function, whereas in Figure 3.2, all proteins and interactions are present precisely due to their role in the *Drosophila* circadian clock mechanism.

One aspect of these two networks that is immediately apparent is the broad range over which the degrees in the graphs vary. The vertices in the protein network range in degree from 1 to 301, and those in the Internet graph, from 1 to 1,071. In addition, it turns out that both degree distributions are highly skewed to the right. The average vertex degree \bar{d} is roughly 6.3 and 12 for the Internet and protein interaction data,

² See <http://www.thebiogrid.org/> and Stark et al. [367].

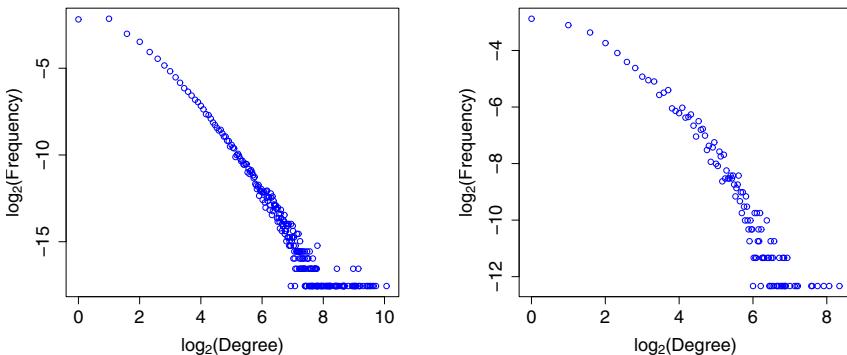


Fig. 4.1 Degree distributions. Left: the router-level Internet network graph described in Section 3.5.2. Right: the network of measured interactions among proteins in *S. cerevisiae* (yeast), as of January 2007. In each plot, both x - and y -axes are in base-2 logarithmic scale.

respectively, while the corresponding medians are only 3 and 6. The averages are actually much closer to the third quartiles, found at 6 and 15, respectively.

In fact, each of these distributions has been plotted on a logarithmic scale on both axes (i.e., on a log-log scale), and as a result, we see that the skewness in these distributions is of a particular nature. Specifically, while the majority of vertices are of very low degree, a nevertheless nontrivial number of vertices are of much higher degree – two to three orders of magnitude higher, in some cases. The roughly linear decay of the points in each plot, over at least some portion of the range of each distribution, suggests the presence of a power-law component to these distributions. That is, it suggests that something like

$$f_d \propto d^{-\alpha} \quad (4.1)$$

holds approximately true in part. \square

In just the past 10 or 15 years, it has been found that approximate power-law degree distributions appear to be ubiquitous in networks across many areas of the sciences. This discovery was originally quite unexpected, as such structure is in direct contrast to that of networks typically studied throughout much of the 20th century, such as networks of circuits or traditional random graphs.³ In these latter cases, vertex degree is of a fairly similar order of magnitude across the graph – homogeneous, rather than heterogeneous. The corresponding degree distributions are hence quite concentrated, rather than diffuse, and typically decay exponentially fast in d , rather than like a power-law. Note that the emergence of power-law-like behavior in network graphs is not simply a function of increasing network size. For example, the

³ We will encounter random graphs of this type in Chapter 6.

power-grid of the western United States, forming a network of roughly 5,000 generators, transformers and substations, connected by just over 13,000 transmission lines, has a degree distribution with exponential decay (e.g., see Newman [296, Fig. 6]).

In any event, given an observed degree distribution, whether homogeneous or heterogeneous, it is tempting – and indeed natural – to want to summarize that distribution. Reporting basic summary statistics, in the form of moments and quantiles, as in Example 4.1, is common. It is also common in some fields to report the fit of some simple, parametric family of distributions to the observed degree distribution. This task, however, can be far from trivial when heterogeneous degree distributions are involved.

Example 4.2 (Fitting Power-Laws to the Internet and Protein-Interaction Networks). Consider the degree distributions for the Internet and protein-interaction networks shown in Figure 4.1, and recall that their shape suggested the possibility of a power-law distribution, like that in (4.1), over at least some portion of their support. Ignoring, for the moment, issues regarding the support of the power-law component, the exponent α is the key parameter that needs to be extracted from the data.

There are a number of ways to proceed in this task, although some are more advisable than others. Most simply, we could note that (4.1) implies the approximate relation

$$\log f_d \sim C - \alpha \log d , \quad (4.2)$$

where C is an arbitrary constant. Hence, in principle, a sensible estimate of α might be obtained by fitting a line to the plot of $\log f_d$ versus $\log d$, using a method like least squares regression. In practice, however, this method is not advisable, due to the disproportionate level of ‘noise’ in the data at the high degrees, where the levels of counts are the lowest.

A common solution to the noise problem is to use cumulative frequencies, rather than the raw frequencies themselves, a process which effectively smooths the noise. For example, since under model (4.1) the tail probabilities take the form

$$\bar{F}(d) = 1 - F(d) \sim d^{-(\alpha-1)} , \quad (4.3)$$

we can then again consider using a regression-based approach to estimate α . Alternatively, in the statistical physics literature there is the related practice of using relative frequencies calculated on intervals of logarithmically increasing size (i.e., so-called logarithmic binning), followed by a linear regression fit. Details may be found in Pastor-Satorras and Vespignani [309, App. 3], for example.

All of these regression-based approaches, although common, have been criticized for their somewhat *ad hoc* nature and the biases that can be introduced. See the numerical study by Goldstein, Morris, and Yen [179], for example. A more rigorous alternative, and one widely used in many fields that routinely encounter power-laws, are estimators of the form

$$\hat{\alpha}_k = 1 + \hat{\gamma}_k^{-1} \quad , \quad \text{with} \quad \hat{\gamma}_k = \frac{1}{k} \sum_{i=0}^{k-1} \log \frac{d_{(N_v-i)}}{d_{(N_v-k)}} \quad , \quad (4.4)$$

where $d_{(1)} \leq \dots \leq d_{(N_v)}$ are the sorted vertex degrees and k is a value chosen by the user.⁴ Typically, instead of a single choice of k , the estimate (4.4) is computed over a range of choices for k , and a plot of $\hat{\alpha}_k$ versus k – called a *Hill plot* – is examined with an eye towards finding areas where the plot ‘settles down’ to some stable values of $\hat{\alpha}$, away from extremes where k is too small or too large. Drees, de Haan, and Resnick [126] provide a discussion of these and similar plots.

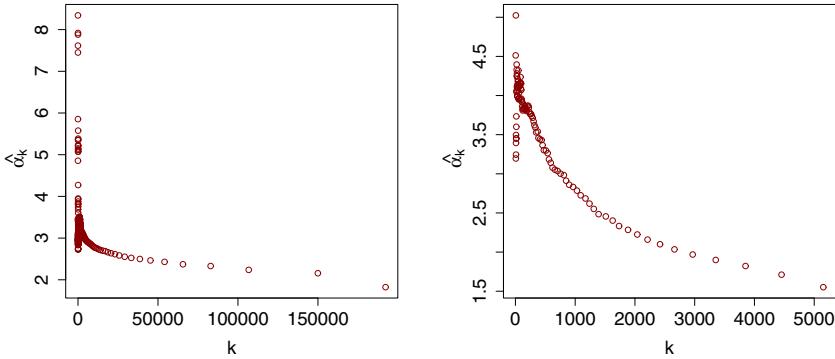


Fig. 4.2 Hill plots of maximum likelihood estimates $\hat{\alpha}_k$, as a function of k (i.e., the number of order statistics used), for the Internet (left) and protein interaction (right) datasets.

Hill plots are shown for our two networks in Figure 4.2. The plot for the Internet network shows a broad stretch of estimates with fairly stable behavior, having just a slight decay from 3 to 2 over many orders of magnitude in k , where the limiting behavior in this range (as $k \rightarrow 0$) arguably tends towards a value a bit below 3.0. This sort of appearance, and interpretation, is fairly typical of Hill plots where a power-law model is reasonably credible. In contrast, the plot for the protein interaction network looks rather different, with estimates decaying much more sharply than in the Internet network, and over a decidedly smaller range of values k . Looking back at Figure 4.1, we note that the degree distribution for the protein interaction network arguably has a more concave nature to it than that for the Internet network. This concavity produces the corresponding convexity in the Hill plot, which

⁴ The estimator $\hat{\gamma}$ is called the *Hill estimator*. When the model (4.1) holds exactly, it is the maximum likelihood estimate of $\gamma = (\alpha - 1)^{-1}$; more generally, it is an approximate maximum likelihood estimate if $1 - F(d) \sim d^{-(\alpha-1)} L(d)$, for some slowly varying function $L(\cdot)$, given appropriate choice of k . Consistency of this estimator has been proved under a variety of conditions. See Drees, de Haan, and Resnick [126].

strongly suggests that a simple power-law-like model is inappropriate in the case of the protein network. \square

Exponents summarizing power-law-like behavior in network degree distributions should be reported with caution, and reported exponents should be interpreted with the proverbial grain of salt.⁵ Note that to even realistically entertain discussion of exponents of this type is only really appropriate for large networks in which the degrees range over a good few orders of magnitude, the number of which is intimately tied to the value α itself. In many settings, it may not even be feasible that vertices possess a degree beyond some modest value d_{max} . For example, while we observed earlier that the largest degree in our Internet network graph was 1,071, at the time that the underlying dataset was collected most routers generally could establish no more than 64 links with other routers, or perhaps 128 in special cases. ‘Routers’ in our graph with larger vertex degree than this are actually composites of many such routing devices. Hence, the interpretation of the results above require some care. Similarly, in the context of social networks, there is often a ceiling on the number of relationships of a given type that an individual can maintain. Most social relationships of substantive interest require personal resources of some nature, and so too many relationships become costly.

In the face of such concerns, a seemingly obvious solution is to consider more complicated models, such as mixtures of power-laws or distributions with power-law behavior and exponential truncation (i.e., $f_d \propto d^{-\alpha} \exp(-d/d^*)$, for some $d^* \gg 0$). Unfortunately, fitting such models and, more broadly, discriminating between various hypothesized models of these types, is usually a delicate task at best. Of course, the challenge here is not restricted to the realm of networks alone, but rather is due to working with power-law-like distributions (i.e., distributions possessing long, heavy tails). See Hernández-Campos, Marron, Samorodnitsky, and Smith [197], for example, for discussion of some of the underlying issues and some proposed solutions, in the context of heavy-tailed Internet traffic data, as well as the references therein. Also see Mitzenmacher [280] for an extensive history of power-law-like distributions in general and discussion of some of the various subtleties and challenges in this area.

On a final note, we point out that, beyond the issue of appropriately summarizing power-law-like degree distributions, it is a topic of intense interest to understand the mechanisms by which such distributions emerge, both from theoretical and practical perspectives. We will return to this topic in Chapter 6. Additionally, there has also been much interest in understanding the impact of this structure on such things as communication, search, and the spread of epidemics in networks, topics we will pick up in Chapter 8.

⁵ We note too that, given such concerns about the estimates of α themselves, the tendency in the literature to also report standard errors for these estimates is somewhat gratuitous.

4.2.1.2 Degree Correlation

The degree distribution is useful as a composite summary of how degree varies across vertices in the graph, but it does not provide any information on precisely which vertices are connected to which others. To capture information of this sort, it is helpful to establish summaries that describe the patterns of association among vertices of given degrees. In fact, such summaries can be quite important, as two graphs may have identical degree sequences and yet otherwise differ noticeably in the way their vertices are paired. See Doyle et al. [125], for example, for a study of this phenomenon and its implications in the context of the topology of the Internet.

A natural starting point is to define a two-dimensional analogue of the degree distribution, capturing the relative frequency with which the two vertices at the end of an arbitrarily selected edge in the graph have a given pair of degrees. For undirected graphs, some care is needed in defining this distribution, in light of the lack of ordering among vertices that characterize an edge. One approach is to first sort the vertices in all edges $e \in E$ so as to have the form $e = (v_1, v_2)$, where $d(v_1) \leq d(v_2)$. Then, for each pair $d_1 < d_2$, we assign half of the relative frequency of the sorted edges with degrees (d_1, d_2) to f_{d_1, d_2} and the other half to f_{d_2, d_1} . For $d_1 = d_2$, we let f_{d_1, d_2} simply be the relative frequency of edges with both vertices of this degree. The resulting distribution, say $\{f_{d, d'}\}$, is symmetric and corresponds to selecting an edge at random from E and reporting it with its vertices in increasing degree or decreasing degree with equal chance.⁶

Example 4.3 (Joint Degree Distributions for the Internet and Protein Interaction Networks). Figure 4.3 shows image representations of the joint degree distributions for the router-level Internet data and the protein interaction data. In both cases we find the joint distribution to be concentrated primarily where pairs (d, d') are both low, as would be expected from our study of the marginal degree distributions. However, interestingly, we can see there is also a noticeable tendency for the vertices of largest degree to be connected to low-degree vertices, rather than other high-degree vertices. The effect appears to be somewhat more pronounced in the protein interaction data. Maslov and Sneppen [270] and Maslov, Sneppen, and Zaliznyak [271] originally made observations of this kind in earlier protein and Internet datasets, where they examined a normalized version of the joint degree distribution we have defined here. □

From the joint degree distribution we can define other useful summaries. For example, we might examine the distribution of the degrees of the neighbors of a vertex, conditional on the degree of that vertex. That is, we might examine relative frequencies of the form $f_{d'|d}$ with which edges from a vertex of degree d are connected to another vertex of degree d' . Plots of the means of these conditional distributions,

⁶ Our definition corresponds to the natural interpretation of an undirected graph as a symmetric directed graph. For general directed graphs, a separate joint degree distribution can be defined for in- and out-degrees, although it is no longer expected that these be symmetric. Alternatively, we can choose just to work with an undirected version of the graph, with degrees $d_i = d_i^{in} + d_i^{out}$.

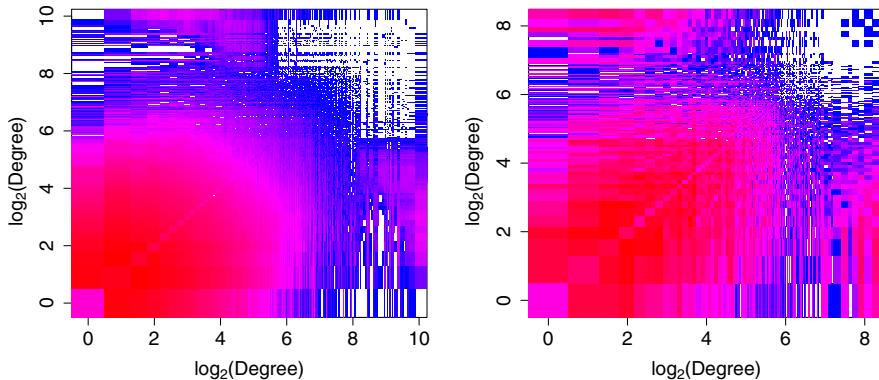


Fig. 4.3 Image representation of the logarithmically transformed joint degree distribution $\{\log_2 f_{d,d'}\}$, for the router-level Internet data (left) and the protein interaction data (right). Colors range from blue (low relative frequency) to red (high relative frequency), with white indicating areas with no data. Note that both x - and y -axes are also on base-2 logarithmic scales.

$$\bar{d}(d) = \sum_{d'} d' f_{d'|d} , \quad (4.5)$$

as a function of d , have been found useful in concisely summarizing the type of association between high- and low-degree vertices remarked upon in Example 4.3. In particular, a negative trend has been observed in $\bar{d}(d)$ as d increases (e.g., Maslov and colleagues [270, 271], Pastor-Satorras, Vázquez, and Vespignani [310], Vázquez, Pastor-Satorras, and Vespignani [387]).

A single number summary of the association between degrees of vertices on an arbitrary edge is provided by the correlation, say $\text{corr}(D, D')$, defined by the joint degree distribution $f_{d,d'}$ and its marginals. For the Internet and protein interaction networks, the degree correlation is found to be 0.023 and -0.083 , respectively. While both of these are relatively small, the difference in sign seems to reinforce our observation above that the tendency of high-degree vertices to pair with vertices of low degree is somewhat more pronounced in the protein interaction network.

Newman [288] has defined this same degree correlation through the concept of the ‘remaining degrees’ of vertices (i.e., the number of edges incident to a vertex, excluding the edge by which one arrives at that vertex). And Doyle et al. [125] have defined another related measure, one they call the ‘degree likelihood,’ which is proportional to the sum, over all edges, of the products $d(v_1)d(v_2)$ of the degrees of the vertices v_1, v_2 defining the edges $e = (v_1, v_2)$.

Of course, we would be remiss not to point out before closing this section that, while all of these various notions of degree correlation – both visual and numerical – can be useful, it will nonetheless typically be important to look beyond the numbers to ask the more interesting question of *why* an observed correlation is as

it is. More likely than not, the association among vertex degrees is actually driven by parallel associations among certain latent vertex characteristics. For example, the function or structure of proteins, or the placement and engineering of a router (e.g., in the Internet backbone, say, versus comparative ‘byways’), are likely to be more meaningful characteristics in our examples above than degree. In the event that such characteristics are measured, there are related notions of correlation – more commonly termed ‘assortativity’ in this context – that may be quantified, as we will see in Section 4.3.4.

4.2.2 Centrality

Many questions that might be asked about a vertex in a network graph essentially seek to understand its ‘importance’ in the network. Which actors in a social network seem to hold the ‘reins of power’? How authoritative does a particular page in the World Wide Web seem to be considered? The deletion of which genes in a gene regulatory network is likely to be lethal to the corresponding organism? How critical is a given router in an Internet network to the flow of traffic? Measures of *centrality* are designed to quantify such notions of ‘importance’ and thereby facilitate the answering of such questions.

There are a vast number of different centrality measures that have been proposed over the years. As observed by Freeman [157] in 1979, and evidently still true today, “There is certainly no unanimity on exactly what centrality is or on its conceptual foundations, and there is little agreement on the proper procedure for its measurement.” Attempts to categorize different centrality measures have involved trying to group them according to concepts or dynamics, like ‘distance,’ ‘flow,’ ‘feedback,’ and ‘control,’ as in Freeman [157] and the volume edited by Brandes and Erlebach [54, Ch. 3]; or, alternatively, by the manner in which they are computed, as in Borgatti and Everett [48].

We have already encountered what is arguably the most widely used measure of vertex centrality: vertex degree. Here we will focus our discussion primarily around the most common versions of three other classic types of vertex centrality measures – typically termed closeness, betweenness, and eigenvector centrality, respectively. Afterward we will comment on extensions, including notions of edge centrality.

4.2.2.1 Three Common Vertex Centrality Measures

We will assume that G is an undirected graph; extensions of the measures below to directed graphs are straightforward. One common notion of ‘central’ is that a vertex be ‘close’ to many other vertices. *Closeness centrality* measures attempt to capture this notion. The standard approach, introduced by Sabidussi [336], is to let the centrality vary inversely with a measure of the total distance of a vertex from all others,

$$c_{CI}(v) = \frac{1}{\sum_{u \in V} \text{dist}(v, u)} , \quad (4.6)$$

where $\text{dist}(v, u)$ is the geodesic distance between the vertices $v, u \in V$. Often, for comparison across graphs and with other centrality measures, this measure is normalized to lie in the interval $[0, 1]$, through multiplication by a factor $N_v - 1$.

To calculate the centrality in (4.6), for all $v \in V$, we need the shortest path distances between all pairs of vertices in G , which may be computed in $O(N_v^2 \log N_v + N_v N_e)$ time – essentially $O(N_v^2 \log N_v)$ on sparse graphs – using Dijkstra's algorithm, as discussed in Chapter 2.1.4. Note that implicitly this measure assumes that the graph G is connected, as otherwise all vertices in principle will have centrality $c_{CI}(v) = 0$, being of infinite distance from at least one other vertex. A conservative solution to the problem of insufficient connectivity would be to compute centrality indices separately for vertices in each connected component. Or, for example, if the graph consisted primarily of a single larger component, and a handful of other much smaller components, it might be reasonable to restrict attention to just the larger component. Another solution is to set a finite upper limit on distances between vertices; a choice of N_v is appealing, as clearly no vertices can be further than $N_v - 1$ from each other within a connected graph.

Another popular class of centralities are based upon the perspective that ‘importance’ relates to where a vertex is located with respect to the paths in the network graph. If we picture those paths as the routes by which, say, communication of some sort or another takes place, vertices that sit on many paths are likely more critical to the communication process. *Betweenness centrality* measures are aimed at summarizing the extent to which a vertex is located ‘between’ other pairs of vertices. The most commonly used betweenness centrality, introduced by Freeman [156], is defined as

$$c_B(v) = \sum_{s \neq t \neq v \in V} \frac{\sigma(s, t|v)}{\sigma(s, t)} , \quad (4.7)$$

where $\sigma(s, t|v)$ is the total number of shortest paths between s and t that pass through v , and $\sigma(s, t) = \sum_v \sigma(s, t|v)$. In the event that shortest paths are unique, $c_B(v)$ just counts the number of shortest paths going through v . This centrality measure can be restricted to the unit interval through division by a factor of $(N_v - 1)(N_v - 2)/2$.

Calculation of the collection of all betweenness centralities $c_B(v)$ requires (i) calculating the lengths of shortest paths among all pairs of vertices, and (ii) computing the summation in (4.7) for each vertex. As a result, the overall computational burden is dominated by the latter step, if implemented in a straightforward fashion, as each of the n sums require $O(N_v^2)$ time. The result is therefore an $O(N_v^3)$ algorithm, which can be prohibitive for large network graphs. However, Brandes [51] offers an algorithm for betweenness centrality that runs in as little as $O(N_v N_e)$ time on unweighted graphs – and $O(N_v^2 \log N_v + N_v N_e)$ time on weighted graphs – as well as requiring only $O(N_v + N_e)$ space, as compared to the $O(N_v^2)$ requirement of the

straightforward algorithm.⁷ These improvements derive from exploiting a clever recursive relation for the partial sums $\sum_{t \in V} \sigma(s, t|v) / \sigma(s, t)$.

A third class of centrality measures are based on notions of ‘status’ or ‘prestige’ or ‘rank.’ That is, they seek to capture the idea that the more central the neighbors of a vertex are, the more central that vertex itself is. These measures are inherently implicit in their definition and typically can be expressed in terms of eigenvector solutions of appropriately defined linear systems of equations. There are many such *eigenvector centrality* measures. For example, Bonacich [47], following work of Katz [220] and others, defined a centrality measure of the form

$$c_{Ei}(v) = \alpha \sum_{\{u,v\} \in E} c_{Ei}(u) . \quad (4.8)$$

The vector $\mathbf{c}_{Ei} = (c_{Ei}(1), \dots, c_{Ei}(N_v))^T$ is the solution to the eigenvalue problem $\mathbf{A}\mathbf{c}_{Ei} = \alpha^{-1}\mathbf{c}_{Ei}$, where \mathbf{A} is the adjacency matrix for the network graph G . Bonacich [47] argues that an optimal choice of α^{-1} is the largest eigenvalue of \mathbf{A} , and hence \mathbf{c}_{Ei} is the corresponding eigenvector. When G is undirected and connected, the largest eigenvector of \mathbf{A} will be simple and its eigenvector will have entries that are all nonzero and share the same sign. Convention is to report the absolute values of these entries, which will automatically lie between 0 and 1 by the orthonormality of eigenvectors.

Calculation of the largest eigenvalue of a matrix and its eigenvector is a standard problem. The power method is generally used, a description of which can be found in most textbooks on computational linear algebra (e.g., Golub and van Loan [181]). This method is iterative and is guaranteed to converge under various conditions, such as when the matrix is symmetric, which \mathbf{A} will be for undirected graphs. The rate of convergence to \mathbf{c}_{Ei} will behave like a power, in the number of iterations, of the ratio of the second largest eigenvalue of \mathbf{A} to the first. Since the power method involves only matrix-vector multiplications, it requires only $O(N_v^2)$ operations per iteration, which places it roughly on par with the other two methods described above.

Each of the three centrality measures defined so far – based on closeness, betweenness, and eigenvectors – are well motivated by particular interpretations of the term ‘central’. However, as these interpretations differ from each other, it should not be surprising that the results obtained by applying them can differ accordingly.

Example 4.4 (Comparing Centrality Measures). Consider the small toy graph shown in Figure 4.4(a), consisting of just $N_v = 11$ vertices and $N_e = 12$ edges. Which vertices are more ‘central’ and which less? Clearly it depends to some extent upon the context in which the question is posed. Arguably the four small green vertices at the extremes of the graph (i.e., top and bottom) are not likely to be considered central under any intuitively appealing definition of the word. But it is less clear how the other vertices compare – particularly the red, yellow, and dark blue vertices.

⁷ In fact, modifications of this algorithm yield similarly efficient algorithms for various other notions of betweenness centrality and for other common centrality measures, such as the closeness betweenness $c_{Cl}(v)$ defined previously. See Brandes [51, 52].

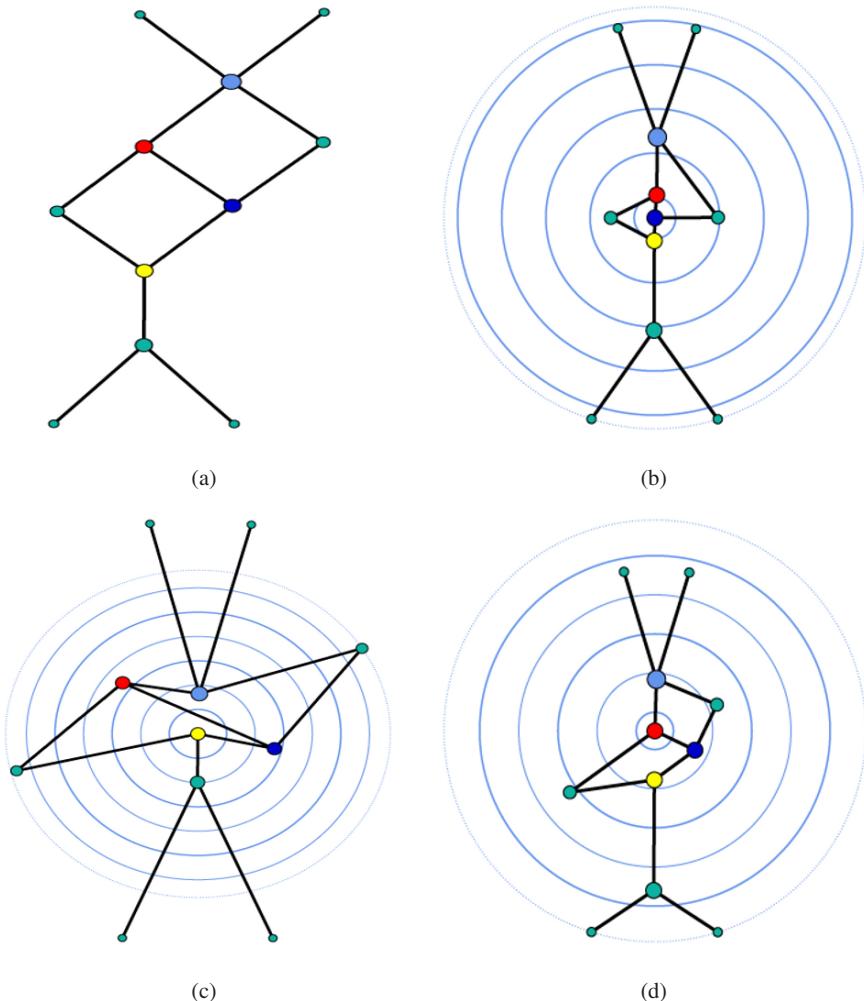


Fig. 4.4 Illustration of (b) closeness, (c) betweenness, and (d) eigenvector centrality measures on the graph in (a). Example and figures courtesy of Ulrik Brandes.

Figures 4.4(b) – (d) provide visual summaries of the closeness, betweenness, and eigenvector centralities, respectively, of the vertices in our toy graph. In each case, vertices are arranged using a radial layout, with more central vertices located closer to the center.

Under the closeness centrality, $c_{Cl}(v)$, the dark blue vertex is judged to be most central under this measure, followed closely by the red and yellow vertices. However, under the betweenness centrality, $c_B(v)$, it can be seen that the yellow vertex is now judged to be most central. In fact, note that the relative positions of all but the most extreme vertices (i.e., small green) have changed noticeably from Fig-

ure 4.4(b). Evidently, that a vertex lies close to other vertices does not necessarily mean it must also lie on many shortest paths between pairs of other vertices. The yellow and light blue vertices, as well as the large green vertex immediately beneath the yellow vertex, all essentially serve as ‘gate-keepers’ on paths connecting to extreme vertices, and the values $c_B(v)$ reflect this accordingly. Lastly, under the eigenvector centrality, $c_{Ei}(v)$, the relationship among vertices is again different from that which it was under the other two centrality measures, although it is arguably closer in nature to that yielded by the closeness centrality. Here the red vertex is now judged to be most central, with four of the other ten vertices sharing a similar centrality as runners up. \square

4.2.2.2 Additional Measures of Centrality

Example 4.4, although a ‘toy,’ is far from diabolical and is intended to illustrate how even in a seemingly unremarkable graph different centrality measures can produce distinctly different orderings of the vertices. As mentioned at the start of this section, there are in fact many additional vertex centrality measures. Deciding which are most appropriate for a given application clearly requires consideration of the context.

The centrality measures defined above have all had numerous variations and extensions defined. Closeness centrality is a simple inverse function of distance on the graph. Another related idea, proposed by Hage and Harary [190], is to measure centrality as inversely proportional to the eccentricity of a vertex (i.e., the length of the longest shortest path on which it serves as an endpoint). Betweenness centrality counts all shortest paths passing through a vertex. Variations on this idea have suggested modifying the underlying path set, instead counting, for example, only shortest paths of length bounded by some k or shortest paths that are disjoint in their vertices and/or edges. See Borgatti and Everett [48] for pointers to a number of such proposals, as well as White and Smyth [402]. Finally, the notion of eigenvector centrality already had a number of very similar precursors prior to Bonacich’s formulation. See Wasserman and Faust [393], for example, for background and references. Additional pointers to references and surveys may be found in Section 4.6.

Also worth special note are two fairly celebrated algorithms of a nature closely related to eigenvector centrality that have emerged from the computer science community in the context of judging the importance of web pages. The first is the PageRank algorithm (Page, Brin, Motwani, and Winograd [307]), a key component of the Google search engine (i.e., see Brin and Page [63]), which poses a model for a random ‘surfing’ of the Web, and produces centralities that are the solution of a rescaled and damped version of the eigensystem in earlier methods of this genre. The second is the HITS algorithm, based on the concept of ‘hubs and authorities,’ introduced by Kleinberg [230], which characterizes the importance of so-called hub web pages by how many authority web pages they point to, and so-called authority web pages by how many hubs point to them. Specifically, given an adjacency matrix \mathbf{A} for a directed web graph, hubs are determined according to the eigenvector centrality of the

matrix $\mathbf{M}_{hub} = \mathbf{A}\mathbf{A}^T$, and authorities, according to that of $\mathbf{M}_{auth} = \mathbf{A}^T\mathbf{A}$. The matrices \mathbf{M}_{hub} and \mathbf{M}_{auth} are analogous to quantities encountered in bibliometrics, where the former counts the number of common third works cited by a pair of works, and the latter counts the number of times two works were both cited by a third work. Both the PageRank and the ‘Hubs and Authorities’ algorithms have already had additional extensions and variations proposed.

There are also centrality measures deriving from the notion of flows on a network graph (e.g., Freeman, Borgatti, and White [159]), and the closely related idea of a random walk on a graph (e.g., Newman [295]). These essentially introduce another perspective on the concept of betweenness, and can be shown to have interesting connections to closeness-based centrality measures (e.g., White and Smyth [402], Brandes and Fleischer [55]).

All of the centralities discussed so far are for vertices, as it seems to be most common in practice that questions of importance are in regard to the vertices of a graph. But there are contexts where edges arguably are of greater concern than the vertices – for example, links in a physical Internet traffic network. Betweenness vertex centrality extends to edges in a straightforward manner. However, many other vertex centrality measures do not. One way around this problem is to apply vertex centrality measures to the vertices in the dual graph of a network graph G . The *dual graph* of G , say $G' = (V', E')$, is obtained essentially by changing vertices of G to edges, and edges, to vertices. More formally, the dual vertices $v' \in V'$ represent the original edges $e \in E$, and the dual edges $e' \in E'$ indicate that the two corresponding original edges in G were incident to a common vertex in G . See Chapter 3 of the edited volume of Brandes and Erlebach [54] for a brief discussion.

Lastly, we point out that for any given centrality measure $c(v)$, while it is defined on a vertex-specific (or edge-specific) basis, it may be interesting and useful in the course of an analysis to look at aggregate graph-level summaries. For instance, we can look at the distribution of the values $\{c(v)\}_{v \in V}$, in analogy to the degree distribution, and, obviously, functions of that distribution, such as moments and quantiles. In the social network literature, it has traditionally also been popular to use the *centralization index* of Freeman [157]

$$c = \frac{\sum_{v \in V} [c^* - c(v)]}{\max \sum_{v \in V} [c^* - c(v)]} , \quad (4.9)$$

where c^* is the maximum of the $c(v)$ over the graph G , and the max in the denominator is over all possible graphs of order N_v . In practice, however, this quantity is not easily calculated for most $c(v)$, outside of certain special cases.

As an intermediate choice, between single-vertex and whole-graph centrality measures, Everett and Borgatti [139] offer extensions of vertex degree, closeness, and betweenness centralities to groups of arbitrary size. See also Kolaczyk, Chua, and Barthélemy [233], for additional work on group-level extensions of vertex betweenness centrality, and Puza, Elovici, and Dolev [316], who provide a fast algorithm for successive computation of group betweenness.

4.3 Characterizing Network Cohesion

A great many questions in network analysis boil down to questions involving *network cohesion*, the extent to which subsets of vertices are cohesive – or ‘stuck together’ – with respect to the relation defining edges in the network graph. Do friends of a given actor in a social network tend to be friends of one another as well? What collections of proteins in a cell appear to work closely together? Does the structure of the pages in the World Wide Web tend to separate with respect to distinct types of content? What portion of a measured Internet topology would seem to constitute the ‘backbone’?

There are many ways that we can define network cohesion, depending on the context of the question being asked. Definitions differ, for example, in scale, ranging from local (e.g., triads) to global (e.g., giant components), and also in the extent to which they are specified explicitly (e.g., cliques) versus implicitly (e.g., ‘clusters’). In this section we discuss a number of common ways to define and summarize ‘cohesion’ in a network graph.⁸

4.3.1 Local Density

Many notions of coherent network structure are based on the principle that a coherent subset of nodes should be locally ‘dense’ in the graph. From a graph-theoretic point of view, one immediately obvious concept to employ in this regard is that of a *clique* – a complete subgraph H of G . Cliques are subsets of vertices that are fully cohesive, in the sense that all vertices within the subset are connected by edges. A case of common practical interest, particularly in social network analysis, is that of 3-cliques (i.e., triangles). In practice, large cliques are relatively rare, as they necessarily require that G itself be fairly dense. For example, Turán [384] shows that a sufficient condition for a clique of size n to exist in G is that $N_e > (N_v^2/2)[(n-2)/(n-1)]$. This condition may be contrasted with the fact that real-world networks generally have been found to have N_e closer to the same order as N_v .

There are many questions that we might ask involving cliques in a graph, but it should be noted that the corresponding algorithms for calculating answers can range in computational complexity from linear to NP -complete. For instance, both the determination of whether a specific subset of nodes $U \subseteq V$ is a clique and whether it is maximal can be done in $O(N_v + N_e)$ time. On the other hand, the problem of determining whether a graph G has a maximal clique of at least size n is NP -complete. Certain other problems lie between these two extremes. For example,

⁸ One could also arguably use the term ‘group’, in its colloquial sense, in reference to the various types of cohesive network sub-structures we seek to characterize with the tools in this section. However, in the field of social network analysis, from whence many of the tools in this chapter come, the notion of a social group has certain more precise connotations. In order to avoid confusion in this regard, we have chosen to use the somewhat more formal sounding term ‘cohesion’.

detecting all triangles (i.e., all cliques of size $n = 3$) in G can be done exhaustively in $O(N_v^3)$ time, which can be improved to $O(N_v^{2.376})$ time in general and $O(N_e^{1.41})$ time in sparse graphs through use of the adjacency matrix \mathbf{A} of G and fast algorithms for multiplication of square matrices. See Chapter 6 of the edited volume of Brandes and Erlebach [54] for additional details on these and similar results.

Ultimately, cliques tend to be an overly restrictive definition of network cohesion, and therefore are not used extensively in practice in and of themselves. Instead, weakened versions of this idea tend to be more practical. One such notion is that of *plexes*. A subgraph H consisting of m vertices is called an n -plex, for $m > n$, if no vertex has degree less than $m - n$. In other words, we specify that no vertex is missing more than n of its possible $m - 1$ edges with the other vertices in the subgraph. Unfortunately, as intuitively appealing as this variation may be, from a computational perspective problems involving n -plexes tend to scale like those involving cliques.

Much more computationally tractable is a dual notion of cohesiveness, that of *cores*, the utility of which we have already seen demonstrated in Chapter 3 in the context of the router-level Internet. Cores also have found successful usage in problems involving protein interaction networks, including the search for molecular complexes (Bader and Hogue [15]), the prediction of protein function (Altaf-Ul-Amin et al. [7]), and the study of evolutionary conservation (Wuchty and Almaas [408]). Recall from Section 3.5.2.3 that a k -core of a graph G is a subgraph H for which all vertices have degree at least k . A maximal k -core subgraph may be computed in as little as $O(N_v + N_e)$ time (Batagelj, Mrvar, and Zaversnik [25]). In fact, the algorithm computes the shell indices for every vertex in the graph, where the shell index of $v \in V$ is the largest value, say c , such that v belongs to the c -core of G but not its $(c + 1)$ -core. The efficiency of the algorithm is due to iterative use of (i) the observation that, for a given vertex, those neighbors with lesser degree lead to a decrease in the potential shell index of that vertex, and (ii) an efficient technique for sorting vertices by degree.

All of the approaches to network cohesion above proceed by first stating a pre-specified notion of locally dense structure and then looking to see whether it is anywhere satisfied in a graph G and, if so, where. Alternatively, we can instead define a measure of local density and then characterize the extent to which subsets of vertices are dense, according to this measure. Such measures are commonly based on ratios of the number of edges among a subset of vertices to the total number of possible edges. For example, in a graph G with no self-loops and no multiple edges, the *density* of a subgraph $H = (V_H, E_H)$ is

$$\text{den}(H) = \frac{|E_H|}{|V_H|(|V_H| - 1)/2} . \quad (4.10)$$

The value of $\text{den}(H)$ will lie between zero and one and provides a measure of how close H is to being a clique. Note that $\text{den}(H)$ is just a rescaling of the average degree $\bar{d}(H)$ of H , since $\text{den}(H) = (|V_H| - 1)\bar{d}(H)$.

There is freedom in the choice of subgraph H defining (4.10). Taking $H = G$ yields the density of the overall graph G . Conversely, taking $H = H_v$ to be the set of neighbors of a vertex $v \in V$, and the edges between them, yields a measure of density in the immediate neighborhood of v . Watts and Strogatz [396] propose $\text{den}(H_v)$ as a summary of the extent to which there is ‘clustering’ of edges local to v and propose that the average of $\text{den}(H_v)$ over all vertices be used as a *clustering coefficient* for the overall graph.

These measures of clustering can be expressed alternatively in terms of the density of triangles among connected triples. A *triangle* is a complete subgraph of order three. A *connected triple* is a subgraph of three vertices connected by two edges (also sometimes called a 2-star). Intuitively, a measure of the frequency with which connected triples ‘close’ to form triangles will provide some indication of the extent to which edges are ‘clustered’ in the graph. Let $\tau_\Delta(v)$ denote the number of triangles in G into which $v \in V$ falls, and $\tau_3(v)$, the number of connected triples in G for which the two edges are both incident to v . Note that $\tau_3(v) = \binom{d_v}{2}$. The Watts-Strogatz local clustering coefficient $\text{den}(H_v)$ can be re-expressed as $\text{cl}(v) = \tau_\Delta(v)/\tau_3(v)$, for those vertices v with $\tau_3(v) > 0$. The corresponding clustering coefficient for G takes the form⁹

$$\text{cl}(G) = \frac{1}{V'} \sum_{v \in V'} \text{cl}(v) , \quad (4.11)$$

where $V' \subseteq V$ is the set of vertices v with $d_v \geq 2$.

The value $\text{cl}(G)$ is a popular choice of clustering coefficient in the literature. However, as Bollobás and Riordan [45] have pointed out, being an average of averages, this quantity may be less informative than the appropriate weighted average, which takes the form

$$\frac{\sum_{v \in V'} \tau_3(v) \text{cl}(v)}{\sum_{v \in V'} \tau_3(v)} . \quad (4.12)$$

The quantity (4.12) in turn is equal to

$$\text{cl}_T(G) = \frac{3\tau_\Delta(G)}{\tau_3(G)} , \quad (4.13)$$

where $\tau_\Delta(G) = (1/3)\sum_{v \in V} \tau_\Delta(v)$ is the number of triangles in the graph, and $\tau_3(G) = \sum_{v \in V} \tau_3(v)$, the number of connected triples. The value $\text{cl}_T(G)$, called the *transitivity* of the graph, is a standard quantity in the social network literature, where it is also referred to as the ‘fraction of transitive triples.’ Transitivity in this context refers to, for example, the case where the friend of your friend is also a friend of yours.

Note that, while the two clustering coefficients cl and cl_T often yield similar values for a graph G , they are not equivalent, and in fact can differ noticeably in some cases. For example, in the karate club network of Chapter 1, where it is easy to see many triads and triangles, the clustering coefficient values are $\text{cl}(G) = 0.5879$

⁹ There is some variation in the literature in defining this quantity, depending on the treatment of vertices v with degree zero or one. See also Barrat and Weigt [19].

and $\text{cl}_T(G) = 0.2557$. It is even possible to define highly imbalanced graphs so that $\text{cl}(G) \rightarrow 1$ while $\text{cl}_T(G) \rightarrow 0$, as N_v grows. See Bollobás and Riordan [45, pg. 18], or the volume edited by Brandes and Erlebach [54, Ch. 11]. In practice, therefore, it is important to clearly state which clustering coefficient is being reported.

The key requirement for calculating either of these clustering coefficients is an ability to identify connected triples $\{v, u_1, u_2\}$ at each vertex $v \in V$ and to check whether the pairs $\{u_1, u_2\}$ form an edge in E . These steps are inherent in the algorithms for enumerating the triangles in a graph, already described above, and hence the complexity for computing these clustering coefficients scales accordingly, using these algorithms.

Clustering coefficients have become a standard quantity used in the analysis of network structure. Interestingly, their values have typically been found to be quite large in real-world networks, in comparison to what otherwise might be expected based on classical random graph models.¹⁰ In large-scale networks with broad degree distributions, it has frequently been found that the local clustering coefficient $\text{cl}(v)$ varies inversely with vertex degree. See Newman [296, Sec. III.B], for a number of references.

Higher-order clustering coefficients have also been proposed, involving cycles of length greater than three (i.e., recall that the triangles, upon which cl and cl_T are based, are 3-cycles). For example, Caldarelli, Pastor-Satorras, and Vespignani [72] and Newman [290] consider the use of four-cycles, and Fronczak, Hołyst, Jedynak, and Sienkiewicz [163] consider arbitrary k -cycles.

4.3.2 Connectivity

In contrast to a local perspective, and the search for small-scale subsets of cohesive vertices, we may also find it useful in some contexts to take a larger, global perspective. A basic question of interest is whether a given graph separates into distinct subgraphs. If it does not, we might seek to quantify how close to being able to do so it is. Intimately related to such issues are questions associated with the flow of ‘information’ in the graph.

4.3.2.1 Connected Components and ‘Small Worlds’

Recall that a graph G is said to be connected if every vertex is reachable from every other (i.e., if for any two vertices, there exists a walk between the two). A *connected component* of a graph is a maximally connected subgraph.

In practice, the process of constructing a network graph, as described in Chapter 3, may not necessarily result in a single connected graph. Rather, the graph

¹⁰ That is, for example, as compared to a graph where edges are assigned to pairs of vertices randomly through independent coin flips with common probability $p \in (0, 1)$. See Section 6.2.1.

may consist of a number of separate connected components. The task of verifying whether a graph is connected and, if not, identifying its connected components, can be done in $O(N_v + N_e)$ time, using either depth-first search (DFS) or breadth-first search (BFS) algorithms. See the text by Cormen, Leiserson, Rivest, and Stein [100], for example.

Often it is the case that one of the connected components in a graph G dominates the others in magnitude, in that it contains the vast majority of the vertices in G . Such a component is called, borrowing terminology from random graph theory, the *giant component*. In the case of an unconnected graph with a giant component, depending on the task at hand, it may be sensible to restrict attention to that component alone in carrying out further analysis and modeling. For example, the network of protein interactions described in Section 4.2.1.1 has a giant component with 5,128 vertices, and 11 smaller components, one with three vertices, and 10 with two vertices. An algorithm that seeks to predict the biological function of a given protein, based on knowledge of the functions of its neighbors in this network, such as those described in Chapter 8, is most sensibly applied to the giant component.

A celebrated characteristic observed in the giant component of many real-world networks is the so-called *small world* property. The initial concept and terminology for this term is usually traced back to the now-famous experiment of Stanley Milgram [277], in the late 1960's, and his assertion that people are only separated by roughly six acquaintances (i.e., by 'six degrees of separation,' as in the play of Guare [188] of the same name). That is, if we were to represent the population of the world and acquaintances among its members as a social network graph, despite its enormous size, the typical number of 'hops' along shortest paths between its vertices would be quite small.

Formally, in analyzing a network graph, the term 'small' here is usually meant to indicate the average distance between distinct vertices, say

$$\bar{l} = \frac{1}{N_v(N_v + 1)/2} \sum_{u \neq v \in V} \text{dist}(u, v) , \quad (4.14)$$

scales as $O(\log N_v)$ or less. For example, the protein interaction network would be considered to have the small-world property in this sense, as there is an average distance of only 3.68 among the 5,128 vertices in its giant component; in fact, the maximum distance is only 9. The discovery that such a property in fact holds for many types of networks is attributed to Watts and Strogatz [396]. These authors also found that often such small average distance \bar{l} is accompanied by a high clustering coefficient cl or cl_T , and they proposed a random graph model for such networks – see Section 6.3. It is now formally in the sense of these two properties holding jointly that the term 'small world' is used, although informally it is still often used to refer to small average distance alone.

It should not be altogether surprising – at least in hindsight – that networks might have such a property, as the following informal argument shows. If each vertex in the network graph has roughly d immediate neighbors, then we expect to find roughly d^2 neighbors within $h = 2$ hops of any vertex, d^3 within $h = 3$ hops, and so on. With

this exponentially increasing size of successive neighborhoods, we soon expect to reach a value h^* for which the expression $d^{h^*} \approx N_v$ holds. From this it can be seen that $h^* \approx \log N_v / \log d$ should scale logarithmically in N_v . Of course, this argument is simplistic in assuming an essentially homogeneous distribution of vertex degrees and ignoring any effects due to variations in local density. Nevertheless, the end result remains valid in more complicated models, as we will discuss in Section 6.3.

On the other hand, surprising or not, it is clear that the ‘small world’ property has potentially important implications on a host of fundamental problems relating to networks and ‘communication’ upon them, whether referring to the exchange of gossip in a social network, the search for content in a World Wide Web network, or the exchange of an infectious disease in an epidemiological network. We will return to this topic in our discussion of dynamic processes on networks in Chapter 8.

4.3.2.2 Vertex/Edge-Connectivity, Cuts, and Flows

A somewhat more refined notion of connectivity derives from asking questions like, “If an arbitrary subset of k vertices (edges) is removed from a graph, is the remaining subgraph connected?” Picture, for example, the extreme case where G consists of two cliques joined by a single edge between a vertex in each. A useful image here is that of a ‘barbell’ – two masses joined by a single, straight rod. Intuitively, we would be inclined to think of G as consisting of ‘nearly two components.’ The concepts of vertex- and edge-connectivity, and the related concepts of vertex- and edge-cuts, help to make these notions precise.

A graph G is called *k-vertex-connected* if (i) $N_v > k$, and (ii) the removal of any subset of vertices $X \subset V$ of cardinality $|X| < k$ leaves a subgraph $G - X$ that is connected. Note that, for graphs G with at least two vertices, G is 1-vertex-connected if and only if it is connected. Similarly, G is called *k-edge-connected* if (i) $N_v \geq 2$, and (ii) the removal of any subset of edges $Y \subseteq E$ of cardinality $|Y| < k$ leaves a subgraph $G - Y$ that is connected. The *vertex (edge) connectivity* of G is the largest integer such that G is k -vertex- (k -edge-) connected. It can be shown that the vertex connectivity is bounded above by the edge connectivity, which in turn is bounded above by the minimum degree d_{min} among vertices in G .

Standard questions of interest involving these quantities include: “Is G k -vertex- (k -edge-) connected?” “What is the vertex(edge)-connectivity of G ?” and “What are the maximal k -vertex- (k -edge-) connected components of G ?” Algorithms exist for answering these questions that run in polynomial time. For example, the vertex-connectivity (edge-connectivity) of a graph can be calculated in time that runs like the connectivity k times N_v^2 (i.e., somewhere between quadratic and cubic time). See the volume edited by Brandes and Erlebach [54, Ch. 7], for example.

A fundamental result in graph theory, known as Menger’s theorem, essentially states that a nontrivial graph G is k -vertex (k -edge) connected if and only if all pairs of distinct vertices $u, v \in V$ can be connected by k vertex-disjoint (edge-disjoint) paths. This result relates the robustness of a graph in the face of removal of its vertices (edges) to the richness of distinct paths running throughout it. A graph

with low vertex (edge) connectivity therefore can have the paths, and hence any ‘information’ flowing over those paths, disrupted by removing an appropriate choice of a correspondingly small number of vertices (edges).

If the removal of a particular set of vertices (edges) in G actually disconnects the graph, that set is called a *vertex-cut (edge-cut)*. Sometimes it is of interest to find cut sets that are in some sense ‘minimal.’ For a given pair of vertices $u, v \in V$, a u - v -cut is a partition of V into two disjoint, non-empty subsets $S, \bar{S} \subset V$, where $u \in S$ and $v \in \bar{S}$. For G equipped with edge weights w_e , a u - v -cut is said to be a *minimum u - v -cut* if the sum of the weights on edges connecting vertices in S to vertices in \bar{S} is a minimum. If the edge weights are all of unit value (i.e., $w_e \equiv 1$), then finding a minimum u - v -cut is equivalent to finding an edge-cut of minimal cardinality, with one component containing u and the other containing v . Note that if the smallest cardinality of such minimum edge-cut sets, over all distinct $u, v \in V$, is say k , then clearly the edge-connectivity of G is $k - 1$.

To find a minimum u - v -cut, the problem typically is translated to an equivalent problem of maximizing a measure of flow on the edges of a derived directed graph, subject to certain constraints, by invoking the Max-Flow Min-Cut theorem of Ford and Fulkerson [146]. There are many algorithms that have been developed for solving the max-flow problem, all of which run in polynomial time. The algorithm of Goldberg and Tarjan [177], for example, runs in $O(N_v N_e \log(N_v^2 / N_e))$ time. To find a minimum cut over all pairs of vertices it is sufficient to make only $N_v - 1$ calls to a max-flow algorithm (Gomory and Hu [182]). Alternatively, more efficient would be to use the non-flow-based algorithm of Stoer and Wagner [369], which runs in $O(N_e N_v + N_v^2 \log N_v)$.

4.3.2.3 Connectivity in Directed Graphs

Most of the concepts introduced above, and their corresponding algorithmic implementations, extend to the case of directed graphs in a straightforward manner. Recall from Chapter 2, for example, that a digraph G is *weakly connected* if its underlying graph (i.e., the result of stripping away the labels ‘tail’ and ‘head’ from G) is connected, and that it is called *strongly connected* if every vertex v is reachable from every other vertex u by a directed walk. The concept of vertex-connectivity (edge-connectivity) extends analogously by requiring that the graphs remaining after removal of vertices (edges) be strongly connected. Similarly, the notion of cuts and flows remain essentially unchanged, except that now there is a directionality designated. For example, in a cut (S, \bar{S}) , we would refer to one of the sets, say S , as the *source*, and other, \bar{S} , as the *sink*, suggestive of the movement of flows from S to \bar{S} .

An often-useful characterization of directed network graphs is that of a ‘bowtie,’ shown schematically in Figure 4.5. The bowtie structure was first described by Broder and colleagues [67], in the context of the World Wide Web graph. This structure consists of five parts. The first part is a strongly connected component (SCC). The second is an *in-component*, which consists of vertices from which the

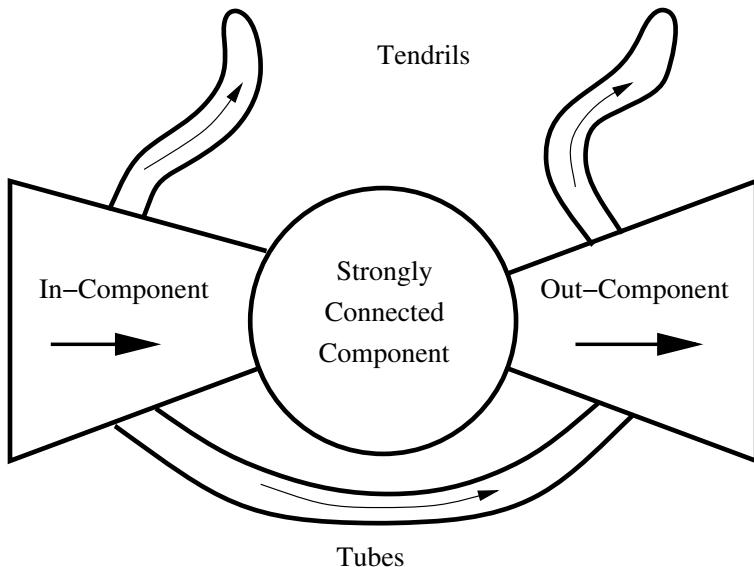


Fig. 4.5 ‘Bowtie’ structure of a directed network graph. Adapted from Broder et al. [67].

SCC can be reached, but which themselves cannot be reached from the SCC. Third is an analogous *out-component*, whose vertices can be reached from the SCC but from which the SCC cannot be reached. The in- and out-components are in some sense ‘upstream’ and ‘downstream’ from the SCC, respectively, and are thus typically drawn to the left and right of the SCC, as in Figure 4.5. The fourth and fifth parts of the bowtie structure are the *tubes* and *tendrils*. The former is composed of vertices between the in- and out-components that are not part of the SCC, while the latter consists of vertices that can neither reach nor be reached from the SCC.

Example 4.5 (Bowtie Structure of the AIDS Blog Network). Figure 4.6 shows a bowtie decomposition of the AIDS blog network from Chapter 1. It can be seen that the SCC is quite small (four vertices), as is the in-component (two vertices). The network is dominated by the out-component (112 vertices), and a few tendrils (28 vertices). Note that in large-scale networks, the digraph G may be disconnected. So the SCC may in fact be a giant strongly connected component (GSCC), with corresponding giant in- and out-components. In the context of measurements on the World Wide Web graph – vastly larger than the tiny blog network shown here – Broder and colleagues found that the relative size of these four parts of the bowtie in their giant component were in fact roughly equal in size. □

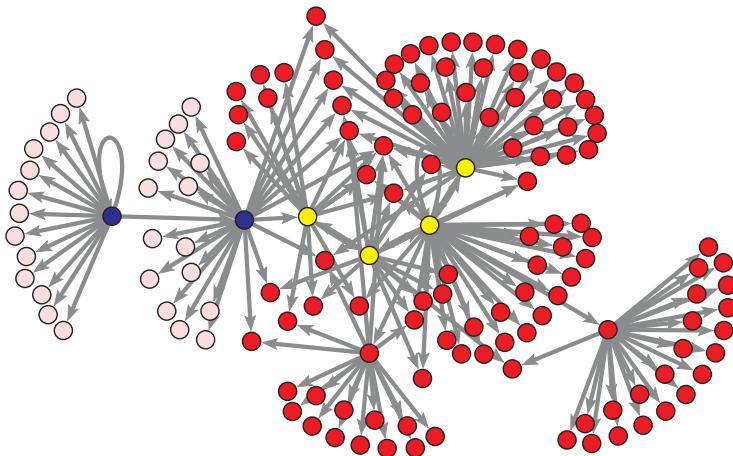


Fig. 4.6 AIDS blog network, from Chapter 1, with nodes colored according to their membership in the ‘bowtie’ decomposition: strongly connected component (yellow), in-component (blue), out-component (red), and tendrils (pink).

4.3.3 Graph Partitioning

Partitioning – broadly speaking – refers to the segmentation of a set of elements into ‘natural’ subsets. More formally, a *partition* $\mathcal{C} = \{C_1, \dots, C_K\}$ of a finite set S is a decomposition of S into K disjoint, nonempty subsets C_k such that $\cup_{k=1}^K C_k = S$. The need for partitioning arises in various scientific disciplines, in the context of problems ranging from image segmentation to pattern recognition to information retrieval. In the analysis of network graphs, partitioning is a useful tool for finding, in an unsupervised fashion, subsets of vertices that demonstrate a ‘cohesiveness’ with respect to the underlying relational patterns. It has been used, for example, in the detection of community structure in social networks (e.g., Girvan and Newman [174]) and in the identification of possible protein complexes from protein interaction networks (e.g., Sen, Kloczkowski, and Jernigan [346], and the many references therein).

A ‘cohesive’ subset of vertices generally is taken to refer to a subset of vertices that (i) are well connected among themselves, and at the same time (ii) are relatively well separated from the remaining vertices. More formally, graph partitioning algorithms typically seek a partition $\mathcal{C} = \{C_1, \dots, C_K\}$ of the vertex set V of a graph $G = (V, E)$ in such a manner that the sets $E(C_k, C_{k'})$ of edges connect-

ing vertices in C_k to vertices in $C_{k'}$ are relatively small in size compared to the sets $E(C_k) = E(C_k, C_k)$ of edges connecting vertices within the C_k .

While there are a number of standard methods of graph partitioning in use, the development of new methods is nevertheless also a very active area of research, a fact that is at least in part suggestive of the inherent difficulty of the problem. We will describe two well-established classes of methods – those based on adaptations of hierarchical clustering and those based on principles of spectral partitioning – in some detail, and then briefly discuss a handful of other methods, including a number of those proposed more recently.

4.3.3.1 Hierarchical Clustering

A great many methods for graph partitioning are essentially variations on the more general concept of *hierarchical clustering* used in data analysis.¹¹ There are numerous techniques that have been proposed for the general clustering problem, differing primarily in (i) how they evaluate the quality of proposed clusterings and (ii) the algorithms by which they seek to optimize that quality. See Jain, Murty, and Flynn [213], for example. These methods take a greedy approach to searching the space of all possible partitions \mathcal{C} , by iteratively modifying successive candidate partitions. Hierarchical methods are classified as either *agglomerative*, being based on the successive coarsening of partitions through the process of merging, or *divisive*, being based on the successive refinement of partitions through the process of splitting. At each stage, the current candidate partition is modified in a way that minimizes a specified measure of cost. In agglomerative methods, the least costly merge of two previously existing partition elements is executed, whereas in divisive methods, it is the least costly split of a single existing partition element into two that is executed.

Whether agglomerative or divisive, when used for network graph partitioning, hierarchical clustering methods actually produce, as the name indicates, an entire hierarchy of nested partitions of the graph, not just a single partition. These partitions can range fully between the two trivial partitions $\{\{v_1\}, \dots, \{v_{N_v}\}\}$ and V . Agglomerative methods begin with the former of these two, while divisive methods begin with the latter. The resulting hierarchy typically is represented in the form of a tree, called a *dendrogram*, such as that shown in Figure 4.7.

The measure of cost incorporated into a hierarchical clustering method used in graph partitioning should reflect our sense of what defines a ‘cohesive’ subset of vertices. It also can have important implications on the computational complexity of the overall algorithm implementing the method. At the core of most of these cost functions is a measure of (dis)similarity, say x_{ij} , between pairs of vertices $v_i, v_j \in V$. Methods differ not only in how the (dis)similarity between vertex pairs is defined,

¹¹ Note that ‘clustering’ as used here, which is standard terminology in the broader data analysis community, differs from ‘clustering’ as used in Section 4.3.1, which arose in the social network community, in reference to the coefficients cl and cl_T used to summarize the relative density of triangles among connected triples.

but also in how (dis)similarity between sets of vertices is defined. For example, in agglomerative algorithms, given two sets of vertices C_1 and C_2 , two standard approaches to assigning a similarity value to this pair of sets is to use the maximum (called *single-linkage*) or the minimum (called *complete linkage*) of the x_{ij} , over all pairs $v_i \in C_1$ and $v_j \in C_2$. Other approaches can be more involved. For example, the default method in the software package **Pajek** is based on a method similar to that of Ward [391]. Ward's method utilizes group sums of squares, but is intended for Euclidean data. The **Pajek** implementation involves an extension of this method to non-Euclidean data, as described in Batagelj [22].

Example 4.6 (Hierarchical Clustering of the Karate Club Network). The dendrogram in Figure 4.7 shows the results of an agglomerative analysis of the karate club network shown in Figure 1.2, and described in Section 1.2.2. These results are based on a dissimilarity x_{ij} for vertices v_i and v_j defined as

$$x_{ij} = \frac{|\mathcal{N}_{v_i} \Delta \mathcal{N}_{v_j}|}{d_{(N_v)} + d_{(N_v-1)}} , \quad (4.15)$$

where \mathcal{N}_v is the set of neighbors of a vertex v , ‘ Δ ’ indicates symmetric difference of two sets,¹² and $d_{(i)}$ is the i -th smallest element in the degree sequence. In other words, x_{ij} is just the number of neighbors of v_i and v_j that are not shared, normalized to lie in the interval $[0, 1]$, so that 0 and 1 indicate perfect similarity and dissimilarity, respectively. Dissimilarities for subsets of vertices were calculated from the x_{ij} using the extension of Ward's method mentioned above, and the lengths of the branches in the dendrogram are in relative proportion to the changes in dissimilarity with successive splits. Generally, interest is in partitions immediately following a comparatively long set of branches. With this particular network, we know that the karate club eventually split into two separate clubs. The first partition, into just two subsets of vertices, separates the network precisely according to the eventual breakup. \square

There are various other common choices of (dis)similarity measures, such as the so-called ‘Euclidean distance’ dissimilarity

$$x_{ij} = \sqrt{\sum_{k \neq i,j} (A_{ik} - A_{jk})^2} . \quad (4.16)$$

Here \mathbf{A} is the adjacency matrix, and x_{ij} measures the standard Euclidean distance between rows i and j . Hierarchical clustering algorithms based on (dis)similarities of this sort are reasonably efficient, running in $O(N_v^2 \log N_v)$ time. Presuming the N_v^2 (dis)similarity values can be computed in $O(N_v^2)$ time at most, as is the case on

¹² The *symmetric difference* of two sets S_1 and S_2 is the set of elements that are in one or the other but not both, $S_1 \Delta S_2 = (S_1 - S_2) \cup (S_2 - S_1)$.

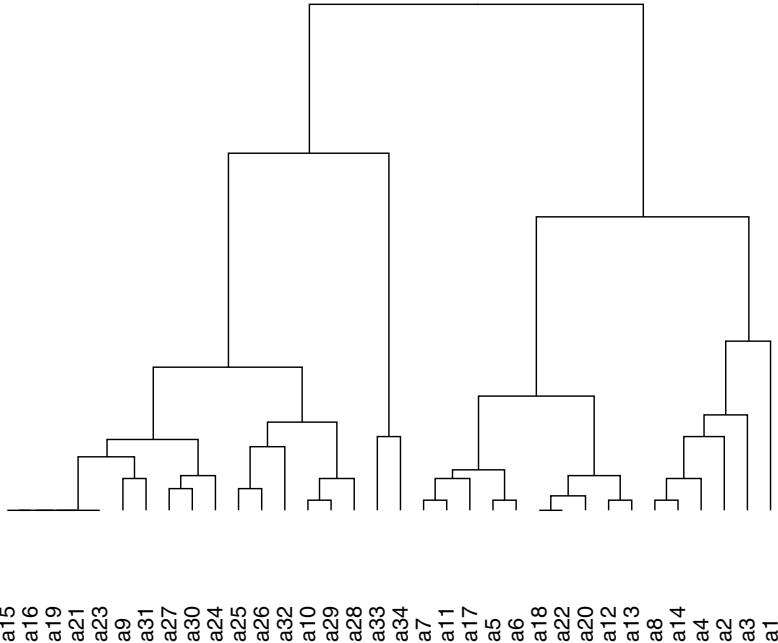


Fig. 4.7 Hierarchical clustering of the karate club network.

sparse graphs for (4.15) and (4.16), the cost is dominated by the operation of sorting these values.¹³

We note that not all hierarchical clustering methods employ vertex-by-vertex (dis)similarity measures. For example, an agglomerative approach proposed by Newman [294] seeks to greedily optimize the so-called *modularity* of a partition. Let $\mathcal{C} = \{C_1, \dots, C_K\}$ be a given candidate partition and define $f_{ij} = f_{ij}(\mathcal{C})$ to be the fraction of edges in the original network that connect vertices in C_i with vertices in C_j . The modularity of \mathcal{C} , as defined by Newman and Girvan [297], is the value

$$\text{mod}(\mathcal{C}) = \sum_{k=1}^K [f_{kk}(\mathcal{C}) - f_{kk}^*]^2 , \quad (4.17)$$

where f_{kk}^* is the expected value of f_{kk} under some model of random edge assignment. Most commonly, f_{kk}^* is defined to be $f_{k+}f_{+k}$, where f_{k+} and f_{+k} are the k -th row and column sums of \mathbf{f} , the $K \times K$ matrix¹⁴ formed by the entries f_{ij} . This choice corresponds to a model in which a graph is constructed to have the same degree distribution as G , but with edges otherwise placed at random, without respect to

¹³ For sorting n objects, good algorithms typically run in $O(n \log n)$ time.

¹⁴ Note that for undirected graphs this matrix will be symmetric, and hence $f_{k+} = f_{+k}$; for directed graphs, however, \mathbf{f} can be asymmetric.

the underlying partition elements dictated by \mathcal{C} . Large values of the modularity are therefore taken to suggest that \mathcal{C} captures nontrivial ‘group’ structure, beyond that expected to occur under the random assignment of edges.

Note that in this approach, because a single common quality measure $\text{mod}(\mathcal{C})$ is applied to all candidate partitions \mathcal{C} , as opposed to using an *ad hoc* propagation of (dis)similarities, we have a natural mechanism not only for generating the hierarchy but also for choosing an optimal partition within that hierarchy. Optimization of the modularity function (4.17) can be shown to be an *NP*-complete problem (Brandes et al. [61]). A greedy agglomerative algorithm is therefore commonly used for heuristic optimization (Newman [294], Clauset, Newman, and Moore [92]). It runs in $O(N_v^2 \log N_v)$ time generally, using sophisticated data structures, but nearly linear time on sparse networks with hierarchical structure, making it appealing for use in the context of certain large networks. See Clauset, Newman, and Moore [92] and Brandes et al. [61] for details. In practice, this algorithm has been found to frequently yield useful results. Other methods of a similar nature are described in the volume edited by Brandes and Erlebach [54, Ch. 8].

4.3.3.2 Spectral Partitioning

Another common approach to graph partitioning is to exploit results in spectral graph theory that associate the connectivity of a graph G with the eigen-analysis of certain matrices, such as the adjacency matrix or the Laplacian. Here we review two popular approaches to spectral partitioning. For general background on spectral graph theory, see the monograph by Chung [90].

The first approach is to conduct a spectral analysis of the adjacency matrix \mathbf{A} of a graph. That is, we begin by calculating the collection of N_v pairs $(\lambda_i, \mathbf{x}_i) \in \mathbb{R} \times \mathbb{R}^{N_v}$ for which $\mathbf{A}\mathbf{x}_i = \lambda_i \mathbf{x}_i$, where $\lambda_1 \leq \dots \leq \lambda_{N_v}$ denote the N_v (not necessarily distinct) eigenvalues of \mathbf{A} , ordered from smallest to largest, and $\mathbf{x}_1, \dots, \mathbf{x}_{N_v}$, their corresponding eigenvectors. Then, starting with the largest (absolute) eigenvalues, the entries of each eigenvector \mathbf{x}_i are sorted, and the vertices corresponding to particularly large positive or negative entries, in conjunction with their immediate neighbors, are declared to be a cluster. Note that in practice attention is usually restricted to just the K -th largest eigenvalue/vector pairs, for $K \sim \log N_v$.

The motivation behind this approach is the fact that, in the case of a graph G consisting of two d -regular graphs joined to each other by just a handful of vertices, the two largest eigenvalues will be roughly equal to d , and the remaining eigenvalues will be of only $O(d^{1/2})$ in magnitude. That is, there will be a gap in the spectrum of eigenvalues – a ‘spectral gap.’ In addition, the two eigenvectors corresponding to the largest two eigenvalues can be expected to have large positive entries on vertices of one sub-network and large negative entries on vertices of the other sub-network, with the two eigenvectors emphasizing the vertices in each sub-network in a complementary fashion. See Gkantsidis, Mihail, and Zegura [175], for example, for a summary of these and related results, in the context of a spectral analysis of Internet topology data.

Example 4.7 (Spectral Analysis of the Karate Club Network). Figure 4.8 shows the eigenvalues, and the first two eigenvectors, from a spectral analysis of the karate club network graph. We see that the first two eigenvalues are fairly distinct from the rest, indicating the possible presence of two sub-graphs. The first eigenvector appears to suggest a separation of the 1st and 34th actors, and some of their nearest neighbors, from the rest of the actors. The second eigenvector in turn provides evidence that these two actors, and certain of their neighbors, should themselves be separated. Actors 1 and 34 represent the director and administrator, respectively, of the original club. Together, these two eigenvectors go some ways towards separating the two clubs into which the actors eventually separated. \square

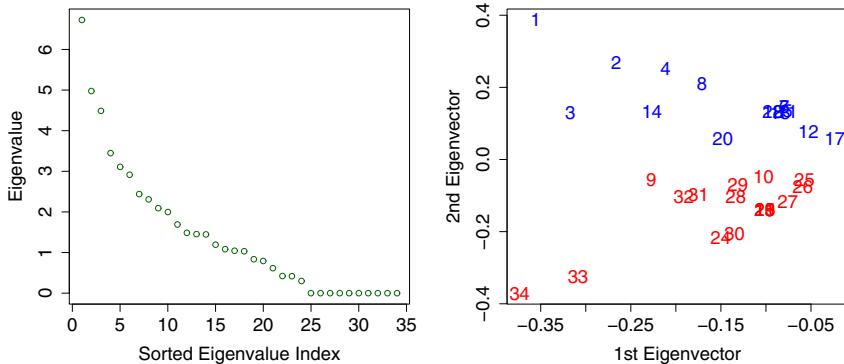


Fig. 4.8 Spectral analysis of the karate club network. Left: the (absolute) eigenvalues. Right: the first two eigenvectors, where points are labeled according to the number of their corresponding actors, and colored according to the sub-group of each actor, as displayed in Figure 1.2.

In reality, the method is known not to work as well when the sub-networks are far from regular and instead the network has a broad degree distribution, as is more common in practice. In addition, it should be noted that the partitions found through spectral analysis will tend to be ordered and separated by vertex degree, since the eigenvalues will mirror quite closely the underlying degree distribution. This has been observed empirically – it can be seen in the analysis of the karate club network above, for example – and has been proved rigorously for various network models like those we will see in Chapter 6. See Gkantsidis, Mihail, and Zegura [175] for some discussion, as well as references to a handful of formal results. Of course, this close relationship between partitioning and degree may not necessarily be seen as a problem. Sen, Kloczkowski, and Jernigan [346], for example, argue that such results are sensible in the context of protein complexes, based on biological evidence. Where it is suspected to be problematic, in yielding partitions that are unrepresentative of the true underlying ‘group’ structure, normalizing the adjacency matrix to

have unit row sums is a commonly proposed solution (e.g., Gkantsidis, Mihail, and Zegura [175]).

The other popular approach to spectral partitioning involves the iterative bisection of a network graph according to spectral properties of the graph Laplacian $\mathbf{L} = \mathbf{D} - \mathbf{A}$, where $\mathbf{D} = \text{diag}[(d_v)]$. A formal result in spectral graph theory states that a graph G will consist of K connected components if and only if $\lambda_1(\mathbf{L}) = \dots = \lambda_K(\mathbf{L}) = 0$ and $\lambda_{K+1}(\mathbf{L}) > 0$. See Godsil and Royle [176, Lemma 13.1.1]. The smallest eigenvalue of \mathbf{L} can be shown to be identically equal to zero, with corresponding eigenvector $\mathbf{x}_1 = (1, \dots, 1)^T$. Therefore, if we suspect a graph G to consist of ‘nearly’ $K = 2$ components, for example, and hence be amenable to bisection, we should expect $\lambda_2(\mathbf{L})$ to be close to zero.

Interestingly, this value λ_2 can be closely associated with a number of measures of graph connectivity and structure. See the survey by Mohar [281] or, more recently, the volume edited by Brandes and Erlebach [54, Ch. 14]. Among these, of particular relevance to the clustering problem is the *isoperimetric number* of a graph,

$$\phi(G) = \min_{S \subset V : |S| \leq N_v/2} \phi(S, \bar{S}) , \quad (4.18)$$

where $\phi(S, \bar{S}) = |E(S, \bar{S})|/|S|$ is called the *ratio* of the cut defined by (S, \bar{S}) . Recalling our discussion of vertex- and edge-cuts in Section 4.3.2, it is clear that the ratio is a natural quantity to minimize in seeking a good bisection of G . Unfortunately, this minimization problem is *NP-hard*. But $\phi(G)$ can be bounded as

$$\frac{\lambda_2}{2} \leq \phi(G) \leq \sqrt{\lambda_2(2d_{\max} - \lambda_2)} , \quad (4.19)$$

where d_{\max} is the maximum degree of G . Thus $\phi(G)$ will be small whenever λ_2 is small and vice versa. See Brandes and Erlebach [54, Ch. 14].

Fiedler [144], the first to associate λ_2 with the connectivity of a graph, suggested partitioning vertices by separating them according to the sign of their entries in the corresponding eigenvector \mathbf{x}_2 , producing a cut of the form

$$S = \{v \in V : \mathbf{x}_2(v) \geq 0\} \quad \text{and} \quad \bar{S} = \{v \in V : \mathbf{x}_2(v) < 0\} . \quad (4.20)$$

The vector \mathbf{x}_2 is hence often called the *Fiedler vector*, and the corresponding eigenvalue λ_2 , the *Fiedler value*. The latter is also known as the *algebraic connectivity* of the graph. Since it can be shown (see Mihail [276]) that

$$\phi(G) \leq \phi(S, \bar{S}) \leq \frac{\phi^2(G)}{d_{\max}} \leq \lambda_2 , \quad (4.21)$$

where $\phi(S, \bar{S})$ is the ratio of the cut in (4.20), spectral bisection can be viewed as a computationally efficient approximation to finding a best cut achieving $\phi(G)$.

Example 4.8 (Spectral Bisection of the Karate Club Network). Figure 4.9 shows a plot of \mathbf{x}_2 for the karate club network, in which it can be seen that spectral bisection

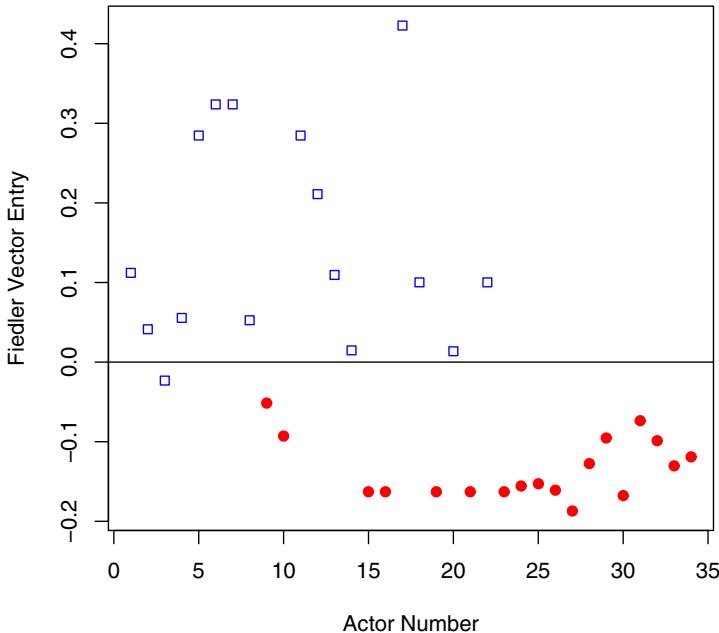


Fig. 4.9 Plot of the Fiedler vector, \mathbf{x}_2 , for the karate club network graph. The horizontal line at $y = 0$ indicates the clustering suggested by spectral bisection. Color and shape of the symbols for individual actors correspond to those of Figure 1.2.

produces a nearly perfect partition of the graph, classifying all but the third actor correctly. In practice, the bisection operation is often applied recursively, first to the graph G as a whole, and then within the subgraphs corresponding to each of the two partition elements identified, and so on. It can be shown rigorously that such an approach works well on bounded-degree planar graphs and certain mesh graphs (Spielman and Teng [366]), although in other contexts it cannot always be expected to give satisfactory results. Here, obviously, continuing to bisect the first two partition elements indicated by \mathbf{x}_2 will lead us farther away from the two-group structure known to exist in this network. \square

For both of the spectral partitioning methods described here, the computational overhead is in principle determined by the cost of computing the eigen-decomposition of an $N_v \times N_v$ matrix, which takes $O(N_v^3)$ time. However, realistically, only a small subset of extreme eigenvalues, and their corresponding eigenvectors, are needed in each case. Furthermore, the matrices \mathbf{A} and \mathbf{L} will typically be quite sparse in practice. Therefore, spectral analyses of this type, for sparse graphs as

large as tens of thousands of vertices and hundreds of thousands of edges, can be implemented efficiently using variants of the iterative Lanczos algorithm, which is specifically designed for such settings. In the case of spectral bisection, for example, the $O(N_v^3)$ time required by the nominal algorithm is reduced to $O(N_e/(\lambda_3 - \lambda_2))$, which will behave almost linearly in the number of edges N_e if the gap between the second and third smallest eigenvalues is sufficiently large (i.e., when the graph G really does nearly separate into two components). See Golub and van Loan [181] for additional details on algorithms of this type.

4.3.3.3 Additional Methods of Graph Partitioning

There are many other methods for partitioning graphs that have been proposed in various literatures – far too many to survey here. Instead, we restrict ourselves to describing just a handful of additional methods, most of which at least partly involve variations or extensions of principles underlying the two basic methods already described above.

For example, a small body of literature has emerged around a proposal of Girvan and Newman [174], involving the iterative removal of edges in a graph. The premise underlying the proposed method is that certain edges in a network graph may carry more of the ‘flow’ in that graph than others, and that it is these edges that likely serve as ‘bridges’ separating ‘cohesive’ subsets of vertices. Using the edge-betweenness centrality mentioned at the end of Section 4.2.2 to quantify the level of flow over an edge, the authors propose a divisive hierarchical algorithm that removes the edge of largest betweenness, recomputes the betweenness of all remaining edges, and continues iteratively until no more edges remain.

This method seems to work quite well in practice, but can be somewhat costly from a computational perspective, requiring $O(N_v^3)$ time. There have been several suggestions for speeding up the basic algorithm (e.g., see Newman [289] for discussion and references), including a parallel implementation proposed by Yang and Lonardi [409] that has yielded nearly linear empirical run times. Of course, another approach to the cost problem – and of interest in its own right – is the substitution of other measures than edge-betweenness to prioritize the removal of edges. Measures based on concepts of resistance and random walks in networks are examined in Newman and Girvan [297], for example, although both of these choices are in fact more computationally costly than edge-betweenness.

Newman [294] has observed that the hierarchical clustering algorithm described earlier in this section, based on the modularity in (4.17), seems to yield results similar to those produced by such edge-removal algorithms. The use of modularity as a general measure of quality for proposed clusterings (i.e., distinct from its use with the agglomerative algorithm proposed in Newman [294]) has itself received some attention in the graph partitioning community, particularly among those interested in the detection of communities in social networks. With respect to the computation of modularity, there have been suggestions for approximate optimization of modularity using spectral methods. For example, Newman [291] proposes a method whose

technical development parallels that of the spectral bisection method quite closely, but with a matrix related to modularity playing the role of the Laplacian L . Similarly, White and Smyth [402] have proposed a method of optimization based on spectral relaxation.

Another notable use of concepts from spectral analysis may be found in the method of Gibson, Kleinberg, and Raghavan [172], who characterize community structure among pages in the World Wide Web. There the eigenvectors of the pair of matrices $\mathbf{M}_{hub} = \mathbf{A}\mathbf{A}^T$ and $\mathbf{M}_{auth} = \mathbf{A}^T\mathbf{A}$ are used to assign vertices membership in clusters, where A is the adjacency matrix of a directed graph G . The matrices \mathbf{M}_{auth} and \mathbf{M}_{hub} were seen to underlie the hubs-and-authorities notion of centrality mentioned in Section 4.2.2.2.

Also no doubt deserving mention here is the general body of techniques referred to as *block modeling*. First proposed by White, Boorman, and Breiger [401], techniques of this sort seek to model the relationships among ‘positions’ in a network, and often bear a strong algorithmic resemblance to clustering methods. Because the motivation for and details of work in this area has tended to be strongly intertwined with various notions of social structure, to do them justice is beyond the scope of this book. We instead refer the reader to the book by Doreian, Batagelj, and Ferligoj [122]. See also Scott [344], Wasserman and Faust [393], and the volume edited by Brandes and Erlebach [54, Chs. 9 and 10], who in addition provide information on the closely related topics of positional and role analysis.

Finally, as a side note, it is perhaps worth pointing out that it is not uncommon to see methods designed for cluster analysis in general datasets directly applied to the context of graphs. Typically the graph is first embedded into a Euclidean space, using a tool like multi-dimensional scaling (as underlies the Kamada-Kawai layout algorithm discussed in Section 3.4.2), and then a standard clustering algorithm, like k -means clustering, is applied to the ‘data’ that are the embedded vertices, using the corresponding Euclidean distances.

4.3.4 Assortativity and Mixing

As has been mentioned a number of times so far in this section on the characterization of network cohesion, it is generally expected, where cohesive subsets of vertices are present in a network graph, that underlying these subsets there is some commonality in certain relevant characteristics (or attributes) of the vertices. For example, proteins found to cluster in a graph of protein interactions often participate in similar biological processes; similarly, actors found to cluster in a social network may share similar relevant interests. Graph partitioning methods are useful precisely because these characteristics will often be unobserved (i.e., they will be latent). However, conversely, given certain measured vertex characteristics, it can be useful to quantify the extent to which vertices may be partitioned according to them.

Selective linking among vertices, according to a certain characteristic(s), is termed *assortative mixing* in the social network literature, and measures that quantify the extent of assortative mixing in a given network have been referred to as *assortativity coefficients*. The assortativity coefficients we describe here are attributed to Newman [293] and are essentially variations on the concept of correlation coefficients.¹⁵

To begin, we note that the vertex characteristics involved could be categorical, ordinal, or continuous. Consider the categorical case, and suppose that each vertex in a graph G can be labeled according to one of M categories. Let f_{ij} be the fraction of edges in G that join a vertex in the i -th category with a vertex in the j -th category; denote the i -th marginal row and column sums of the resulting matrix \mathbf{f} by f_{i+} and f_{+i} , respectively. Note that these quantities are defined in complete analogy to the same quantities that underlie our definition of modularity (4.17). We then define the assortativity coefficient r_a to be

$$r_a = \frac{\sum_i f_{ii} - \sum_i f_{i+} f_{+i}}{1 - \sum_i f_{i+} f_{+i}} . \quad (4.22)$$

The value r_a is equal to zero when the mixing in the graph is no different from that obtained through a random assignment of edges that preserves the marginal degree distribution (see the related discussion surrounding (4.17) above). Similarly, it is equal to one when there is perfect assortative mixing (i.e., when edges only connect vertices of the same category). When the mixing is perfectly disassortative, in the sense that every edge in the graph connects vertices of two different categories, then the coefficient in (4.22) takes a minimum value of

$$r_a^{\min} = -\frac{\sum_i f_{i+} f_{+i}}{1 - \sum_i f_{i+} f_{+i}} . \quad (4.23)$$

Hence, the value r_a lies between -1 and 1 , but does not achieve -1 in the case of perfect disassortative mixing. Newman [293] argues that this behavior may be interpreted as reflecting that disassortative networks can range less ‘far’ from randomness than assortative networks.

Example 4.9 (Assortativity in the Abilene Network). Consider the Abilene network in Figure 1.1, described in Section 1.2.1. A quick glance at the network graph in this figure is sufficient to strongly suggest the presence of disassortative mixing. The Abilene network, like the Internet as a whole, is dominated by a clear hierarchical structure. There are 11 ‘core nodes’ in the network. The core nodes are linked both to each other and to additional nodes of certain types. However, most of these latter nodes serve only to join Abilene with other Internet networks. And neither these nodes nor the other networks tend to connect with each other directly. Table 4.1

¹⁵ Newman’s ideas in turn extend a popular assortativity coefficient originally proposed by Gupta, Anderson, and May [189]. However, the latter is somewhat problematic in that it can allow arbitrarily small subsets of vertices to have a disproportionately large effect. See Newman [293, App. A] for discussion.

shows the actual numbers of category-to-category edges in this network, for the six categories of nodes defined in the figure. The f_{ij} are obtained by scaling these values by the total 675. The assortativity coefficient is found to be $r_a = -0.3162$, which is not far from the minimum value $r_a^{min} = -0.3461$, thus strongly supporting our suspicions of disassortative mixing. \square

	Core	Exchange	Peer	Connector	Participant	Conn./Part.
Core	14	6	5	17	0	16
Exchange	6	1	46	2	0	0
Peer	5	46	0	0	0	1
Connector	17	2	0	0	203	0
Participant	0	0	0	203	0	34
Conn./Part.	16	0	1	34	34	0

Table 4.1 Counts of edges between vertices of six different types in the Abilene network depicted in Figure 1.1.

The extension of these ideas to ordinal or continuous characteristics is straightforward. An example, in the context of social networks, of a vertex characteristic of this type for which there might be assortative mixing, is the age of the actor corresponding to each vertex. See Newman [293, Fig 1], for example. Note that age can be treated as either continuous or, if rounded, ordinal. For a given characteristic of interest, denote by (x_e, y_e) the values of that characteristic for the vertices joined by an edge $e \in E$. Then a natural candidate for quantifying the assortativity in this characteristic is just the Pearson correlation coefficient of the pairs (x_e, y_e) ,

$$r = \frac{\sum_{x,y} xy(f_{xy} - f_{x+}f_{+y})}{\sigma_x \sigma_y} . \quad (4.24)$$

Here the summation is over all unique observed pairs (x, y) , with f_{xy} , f_{x+} , and f_{+y} defined in analogy to the categorical case, and σ_x and σ_y are the standard deviations corresponding to the distribution of frequencies $\{f_{x+}\}$ and $\{f_{+y}\}$, respectively. In the case where the characteristic of interest is vertex degree, note that we simply recover the degree correlation described in Section 4.2.1.

On a related note, we mention that the similar structure underlying the definition of the modularity $\text{mod}(\mathcal{C})$ of a partitioning \mathcal{C} in (4.17), on the one hand, and the assortativity coefficients r_a and r in (4.22) and (4.24), on the other, allow for interesting connections to be made between modularity and assortativity, as discussed in Newman [291, Sec. VIII.A].

4.4 Case Study: Analysis of an Epileptic Seizure

Epilepsy is a chronic neurological condition and the world's most common serious brain disorder. It is characterized by seizures that are associated with abnormal activity among neurons in the brain. Epilepsy is commonly controlled through medication, but in those cases where medication no longer prevents seizures, surgical removal of the affected zone of the brain may be necessary. Perhaps somewhat surprisingly, this surgical procedure has not changed substantially since its inception in the late 19th century, although improved imaging and analysis techniques – including network analysis techniques – are helping to refine it. In order to illustrate the integrated usage of some of the concepts and tools introduced in this chapter, we present a network-oriented analysis of epileptic seizure data in humans, following Kramer, Kolaczyk, and Kirsch [239].

Figure 4.10 shows the nodes in an 8×8 electrode grid that was implanted in the cortical surface of the brain of a patient with epilepsy. Accompanying this grid were two strips (not shown) of six electrodes each, implanted deeper into the brain. Together, these implants allow for the collection of electrocorticogram (ECoG) data, which consist of time series of voltage levels in the vicinity of each of the 76 electrodes. The voltage levels, in turn, are indicative of the levels of local brain activity. An example of one such time series is shown in the figure. Also indicated are two separate portions of this series, corresponding to 10-second intervals identified as being prior to and immediately after the start of a seizure in the patient, referred to as the ‘preictal’ and ‘ictal’ periods, respectively. Our interest will be in comparing the activity of the brain during these two periods.

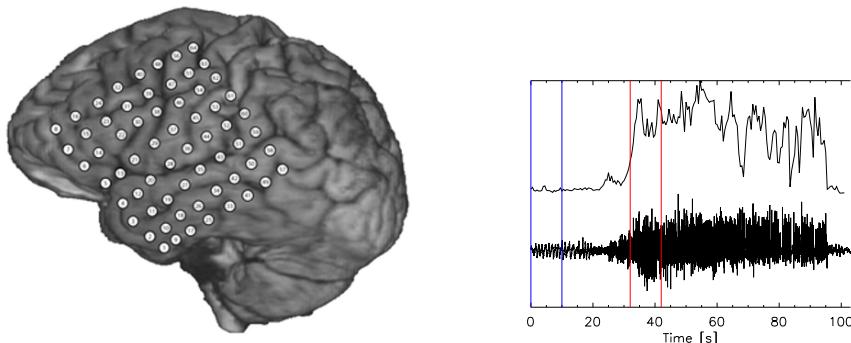


Fig. 4.10 Left: Three-dimensional reconstruction of the brain of an epilepsy patient, with ECoG implant grid super-imposed. Right: Example of an ECoG time series at one electrode, both for one seizure (bottom) and after smoothing (i.e., bandpass filtering) and averaging over eight such seizures (top). Preictal (blue) and ictal (red) periods are indicated between pairs of parallel lines.

Network graphs are a natural way to represent the interactive activity – or coupling – among different regions of the brain. The review of Boccaletti et al. [41]

includes a short survey of various examples of such usage of network graphs, at different scales and based on different types of measurements. Here, network graphs were generated to summarize the cortical-level coupling among the different measured regions of the brain, for both preictal and ictal periods. Specifically, within each period, for each pair among the 76 electrodes, a measure of correlation was computed between the two corresponding 10-second time series, a threshold was applied, and an edge was assigned to the vertices representing these electrodes if their correlation exceeded that threshold. This process was carried out for data on eight different seizures in this patient, measured over five consecutive days, essentially following Kramer, Kolaczyk, and Kirsch [239].

An example of the resulting network graphs, corresponding to one seizure, is shown in Figure 4.11. A mere glance at the two graphs in the figure is enough to see that they strongly suggest that the brain is in two very different states prior to and during the seizure. In particular, there appears to be a thinning of edges throughout the network, indicating a reduced level of coupling between different cortical regions in the brain. This behavior is especially apparent in the bottom corner of the grid, closest to the two strips of electrodes, which in turn were located close to where the seizure was suspected to emanate.

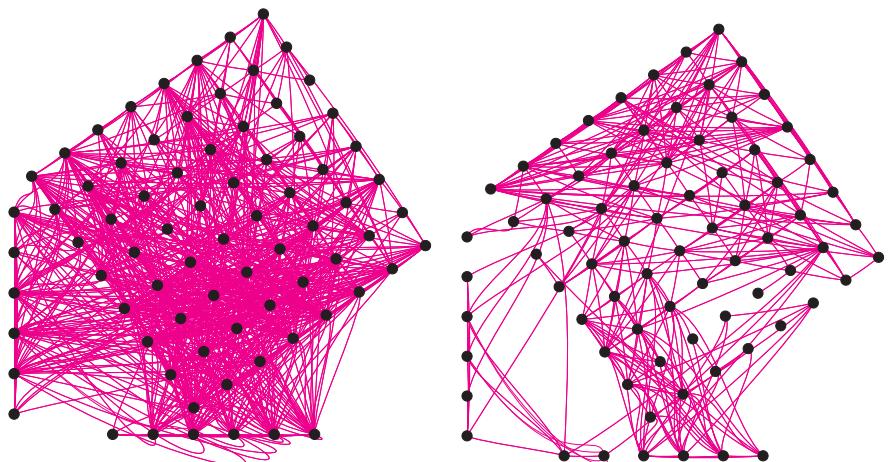


Fig. 4.11 Network representations of cortical-level coupling between brain regions about each electrode, during preictal (left) and ictal (right) periods.

A number of the summaries of network characteristics described in this chapter are useful for quantifying the apparent preictal-to-ictal changes. For example, Figure 4.12 shows visual representations of degree, closeness and betweenness centrality, and clustering coefficient values for each of the 76 vertices in our network. Preictal and ictal periods, as well as their difference, are shown separately, averaged over the eight seizures at each vertex. We can see, looking at the plots of vertex degree, that our initial impression of a global decrease in coupling, with a partic-

ular emphasis in the bottom corner, is confirmed. The plots of closeness centrality show an accompanying global increase in distance (i.e., decreased closeness) between vertices, again with rather greater changes in the bottom corner of the grid. Betweenness centrality is seen to have changed as well, but in a much more irregular manner. There are both increases and decreases in vertex betweenness, with some vertices having experienced particularly large increases. Note, however, that most of these vertices with large increases in betweenness experienced relatively little change in degree. Finally, both positive and negative changes are also observed in the vertex clustering coefficients, though the relative magnitudes are a bit smaller than for the other summary measures. The more pronounced changes are decreases in local clustering, and many of these seem to occur close to regions where betweenness changed the most.

Together these results suggest that for areas where local density decreases upon seizure onset, and for which a region nearby remains coupled at roughly pre-seizure levels, there is an increased burden in the form of more paths ‘squeezing’ through. Note that since there are eight seizures from the same patient at our disposal, treating these seizures as replicates allows for formal assessment of statistical significance of the changes noted above.¹⁶ For example, changes in global quantities like average degree, average path length, closeness and betweenness centralization, and average clustering coefficient all are found statistically significant at the 0.05 level using paired *t*-tests on seven degrees of freedom. Similarly, the use of such tests on a vertex-by-vertex basis, with control for multiple testing, allows us to pinpoint the locations of the most significant contributions to these changes. For instance, at a false discovery rate (FDR) of $q = 0.05$, the 10 vertices marked with stars in Figure 4.12 are identified as locations of significant change in betweenness. Kramer, Kolaczyk, and Kirsch [239] suggest that such locations may potentially serve to more precisely guide surgical intervention.

4.5 Characterizing Dynamic Network Graphs

We address briefly here the problem of characterizing dynamic network graphs, that is, of characterizing a collection $\{G^{(t)}\}$ of network graphs indexed over times t in some set \mathbb{T} , as introduced in Section 3.6. The development of a comprehensive body of tools for such characterization, and of a corresponding understanding of their properties and usage, currently lags far behind that for static graphs G that has been the focus of most of this chapter. This lacking is probably due in part to insufficiently pressing need, in that traditionally it has been notably more difficult in most network studies to obtain time-indexed network data that are rich in

¹⁶ Up until now, we have mentioned little in the way of how to assess statistical significance in the summary measures introduced in this chapter, because when given only a single network graph (or pair of graphs, in the context of our epilepsy case study), such assessments necessarily must rely on network graph models, such as we will see in Chapter 6. Here, however, the availability of repeated observations allows us to assess significance based on sample variability.

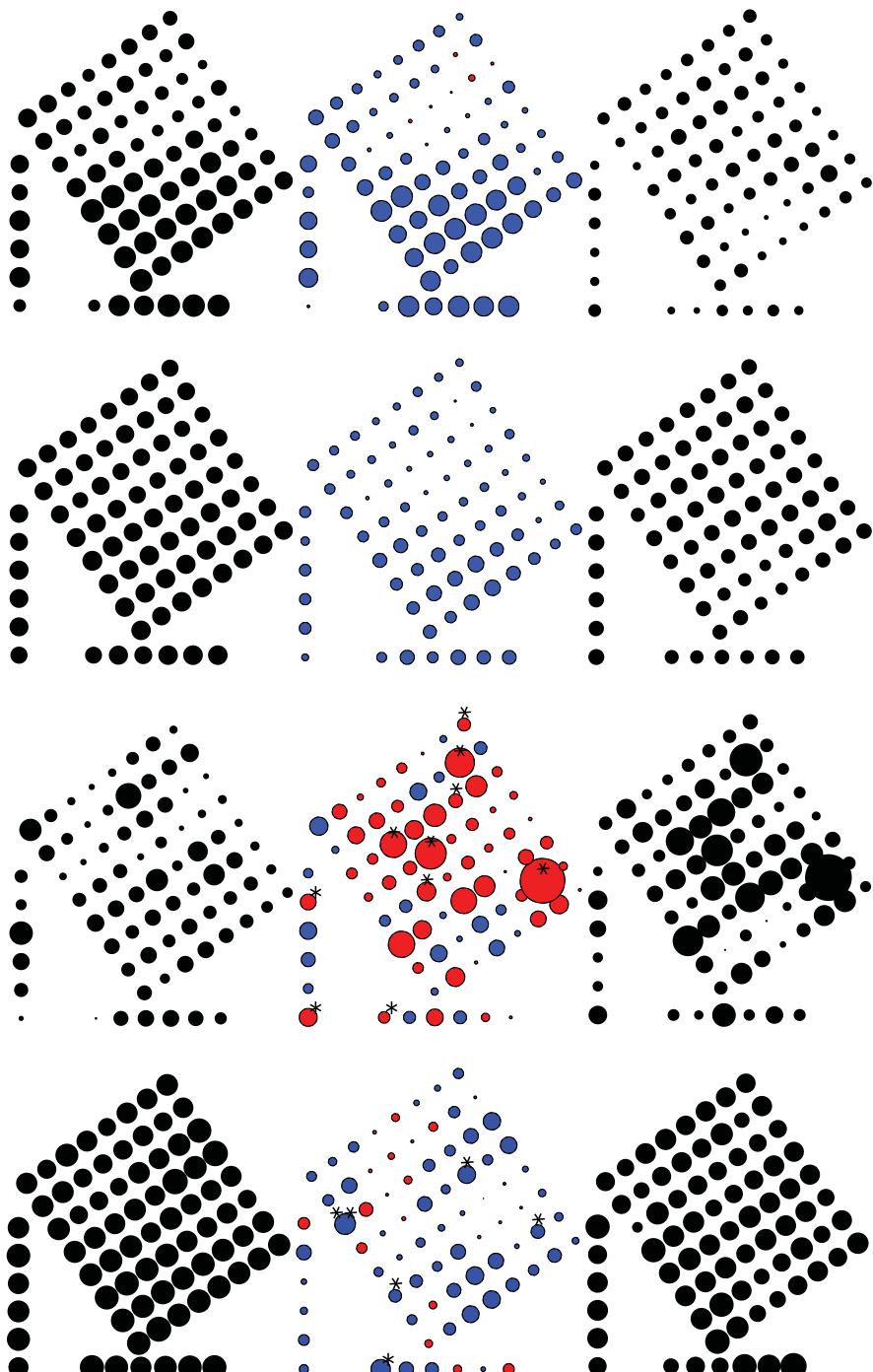


Fig. 4.12 Visual summaries of vertex degree, closeness centrality, betweenness centrality, and clustering coefficient (Rows 1 – 4), averaged over eight seizures. Columns correspond to preictal (left) and ictal (right) periods, and to their difference (center). Vertices are drawn with radius proportional to the square root of the relative magnitude, which allows for meaningful visual comparison of their areas. In the middle column, blue indicates positive differences, and red, negative.

both quality and quantity. The situation, however, is beginning to change, particularly where Internet-based measurement and archival resources can be leveraged, as in Leskovec, Kleinberg, and Faloutsos [255] or Kossinets and Watts [238]. Also contributing to the gap between static and dynamic methods of characterization is possibly the absence of a mature graph-theoretic infrastructure. For example, an extension to dynamic networks of Menger's fundamental theorem (relating paths and cut-sets) was only proposed relatively recently, by Kempe, Kleinberg, and Kumar [224]. Finally, another important factor is the sheer combinatorial explosion in problem variants, once the factor of 'time' is to be incorporated. See Moody, McFarland, and Bender-deMoll [285], for example, for a detailed discussion of how and why such variants arise, in the context of social networks.

There are various representations of a dynamic network with which we might work, roughly paralleling those found in traditional statistical time series analysis. For example, we might study cumulative versions of the network, in the sense that a vertex or edge present in $G^{(t)}$ is present in all $G^{(t')}$, for $t' > t$ (e.g., Leskovec, Kleinberg, and Faloutsos [255]). Or rather we might study discrete 'snapshots,' where each $G^{(t)}$ features only those vertices and edges present in a certain interval of time around t . Depending on the temporal resolution of the measurements in such cases, some authors have found it useful to work with a smoothed sequence of graphs across time, based on, for example, moving windows (e.g., Kossinets and Watts [238]) and moving averages (e.g., Hill, Agarwal, Bell, and Volinksy [198]). In some cases, it may be possible to obtain network information continuous in time, which can allow for analyses not possible with measurements at other temporal resolutions (e.g., Kumar, Raghavan, Novak, and Tomkins [242], Priebe, Conroy, Marchette, and Park [315]).

The potential effects of different representations of a dynamic network on the analysis of its structural characteristics is presumably an important issue, but one that has received little formal attention to date. In studies analyzing dynamic network graphs, two issues that have received fairly consistent attention are (i) the evolution over time of standard static global graph properties, such as degree characteristics, diameter, and clustering behavior, and (ii) the dynamics of the underlying formation and dissolution of edges (or arcs, in the directed case) among vertices.

With respect to the evolution of standard global properties, an important question is whether various properties are stable in time. In Leskovec, Kleinberg, and Faloutsos [255], for example, it is demonstrated that – contrary to current wisdom – the network graphs for a number of large-scale systems (e.g., citations and co-authorship in various literatures, and autonomous systems in the Internet) experience both an increase in density and a shrinking in diameter over time. On the other hand, Kossinets and Watts [238] found that many characteristics of a large email-based social network of university students and faculty remained fairly stable over time, outside of expected periods of fluctuation, like summer and winter recess.

Regarding the dynamics of the network edges themselves, this is a topic that a subset of researchers in social network analysis have concerned themselves with for some time, particularly with respect to the role of potential social mechanisms. See the edited volume by Doreian and Stokman [121], for example, and the ref-

erences therein. However, most studies in that area have revolved around datasets for rather small social groups, which makes rigorous empirical assessment of proposed mechanisms and models difficult. Datasets across a range of scales – both in terms of network size and in terms of longitudinal time points – are needed to help gain additional insight into potentially non-trivial dynamic process structures. Furthermore, information beyond just vertex and edge lists can be expected to be necessary, such as that provided by longitudinal measurements on appropriate vertex characteristics (e.g., group affiliations of social actors). The email-based social network in Kossinets and Watts [238], for example, involved roughly 14.5 million exchanges within a university population of about 43,500 people, over a period of 355 days, as well as various personal attributes on each individual. Such data allow for use of, for instance, formal methods of statistical survival analysis in assessing the effect of vertex characteristics on the rates at which triadic closure (i.e., triads turning into triangles) occurs.

When continuous-time data are available, it is possible to ask questions about changes in graphs on very short time scales. For example, Kumar, Raghavan, Novak, and Tomkins [242] present an analysis of large-scale blogging activity in the World Wide Web in which such continuous-time data allowed for the discovery of short, dense ‘bursty’ periods of the creation of edges within communities. Similarly, Priebe, Conroy, Marchette, and Park [315] introduce methods for the detection of change-point anomalies in graph structure, illustrated in the context of the network of email correspondence surrounding the Enron scandal.

4.6 Additional Related Topics and Reading

The description and summary of structural characteristics of an appropriate network graph is frequently an important and necessary first step in the analysis of network data, and often it serves as the primary part of such analyses. The array of tools and techniques proposed for this purpose is quite large and still growing. Here in this chapter we have contented ourselves simply with presenting material on topics arguably at the core of this area.

From this vantage, the interested reader has numerous additional avenues that might be explored. The volume edited by Brandes and Erlebach [54], cited frequently throughout this chapter, provides a much more detailed – yet highly readable – presentation of most of the topics in this chapter, as well a number of others not covered here, from the perspective of computer science and applied graph theory. In the statistical physics literature, there have been a number of comprehensive review articles written from the perspective of that discipline, including that of Newman [296] and, more recently, that of Boccaletti et al. [41]. Both of these articles cover materials related both to this chapter and, particularly in the case of Boccaletti et al. [41], to later chapters on network modeling (Chapter 6) and network processes (Chapter 8). In the literature on social networks, two canonical references are the small volume by Scott [344] and the much larger volume by Wasserman and

Faust [393]. Lastly, we point out that there are nontrivial points of tangency between work on methods of structural analysis in network graphs and work on graph-based approaches to high-dimensional data analysis (e.g., manifold learning). As both areas of research are quite active currently, this is likely to be a fertile interaction to watch.¹⁷

Exercises

4.1. Consider the problem of estimating the exponent α for a distribution with density proportional to $x^{-\alpha}$.

- a. A random variable X is said to follow a Pareto distribution if it has a density of the form

$$f(x) = \frac{\alpha x_0^\alpha}{x^{\alpha+1}}, \quad (4.25)$$

for $x > x_0$, where $x_0, \alpha > 0$. Suppose that we observe an independent and identically distributed sample x_1, \dots, x_n according to (4.25), and let x_0 be known. Show that under the Pareto model the log-likelihood

$$\ell(\alpha) = \sum_{i=1}^n \log \alpha + \alpha \log x_0 - (\alpha + 1) \log x_i$$

is maximized by the estimator in (4.4). That is, show that the Hill estimator is the maximum likelihood estimator for α under this model.

- b. Use the computer to generate independent and identically distributed samples x_1, \dots, x_n from a Pareto distribution,¹⁸ for various choices of α and n . For each choice of these values, estimate α using the Hill estimator, and compare the performance of this estimator to that of the naive regression estimators motivated by the relation in (4.2) and the logarithm of the relation in (4.3).

4.2. Let G be an undirected, connected graph, with weights $\{w_e\}_{e \in E}$. Recall the definition of the betweenness centrality $c_B(v)$ in (4.7). The key result underlying the algorithm of Brandes [51] for computing betweenness centrality is a recursive expression for the partial sums

$$\delta_{s+}(v) = \sum_{t \in V} \delta_{st}(v) ,$$

where $\delta_{st}(v) = \sigma(s, t | v) / \sigma(s, t)$ is the *pair-dependency* of $s, t \in V$.

¹⁷ See, for example, <http://www.ipam.ucla.edu/programs/gss2005/>.

¹⁸ Samples may be drawn from a Pareto distribution, with parameters $x_0, \alpha > 0$ by using the inverse transform method (e.g., see Liu [264, Ch. 2.1]). Specifically, having generated (pseudo)random numbers u_1, \dots, u_n uniformly on $(0, 1)$, the values $x_i = x_0 / u_i^{1/\alpha}$ will follow the desired Pareto distribution.

Define the set of *predecessors* of v on shortest paths from s as

$$P_s(v) = \{u \in V : \{u, v\} \in E, \text{dist}(s, v) = \text{dist}(s, u) + w(u, v)\} ,$$

where dist is the geodesic distance and $w(u, v)$ is the weight on the edge incident to vertices $u, v \in V$.

- a.** Suppose that there is exactly one shortest path from $s \in V$ to each $t \in V$. Show that

$$\delta_{s+}(v) = \sum_{z:v \in P_s(z)} (1 + \delta_{s+}(z)) .$$

- b.** Show that in the general case,

$$\delta_{s+}(v) = \sum_{z:v \in P_s(z)} \frac{\sigma(s, v)}{\sigma(s, z)} \cdot (1 + \delta_{s+}(z)) .$$

4.3. Recall the two definitions of clustering coefficient cl and cl_T in equations (4.11) and (4.13), respectively.

- a.** Verify the relation between cl and cl_T in equation (4.12).
- b.** Construct an example of a class of graphs for which cl can be made arbitrarily close to one while cl_T becomes arbitrarily close to zero. (Hint: Consider graphs composed of a set of ‘nested’ triangles.)

4.4. Recall the use of the graph Laplacian \mathbf{L} for spectral graph clustering, described in Section 4.3.3.

- a.** Suppose that G is composed of $K = 2$ separate connected components, say G_1 and G_2 . Then \mathbf{L} will be a block-diagonal matrix with two blocks. Recall that the smallest eigenvalue λ_1 is equal to zero, and that it corresponds to the eigenvector $\mathbf{x}_1 = (1, \dots, 1)^T$. Show that in addition the next two smallest eigenvalues will also be zero, with degenerate eigenvectors \mathbf{x}_k defined so that $x_{k,i} = 1$ if vertex i is in G_k , and zero otherwise.
- b.** Generate graphs G that deviate from the scenario of strictly separated components G_1 and G_2 , underlying part (a), to increasingly greater extents. For example, you might let G_1 and G_2 be two identical cliques of size n , and add an increasingly larger number of edges between randomly selected pairs of vertices, one in G_1 and the other in G_2 . Study the effects of these changes on the behavior of the Fiedler vector and the corresponding effect on your ability to correctly identify the two clustered subgraphs G_1 and G_2 .

4.5. Select a network graph of interest to you. In the spirit of the case study in Section 4.4, identify a handful of questions, relating to the underlying complex system from which the network derives, for which the analysis of certain network graph characteristics may be relevant. Using software of your choice, conduct such an analysis and report on your findings.

Chapter 5

Sampling and Estimation in Network Graphs

Frequently it is the case that relational information is observed on only a portion of a complex system being studied, and the network resulting from such measurements may be thought of as a sample from a larger underlying network. If the goal is to use the sampled network data to infer properties of the underling network, this task may be approached using principles of statistical sampling theory. However, sampling in a network context introduces various potential complications. In this chapter we formalize the problem of sampling and estimation in network graphs, describe a handful of common network sampling designs, and develop estimators of a number of quantities of interest.

5.1 Introduction

The material in the previous two chapters has been developed under the implicit assumption that the network graph being mapped and explored is itself the primary object of interest. However, as was mentioned in Section 3.2, this perspective is one of ‘measurement as enumeration’ and, in particular, ignores the possible effects of sampling.

Although there are many cases in which a network graph is essentially observed in its entirety, arguably sampled graphs may be more the rule than the exception in practice. For example, in social networks, while it may be possible to fully construct the friendship network among school children in a small classroom, it may be cumbersome to attempt to do so for all employees in a large corporation. Similarly, as we saw in the case study on mapping the Internet, in Section 3.5.2, sampling is a necessary part of Internet topology studies.

Formally, assume there is a system under study that may be represented by a network graph G , which we will call the *population graph*. Suppose in addition, however, that instead of having all of G available to us, we take measurements that effectively yield a sample of vertices and edges, which we compile into a graph $G^* = (V^*, E^*)$. We will refer to G^* as a *sampled graph*. The sampled graph G^* will

often be a subgraph of G (i.e., $V^* \subseteq V$ and $E^* \subseteq E$), although this will not always be the case. For example, in situations where there is error in assessing the existence of vertices or edges, there is chance of spurious observations.

Now suppose that there is a particular characteristic of G , denoted $\eta(G)$, that is of interest. For example, $\eta(G)$ might be a structural characteristic of G , such as the number of edges N_e , the average degree, or the distribution of vertex betweenness centrality scores. Similarly, it might be a summary of some quantity with which the nodes or edges in G are decorated, such as the proportion of men with more female than male friends in a social network. If G is sampled, then typically it will be impossible to recover the exact value of $\eta(G)$ from only the partial information supplied by G^* . The question thus arises as to whether we may still obtain a useful estimate of $\eta(G)$, say $\hat{\eta}$, from G^* .

Intuitively, it is attractive to think that we might simply estimate $\eta(G)$ by $\hat{\eta} = \eta(G^*)$. That is, we might use a ‘plug-in’ method and estimate the characteristic of interest by the value of that characteristic observed in the sampled graph G^* . This approach is in fact implicitly what is used in any network study that asserts that properties of an observed network graph are indicative of those same properties for the graph of the network from which the data were sampled. And certainly many familiar estimators in general practice are plug-in estimators of this type. Sample means, standard deviations, and quantiles, for example, are all both natural and valid estimates of their population equivalents under standard assumptions of a sample with independent and identically distributed observations.

Unfortunately, in estimating graph characteristics from sample graphs, this line of reasoning can often go awry, as the following example illustrates.

Example 5.1 (Estimating Average Degree). Suppose that the characteristic of interest is the average degree of a graph G ,

$$\eta(G) = (1/N_v) \sum_{i \in V} d_i . \quad (5.1)$$

Let our sample graph G^* be based on the n vertices $V^* = \{i_1, \dots, i_n\}$, and denote its observed degree sequence by $\{d_i^*\}_{i \in V^*}$. The plug-in estimator of $\eta(G)$ in (5.1) is just the average of the observed degree sequence,

$$\hat{\eta} = \eta(G^*) = (1/n) \sum_{i \in V^*} d_i^* . \quad (5.2)$$

To evaluate this estimator, we consider two sampling designs by which G^* might be obtained. In both cases, we begin with a simple random sample without replacement¹ of n vertices $V^* = \{i_1, \dots, i_n\}$. Then, in Design 1, for each vertex $i \in V^*$, we observe all edges $\{i, j\} \in E$ involving i ; each such edge becomes an element of E^* . On the other hand, in Design 2, we only observe, for each pair $i, j \in V^*$, whether or not $\{i, j\} \in E$; if it is, that edge becomes an element of E^* . Hence, both designs

¹ That is, elements are drawn sequentially and uniformly at random from a population in such a way as to avoid sampling any element more than once. See Example 5.2.

consist of two steps – first sampling a set of vertices V^* and then observing a set of edges E^* – but differ in the manner in which the edges are observed. In either case, the final sampled graph is simply $G^* = (V^*, E^*)$.

Figure 5.1 shows histograms showing the results of calculating the sample average $\hat{\eta} = \eta(G^*)$ under each of these two sampling designs, in the case where the true graph G is taken to be the network of protein interactions in yeast introduced in Section 4.2.1.1. Recall that G consists of $N_v = 5,151$ vertices and $N_e = 31,201$ edges; it has an average degree of $\eta(G) = 12.115$. A random sample of $n = 1,500$ vertices was drawn, and edges were sampled per the specifications of Designs 1 and 2, respectively. This process was repeated for 10,000 trials.

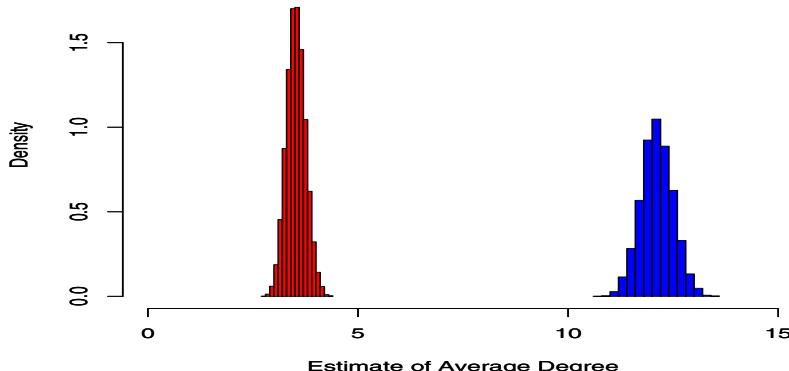


Fig. 5.1 Histograms of estimated average degree in the yeast protein interaction network, based on sampling under Design 1 (blue) and Design 2 (red), over 10,000 trials.

Looking at the figure, it is easy to see that under Design 1 the plug-in estimator $\hat{\eta}$ is quite accurate, with a mean of 12.117 and a standard error of 0.3797. However, under Design 2 it has a substantial bias, with a mean of only 3.528, although its standard deviation is a bit smaller, at 0.2260. The difference in the performance of the estimators in this example is, of course, a function of the difference in the observed degrees $\{d_i^*\}_{i \in V^*}$, induced by the difference in which edges are observed. In the case of Design 1, we observe the actual degree of a vertex $i \in V^*$ i.e., $d_i^* = d_i$. But in the case of Design 2, we observe a degree that typically under-shoots the actual degree, by a factor of roughly n/N_v i.e., $d_i^* \approx nd_i/N_v$. Hence, despite the fact that in both cases vertices are drawn through simple random sampling, in the latter design our estimate of the average degree is an under-estimate. \square

While the sampling designs in Example 5.1 are relatively simple, they are by no means uncommon, and rather than being an exceptional case, the behavior witnessed above has been seen to occur under a variety of sampling designs for various

choices of characteristic $\eta(G)$ (e.g., Lee, Kim, and Jeong [253]). In the specific context of protein interaction networks, Han et al. [192] have documented analogous results using sampling designs more representative of experimental protocols in proteomics.

We shall see in this chapter that, when viewed from the perspective of classical statistical sampling theory, such problems with network graph estimation come as little surprise and, indeed, can be compensated for in many cases through appropriately designed estimation strategies. In Example 5.1, for instance, adjusting the plug-in estimator $\hat{\eta}$ upward by a factor of $N_v/n = 5,151/1,500 \approx 3.434$ yields an estimator with mean 12.115. In this case the correction is easy, but in general, deriving such correction strategies can be more challenging, depending on the manner in which the topology of the graph G , the characteristics of $\eta(\cdot)$, and the nature of the sampling design interact.

5.2 Background on Statistical Sampling Theory

Statistical inference in classical sampling theory is based upon a slightly different paradigm than, say, the likelihood-based or Bayesian frameworks described in Chapter 2. In particular, the standard setup assumes that observations are made without measurement error, and hence that the only source of randomness is the sampling design. Inferential procedures constructed under these assumptions are referred to as design-based procedures. In this chapter we adopt the design-based perspective throughout, beginning here with a brief review of some basic concepts and techniques, which will provide us with an appropriate set of tools to attack the network graph estimation problem. In order to incorporate measurement errors (and other such sources of randomness) into an analysis, it is generally necessary to specify a model, leading to so-called model-based procedures. We will revisit and illustrate the distinction between design-based and model-based procedures, in the context of network graph estimation, in Section 6.2.4.1.

5.2.1 Horvitz-Thompson Estimation for Totals

Suppose we have a population $\mathcal{U} = \{1, \dots, N_u\}$ of N_u units (e.g., people, animals, widgets, etc.), and suppose that with each unit $i \in \mathcal{U}$ there is associated a value y_i of interest (e.g., height, age, gender, etc.). Let $\tau = \sum_i y_i$ and $\mu = \tau/N_u$ be the total and average values of the y 's in the population. Finally, let $S = \{i_1, \dots, i_n\}$ be a sample of n units from \mathcal{U} , and assume that we observe y_i for each $i \in S$.

Estimation of a population total τ or average μ from a random sample is an ubiquitous task. In the canonical case in which S is chosen by drawing n units uniformly from \mathcal{U} , with replacement, a natural estimate of μ is simply the sample mean $\bar{y} = (1/n) \sum_{i \in S} y_i$. The corresponding estimate of the total is then just

$\hat{\tau} = N_u \bar{y}$. These estimates are unbiased (i.e., $\mathbb{E}(\bar{y}) = \mu$ and $\mathbb{E}(\hat{\tau}) = \tau$), in the sense that the average of their values over all possible samples of size n are the parameters they are estimating. The variances of these estimators take the forms $\mathbb{V}(\bar{y}) = \sigma^2/n$ and $\mathbb{V}(\hat{\tau}) = N_u^2 \sigma^2/n$, where σ^2 is the variance of the values y in the full population \mathcal{U} . These variances can themselves be estimated in an unbiased fashion through the quantities $\widehat{\mathbb{V}}(\bar{y}) = s^2/n$ and $\widehat{\mathbb{V}}(\hat{\tau}) = N_u^2 s^2/n$, respectively, where $s^2 = (1/(n-1)) \sum_{i \in S} (y_i - \bar{y})^2$ is the sample variance.

In many situations, however, the sample S is far from being a simple random sample with replacement. In particular, often some units are more likely than others to be included in a sample, either by design or by accident. A marketing survey might favor calling larger households. A census survey may have more trouble locating homeless people than homeowners. As we will see below, similar scenarios can often arise in the sampling of networks. Sampling under designs of this sort is called *unequal probability sampling*.

When sampling with unequal probabilities, the sample mean can be a poor choice of estimator, in that it will be a biased estimator. The *Horvitz-Thompson estimator* is constructed in a manner that remedies this problem, through the use of weighted-averaging. Suppose that, under a given sampling design, each unit $i \in \mathcal{U}$ has probability π_i of being included in a sample of size n . Let $S \subset \mathcal{U}$ now be the set of distinct units in the sample. Then the Horvitz-Thompson estimate of the total τ takes the form

$$\hat{\tau}_\pi = \sum_{i \in S} \frac{y_i}{\pi_i} . \quad (5.3)$$

The corresponding estimate of the mean μ is obtained as $\hat{\mu}_\pi = (1/N_u) \hat{\tau}_\pi$.

The estimator $\hat{\tau}_\pi$ is an unbiased estimate of τ , assuming $\pi_i > 0$ for all $i \in \mathcal{U}$. To see this, let Z_1, \dots, Z_{N_u} be a set of binary random variables such that $Z_i = 1$ if unit i is in S , and zero otherwise. Then, since

$$\mathbb{E}(\hat{\tau}_\pi) = \mathbb{E}\left(\sum_{i \in S} \frac{y_i}{\pi_i}\right) = \mathbb{E}\left(\sum_{i \in \mathcal{U}} \frac{y_i}{\pi_i} Z_i\right) = \sum_{i \in \mathcal{U}} \frac{y_i}{\pi_i} \mathbb{E}(Z_i) , \quad (5.4)$$

and $\mathbb{E}(Z_i) = \mathbb{P}(Z_i = 1) = \pi_i$, by definition, the result follows. Similarly, $\hat{\mu}_\pi$ is an unbiased estimate of μ .

If we define π_{ij} to be the probability that units i and j are both in the sample S , with $\pi_{ii} = \pi_i$ for convenience when $i = j$, then the variance of our estimator $\hat{\tau}_\pi$ can be expressed as

$$\mathbb{V}(\hat{\tau}_\pi) = \sum_{i \in \mathcal{U}} \sum_{j \in \mathcal{U}} y_i y_j \left(\frac{\pi_{ij}}{\pi_i \pi_j} - 1 \right) , \quad (5.5)$$

and may itself be estimated in an unbiased fashion by the quantity

$$\widehat{\mathbb{V}}(\hat{\tau}_\pi) = \sum_{i \in S} \sum_{j \in S} y_i y_j \left(\frac{1}{\pi_i \pi_j} - \frac{1}{\pi_{ij}} \right) , \quad (5.6)$$

assuming $\pi_{ij} > 0$ for all pairs i, j . Analogous quantities for the estimator $\hat{\mu}_\pi$ follow from the expressions $\mathbb{V}(\hat{\mu}_\pi) = (1/N^2)\mathbb{V}(\hat{\tau}_\pi)$ and $\widehat{\mathbb{V}}(\hat{\mu}_\pi) = (1/N^2)\widehat{\mathbb{V}}(\hat{\tau}_\pi)$. Note that in the case that inclusion of units i and j are independent events, we have $\pi_{ij} = \pi_i \pi_j$, and some simplification of the expressions in (5.5) and (5.6) is possible.

Example 5.2 (Simple Random Sampling Without Replacement). Consider the case where n units are sampled from \mathcal{U} uniformly, but without replacement. That is, i_1 is chosen uniformly from \mathcal{U} and then set aside. Next, i_2 is chosen uniformly from $\mathcal{U} \setminus \{i_1\}$, and so on, until finally i_n is chosen uniformly from $\mathcal{U} \setminus \{i_1, \dots, i_{n-1}\}$.

As there are $\binom{N_u}{n}$ possible samples of size n that may be chosen in this manner, and $\binom{N_u-1}{n-1}$ samples that may be chosen to include a given unit i , it follows that

$$\pi_i = \frac{\binom{N_u-1}{n-1}}{\binom{N_u}{n}} = \frac{n}{N_u}, \quad (5.7)$$

for all $i \in \mathcal{U}$. Arguing similarly, we have $\pi_{ij} = [n(n-1)]/[N_u(N_u-1)]$.

Therefore, it follows that under this design the Horvitz-Thompson estimates of the total and mean have the form

$$\hat{\tau}_\pi = N_u \bar{y} \quad \text{and} \quad \hat{\mu}_\pi = \bar{y}. \quad (5.8)$$

Note that these are the same unbiased estimators described at the start of this section for simple random sampling *with* replacement. However, the variance of, for example, $\hat{\tau}_\pi$ may be shown to be $N_u(N_u - n)\sigma^2/n$, which is strictly less than the value $N_u^2\sigma^2/n$ we saw earlier under simple random sampling with replacement.

In other words, under simple random sampling, the sample mean \bar{y} is unbiased whether sampling with and without replacement. However, \bar{y} has a lower variance when sampling without replacement, as the same number of samples n is used more efficiently by enforcing that all units in S be distinct. \square

Example 5.3 (Probability Proportional to Size Sampling). In many contexts, while sampling is done with replacement, the units are not sampled with equal probability at each stage. Rather, for each of the n times a unit is selected from \mathcal{U} , the unit is selected with respect to a probability distribution $\{p_1, \dots, p_{N_u}\}$ on \mathcal{U} .

This situation naturally arises when units are selected from the population with probabilities p_i directly proportional to the values c_i of some characteristic. For example, households might be selected for a marketing survey by drawing names from a database, in which case those households with more members in the database have a larger chance of being included. Alternatively, a study of wildlife in a remote geographical region might be conducted by flying an aircraft over adjacent strips of land of a given width. But natural geographic features often result in strips being of different lengths, and hence different areas, which is likely to make the chance of seeing a given type of animal vary. Sampling conducted in this manner is called *probability proportional to size* (PPS) sampling.

Because sampling is done with replacement, it is easy to see that the inclusion probabilities take the form $\pi_i = 1 - (1 - p_i)^n$, where $p_i = c_i / \sum_i c_i$, and hence are unequal if the p_i are unequal. The Horvitz-Thompson estimators of τ and μ are therefore more appropriate here than the sample mean, particularly if the p_i are decidedly non-uniform. Accompanying estimates of the variance of these estimators can be computed using the values $\pi_{ij} = \pi_i + \pi_j - [1 - (1 - p_i - p_j)^n]$. \square

As can be seen, Horvitz-Thompson estimators are in principle quite general in their applicability. In reality, of course, they are limited to those contexts in which the values of the π_i may be computed for the underlying sampling design. Such computations tend to be noticeably more difficult for designs that sample without replacement. Nevertheless, we will see in Section 5.4 a number of different network graph sampling designs for which the values π_i may be computed. And even in cases for which it appears difficult or impossible to compute these inclusion probabilities, the Horvitz-Thompson framework nonetheless provides a perspective by which valuable insight may be gained.

5.2.2 Estimation of Group Size

A special type of population total that is often of interest is the size of a group. Note, for example, that in the previous material it was implicitly assumed that N_u , the size of the population \mathcal{U} , is known. Many times this is simply not true. In fact, in some cases estimating N_u is an important goal in and of itself. For example, there are many populations that are ‘hard to find’ and yet whose size is important to know for planning purposes, such as populations of endangered animal species or populations of humans at risk for a particular rare disease or condition.

In principle, we may write $N_u = \sum_{i \in \mathcal{U}} 1$, which would suggest the Horvitz-Thompson estimator $\hat{N}_u = \sum_{i \in S} \pi_i^{-1}$. Unfortunately, as we have seen in Examples 5.2 and 5.3, knowledge of N_u is typically needed to compute the π_i , which makes this approach infeasible.

Instead, alternative techniques have been developed for this particular estimation problem, the primary example of which is the class of *capture-recapture* estimators. The simplest version of capture-recapture involves two stages of simple random sampling without replacement, yielding two samples, say S_1 and S_2 . In the first stage, the sample S_1 of size n_1 is taken, and all of the units in S_1 are ‘marked.’ Marking might correspond to literally tagging a fish or animal, or simply noting the ID number of a record in a database. All of the units in S_1 are then ‘returned’ to the population – either literally or figuratively – and, at the second stage of sampling, a sample of size n_2 is taken from \mathcal{U} . The value

$$\hat{N}_u^{(c/r)} = \frac{n_2}{m} n_1 \tag{5.9}$$

is then used as an estimate of N_u , where $m = |S_1 \cap S_2|$ is the number of marked units observed in the second sample. An estimator of the variance is

$$\widehat{\mathbb{V}}(\hat{N}_u^{(c/r)}) = \frac{n_1 n_2 (n_1 - m)(n_2 - m)}{m^3} . \quad (5.10)$$

The capture-recapture estimator in (5.9) may be seen to adjust upward the size n_1 of the first sample by one over the factor m/n_2 , an indication of what fraction of the overall population was marked. It has been derived using a number of different arguments. For example, if n_1 and n_2 are fixed in advance, m has a hypergeometric distribution, and the integer part of $\hat{N}_u^{(c/r)}$ can be shown to be the corresponding maximum likelihood estimate of N_u . Similar estimators have been derived under various other models and assumptions, allowing for such changes as random n_1, n_2 , unequal probability sampling of units, changes in inclusion probabilities between the first and second stages of sampling, and more than two stages of sampling.

In Section 5.5 we will consider the problem of estimating the size of a ‘hard to find’ sub-population, not using capture-recapture sampling but rather a certain type of network sampling. Nevertheless, we will see that useful estimators may be derived with a form quite similar to that in (5.9).

An important related problem of group-size estimation is the so-called *species problem*. This problem takes its name from biology, where the standard caricature is that of a lonely biologist sitting in the middle of a forest observing animals that pass within view. While the biologist records each animal sighted, in the end it is actually the number of species of animals in the forest, say N_s , that is of interest. Despite the biological motivation of its name, this problem arises quite widely. For example, the estimation of the size of an author’s apparent vocabulary from published works (e.g., “How many words did Shakespeare know?”), or the estimation of the number of ancient coins minted by a society based on archaeological finds, have both been cast as species problems.

In the special case that each species has the same proportion of members in \mathcal{U} , under random sampling the problem of estimating N_s arguably is not much different from that of estimating N_u . Of course, the assumption of equal proportions of members among species is typically not true. In reality, unfortunately, unless there is prior knowledge to the contrary, there is always the possibility that there are an arbitrary number of species in the population in arbitrarily small proportions (given a population of sufficient size). This fact allows for the species problem to potentially be quite ill-posed.

There are numerous estimators of N_s that have been proposed in the literature, ranging from parametric to nonparametric, and including both frequentist and Bayesian. A common method, attributed to Good [183], is to take the observed number of species, say n_{obs} , and adjust it upward by a certain factor,

$$\hat{N}_s^{(cov)} = \frac{n_s^{obs}}{\hat{c}} , \quad (5.11)$$

where $\hat{c} = 1 - x_1/n$, for x_1 equal to the number of species observed only once in the sample of size n . The estimator $\hat{N}_s^{(cov)}$ is a so-called *coverage estimator*, because the factor \hat{c} is an estimate of the coverage c of the sample – that is, of the fraction of the population corresponding to those species observed at least once. This is a nonparametric estimator, but it has asymptotic behavior quite close to that of the maximum likelihood estimator, and it is generally much easier than the latter to compute. It has been noted, however, that in small samples these estimators can suffer from significant bias and large variance. Relevant details may be found in Darroch and Ratcliff [108] or Esty [138], for example.

We will see in Section 5.5 an example of a network species problem in the context of the Internet and a corresponding nonparametric estimator similar in spirit to (5.11).

5.3 Common Network Graph Sampling Designs

We here introduce some common designs for sampling a population network graph G . For convenience, our treatment is entirely in terms of undirected graphs, but most of the expressions can be generalized in a straightforward fashion to the case of directed graphs.

Graph sampling designs are somewhat distinct from typical sampling designs in non-network contexts (such as those seen above), in that there are effectively two inter-related sets of units being sampled, vertices i and edges $\{i, j\}$. Often these designs can be characterized as having two stages: a selection stage, followed by an observation stage. Selection is generally made among one class of units (e.g., vertices), which then leads to observation of units from the other class (e.g., edges) or even both classes. We will derive vertex and edge inclusion probabilities for the designs introduced below. These probabilities will be valuable both in lending insight into the nature of the designs and in providing us with a central element necessary to define Horvitz-Thompson estimators for certain key classes of network graph characteristics $\eta(G)$ in the next section.

5.3.1 Induced and Incident Subgraph Sampling

We first consider *induced subgraph sampling*, which consists of taking a random sample of vertices in a graph G and observing their induced subgraph. More precisely, in this design a simple random sample of n vertices is selected from V , without replacement, which yields the set $V^* = \{i_1, \dots, i_n\}$. Edges are then observed for all vertex pairs $i, j \in V^*$ for which $\{i, j\} \in E$, yielding the set E^* . Note that this same design was encountered already in Example 5.1. A schematic illustration is shown in Figure 5.2. This type of sampling is representative of, say, the construction of contact networks in social network research, when a sample of individuals is first

selected and then the individuals are interviewed regarding some measure of contact among themselves (e.g., friendship, likes or dislikes, etc.).

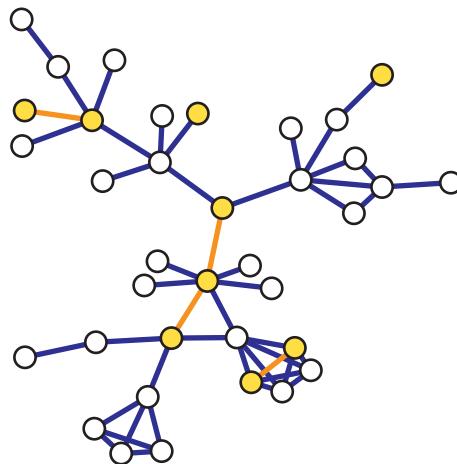


Fig. 5.2 Schematic illustration of induced subgraph sampling. Selected nodes are shown in yellow, while observed edges are shown in orange.

For induced subgraph sampling, the vertex and edge inclusion probabilities are uniformly equal to

$$\pi_i = \frac{n}{N_v} \quad \text{and} \quad \pi_{\{i,j\}} = \frac{n(n-1)}{N_v(N_v-1)}, \quad (5.12)$$

for all $i \in V$ and all $(i, j) \in V^{(2)}$, where $V^{(2)}$ is the set of all unordered pairs of vertices. These expressions follow from exactly the same arguments as underlie the calculation of the probabilities π_i and π_{ij} in Example 5.2. Note that they require knowledge of N_v . In the case of sampling a contact network, for example, this value would likely be available from database records if the overall population consisted of, say, all employees of a corporate entity.

Complementary to induced subgraph sampling is *incident subgraph sampling*. Instead of selecting n vertices in the initial stage, n edges are selected, again through simple random sampling without replacement, directly yielding the set E^* . All vertices incident to the selected edges are then observed, thus providing V^* . This design is illustrated in Figure 5.3. Such a design is, for example, implicit in the construction of sampled telephone call graphs, wherein telephone calls are sampled from a database, after which the phone numbers of the initiator and the receiver of the call are observed.

Regarding the inclusion probabilities for incident subgraph sampling, clearly the edge inclusion probabilities are just $\pi_{\{i,j\}} = n/N_e$. The form of the vertex inclusion probabilities, however, is more complicated, due to the fact that a vertex is included

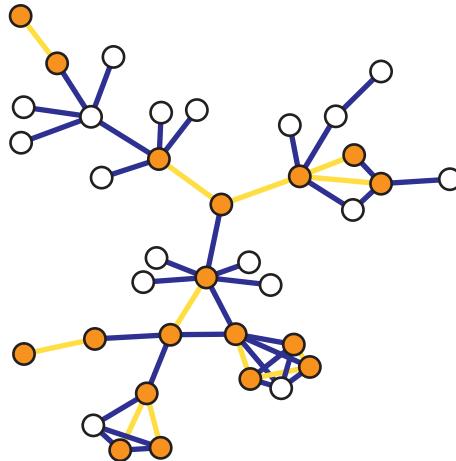


Fig. 5.3 Schematic illustration of incident subgraph sampling. Selected edges are shown in yellow, while observed nodes are shown in orange.

whenever any one or more of its incident edges is sampled. Specifically, we have

$$\begin{aligned}
 \pi_i &= \mathbb{P}(\text{ vertex } i \text{ is sampled}) \\
 &= 1 - \mathbb{P}(\text{ no edge incident to } i \text{ is sampled}) \\
 &= \begin{cases} 1 - \frac{\binom{N_e - d_i}{n}}{\binom{N_e}{n}}, & \text{if } n \leq N_e - d_i, \\ 1, & \text{if } n > N_e - d_i, \end{cases} \tag{5.13}
 \end{aligned}$$

where d_i is the degree of vertex i .

Hence, in incident subgraph sampling, while the edges are included in the sample graph G^* with equal probability, the vertices are included with unequal probabilities that depend on the degree sequence. Looked at another way, incident subgraph sampling induces probability proportional to size sampling of vertices, reminiscent of Example 5.3, where ‘size’ here is driven by vertex degree. Clearly, to compute these inclusion probabilities, knowledge of both the number of edges N_e and the degrees of the nodes must be available. In the example of sampling a telephone call graph, for instance, this would require having access to marginal summaries of the total number of calls (say, in a given month) as well as the number of calls in which a given phone number had participated.

5.3.2 Star and Snowball Sampling

Another possible design is *star sampling*. In star sampling, an initial vertex sample V_0^* of size n is taken. Then all edges incident to vertices $i \in V_0^*$ are observed, yielding

E^* , and thus $G^* = (V^*, E^*)$, with $V^* = V_0^*$. Additionally, it may be that it is possible to observe not only these incident edges, but also the vertices $i \in V \setminus V_0^*$ to which these edges are also incident, thus enlarging the set V^* accordingly. These two variations have been referred to as *unlabeled* and *labeled* star sampling, respectively. In conducting a study of co-authorship among people within a given scientific literature, for example, randomly sampling records of n authors and recording the total number of co-authors of each author would correspond to unlabeled star sampling; if not only the number but the identities of the co-authors are recorded, this would correspond to labeled star sampling. A variant of unlabeled star sampling was also seen already in Example 5.1.

The vertex inclusion probabilities under unlabeled star sampling are just $\pi_i = n/N_v$, if the initial vertex set V_0^* is selected as a simple random sample without replacement. The edge inclusion probabilities, using an argument similar to that underlying (5.13), take the form

$$\begin{aligned}\pi_{\{i,j\}} &= \mathbb{P}(\text{neither } i \text{ nor } j \text{ are sampled}) \\ &= 1 - \frac{\binom{N_v-2}{n}}{\binom{N_v}{n}},\end{aligned}\tag{5.14}$$

which is roughly equal to $1 - (1 - n/N_v)^2$ for large n and N_v .

Note that the expression for the edge inclusion probabilities remains unchanged under the labeled version of star sampling. However, the vertex probabilities are indeed affected and can be shown to look like

$$\pi_i = \sum_{L \subseteq \mathcal{N}_i^+} (-1)^{|L|+1} \mathbb{P}(L),\tag{5.15}$$

where \mathcal{N}_i^+ is the union of vertex i and those vertices that are immediate neighbors, $|L|$ is the cardinality of a given set L , and $\mathbb{P}(L)$ is the probability of selecting the set L in obtaining S . For example, under simple random sampling without replacement, we have

$$\mathbb{P}(L) = \frac{\binom{N_v-|L|}{n-|L|}}{\binom{N_v}{n}}.\tag{5.16}$$

The calculation of probabilities like those in (5.15) and (5.16) is simplified somewhat if we model the selection of vertices for the initial vertex subset V_0^* as arising through Bernoulli sampling. That is, the presence or absence of each vertex $i \in V$ in the sample V_0^* is held to be determined by an independent coin flip with probability p . In the case where N_v is large and $p \approx n/N_v$ is small, this Bernoulli sampling design is a quite reasonable approximation to simple random sampling without replacement.

An extension of the idea of star sampling is *snowball sampling*. Specifically, whereas (labeled) star sampling extends an initial vertex sample V_0^* to only those vertices that are immediate neighbors of these, a snowball sample iterates this process in a natural manner. Define $\mathcal{N}(S)$ to be the set of all neighbors of vertices in a

set S . Then snowball sampling extends V_0^* to $V_1^* = \mathcal{N}(V_0^*) \cap \bar{V}_0^*$, and extends V_1^* to $V_2^* = \mathcal{N}(V_1^*) \cap \bar{V}_0^* \cap \bar{V}_1^*$, and so on. The set V_k^* is called the k -th *wave* of the sampling process. Sampling can be continued until a wave V_k^* is reached that is empty, or it can be stopped after some number K stages. Star sampling corresponds to the case $K = 1$. The final graph G^* obtained through snowball sampling consists of the vertices in $V^* = V_0^* \cup V_1^* \cup \dots \cup V_K^*$ and, by construction, their incident edges.

An illustration of two-stage snowball sampling is shown in Figure 5.4. Some surveys of the World Wide Web graph can be thought of as arising through a variant of snowball sampling. Computer programs called ‘spiders’ are written to follow an initially compiled subset V_0^* of web pages to those pages corresponding to the HTML addresses listed in the V_0^* -pages. The newly discovered pages constitute S_1 , and these in turn are then examined for new HTML addresses, which are then pursued. In other words, the ‘spiders’ mimic a human using a web browser by exhaustively following hyperlinks on discovered web pages.²

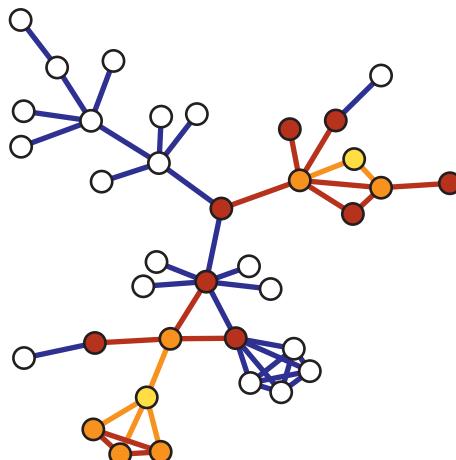


Fig. 5.4 Schematic illustration of two-stage snowball sampling. Nodes selected in the initial sampling are shown in yellow, while edges and nodes observed in the first and second waves of sampling are shown in orange and brown, respectively.

Unfortunately, although not surprisingly, inclusion probabilities for snowball sampling become increasingly intractable to calculate after the one-stage level corresponding to star sampling. The underlying formulas follow in principle from application of inclusion-exclusion arguments, but these lead to combinatorial issues that must be faced. A useful resource in this regard is Frank [148], although the treatment therein is still almost exclusively that of one-stage snowball sampling.

² Hyperlinks are directional, with one web page referencing another, and thus the web graph is more properly considered as a directed graph, a detail we have ignored here for the sake of exposition.

5.3.3 Link Tracing

There are various other designs for sampling network graphs. Many of them fall under the general label of *link-tracing* designs, in which after the selection of an initial sample, some subset of the edges ('links') from vertices in this sample are traced to additional vertices. Snowball sampling is a special case of link-tracing, in that all edges are followed until termination. But often it is not feasible, nor perhaps even desirable, to follow all edges. For example, in sampling social contact networks, it may be that individuals are unaware of or cannot recall all of their contacts, or that they do not wish to divulge some of them.

The inclusion probabilities under link-sampling of course vary with the design. As an illustration, consider the following idealized version of the traceroute sampling underlying Internet topology surveys like that described in Section 3.5.2. A sample $S = \{s_1, \dots, s_{n_s}\}$ of n_s 'sources' are selected from the vertex set V of an Internet network graph G . Then a sample $T = \{t_1, \dots, t_{n_t}\}$ of n_t 'targets' are selected from $V \setminus S$. Finally, a route is traced from each source node in S to each target node in T . That is, effectively a path is sampled between each pair $(s_i, t_j) \in S \times T$, and all vertices and edges in the paths are observed. The sampled graph $G^* = (V^*, E^*)$ is then constructed as the union of vertices and edges over all sampled paths. An illustration is shown in Figure 5.5.

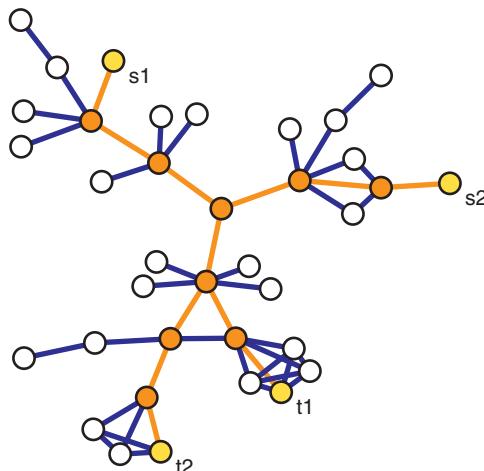


Fig. 5.5 Schematic illustration of the traceroute version of link-tracing. Selected source nodes $\{s_1, s_2\}$ and target nodes $\{t_1, t_2\}$ are shown in yellow, while nodes and edges observed on traces from sources to targets are shown in orange.

If the source and target sets are assumed to be obtained through simple random sampling without replacement, and if it is assumed that the corresponding paths sampled are shortest paths with respect to some set of edge weights, then it is argued in Dall'Asta et al. [107] that the vertex and edge inclusion probabilities behave like

$$\pi_i \approx 1 - (1 - \rho_s - \rho_t) \exp(-\rho_s \rho_t b_i) \quad (5.17)$$

and

$$\pi_{\{i,j\}} \approx 1 - \exp(-\rho_s \rho_t b_{i,j}) , \quad (5.18)$$

where b_i is the vertex betweenness centrality of i , $b_{i,j}$ is the edge betweenness centrality of edge $\{i,j\}$, and $\rho_s = n_s/N_v$ and $\rho_t = n_t/N_v$ are the source and target sampling fractions, respectively. From these expressions we see that traceroute sampling induces a type of probability proportional to size sampling of vertices and edges, where ‘size’ varies with the vertex and edge betweenness centralities.

Although the probabilities in (5.17) and (5.18) are not calculable from the traceroute observations themselves, they lend interesting insight into the nature of this sampling design, which we will return to in Section 5.5.

5.4 Estimation of Totals in Network Graphs

With appropriate choice of a population of units \mathcal{U} and unit values y , various graph summary characteristics $\eta(G)$ encountered in Chapter 4 can be written in a form that involves a total $\tau = \sum_{i \in \mathcal{U}} y_i$. Examples include the following.

- Let $\mathcal{U} = V$ and $y_i = d_i$. The average degree of a graph G is obtained by scaling the total $\sum_{i \in V} d_i$ by N_v .
- Let $\mathcal{U} = E$ and $y_{\{i,j\}} = 1$. The number of edges N_e is given by the total $\sum_{\{i,j\} \in E} 1$.
- Let $\mathcal{U} = V^{(2)}$ and $y_{(i,j)} = I_{k \in \mathcal{P}(i,j)}$ be the indicator of the event that the shortest path between i and j contains node k . In the case of unique shortest paths, the betweenness centrality $c_B(k)$ of a vertex $k \in V$ is given by the total $\sum_{(i,j) \in V^{(2)}} I_{k \in \mathcal{P}(i,j)}$.
- Let $\mathcal{U} = V^{(3)}$ be the set of all triples of distinct vertices (i, j, k) . The clustering coefficient $cl_T(G)$ – that is, the transitivity of the graph G – can be written as three times the ratio of the total number of triples that form triangles to the total number of connected triples.

Often totals such as these can be estimated from a sampled graph G^* using generalizations of the Horvitz-Thompson estimator in (5.3), some instances of which we describe now.

5.4.1 Vertex Totals

As a starting point, consider the case of a *vertex total* (i.e., a total of the form $\tau = \sum_{i \in V} y_i$, for some choice of y). The first example above, involving the average degree, may be viewed as a vertex total. Another example is when y is a binary

variable indicating that a vertex has a given characteristic, in which case τ counts the number of vertices with that characteristic, and τ/N_v , the proportion. Quantities of this sort might include the fraction of web pages on the World Wide Web with links pointing to the Google homepage or the number of genes in a gene regulatory network responsible for the growth of an organism.

Given a sample of vertices $V^* \subseteq V$, we see from (5.3) that the Horvitz-Thompson estimator for vertex totals takes the form

$$\hat{\tau}_\pi = \sum_{i \in V^*} \frac{y_i}{\pi_i}, \quad (5.19)$$

where the π_i are the vertex inclusion probabilities corresponding to the underlying network sampling design. The variance of $\hat{\tau}_\pi$ in (5.19) and an unbiased estimate of that variance similarly follow directly from (5.5) and (5.6), with \mathcal{U} and S replaced by V and V^* , respectively.

Note that in some cases the role of the network will be irrelevant for estimating a vertex total, such as when the characteristic y has nothing to do with the relational measurements defining the network and vertices are sampled through simple random sampling without replacement. In such cases, the estimator in (5.19) is just the conventional Horvitz-Thompson estimator. On the other hand, the network would be relevant if snowball sampling were used, since in that case the network structure itself plays a key role in the sampling of V^* and hence the calculation of the π_i .

5.4.2 Totals on Vertex Pairs

Now suppose that a quantity y_{ij} corresponding to vertex pairs $(i, j) \in V^{(2)}$ is of interest. Totals of the form

$$\tau = \sum_{(i,j) \in V^{(2)}} y_{ij} \quad (5.20)$$

are then relevant. The second and third examples at the start of this section, involving the number of edges N_e and the vertex betweenness c_B , are examples of this type of total. Examples of other quantities involving such totals include the total number of adjacent vertices with mutual labels (e.g., shared gender in a friendship network) and the average over the graph of some sort of (dis)similarity value between vertex pairs.

The Horvitz-Thompson estimator in this context takes the form

$$\hat{\tau}_\pi = \sum_{(i,j) \in V^{*(2)}} \frac{y_{ij}}{\pi_{ij}} \quad (5.21)$$

for estimating the total τ in (5.20). In the case that y_{ij} defining τ are nonzero only for vertex pairs (i, j) corresponding to edges $\{i, j\} \in E$, τ is an *edge total*, and

the inclusion probabilities in (5.21) are just the edge inclusion probabilities (i.e., $\pi_{ij} = \pi_{\{i,j\}}$).

Generalizing (5.5), the variance of the above estimator is given by

$$\mathbb{V}(\hat{\tau}_\pi) = \sum_{(i,j) \in V^{(2)}} \sum_{(k,l) \in V^{(2)}} y_{ij} y_{kl} \left(\frac{\pi_{ijkl}}{\pi_{ij}\pi_{kl}} - 1 \right) , \quad (5.22)$$

where π_{ijkl} is the probability that vertices i, j, k , and l are all included in the sample, and $\pi_{ijkl} = \pi_{ij}$ for convenience when $(i,j) = (k,l)$. The corresponding unbiased estimate of variance is given by

$$\hat{\mathbb{V}}(\hat{\tau}_\pi) = \sum_{(i,j) \in V^{*(2)}} \sum_{(k,l) \in V^{*(2)}} y_{ij} y_{kl} \left(\frac{1}{\pi_{ij}\pi_{kl}} - \frac{1}{\pi_{ijkl}} \right) . \quad (5.23)$$

Note that these quantities can become increasingly complicated to compute under some sampling designs, since it is necessary to be able to evaluate probabilities π_{ijkl} that four-tuples of vertices are sampled.

Example 5.4 (Estimating the Size of a Network Graph). Consider the problem of estimating N_e as an edge total, i.e.,

$$N_e = \sum_{\{i,j\} \in E} 1 = \sum_{(i,j) \in V^{(2)}} I_{\{i,j\} \in E} . \quad (5.24)$$

Frank [149] derives explicit formulas for the Horvitz-Thompson estimator, its variance, and an unbiased estimate of this variance, in the context of induced subgraph sampling, where the vertices are sampled either with replacement, without replacement, or through Bernoulli sampling. For such designs – which Frank terms *symmetric sampling designs* – the quadruple vertex inclusion probabilities π_{ijkl} are functions only of the number of different vertices among i, j, k , and l , and not of the actual vertices themselves. That is, if there are $1 \leq r \leq 4$ different vertices among i, j, k , and l , then $\pi_{ijkl} = p_r$, where

$$p_r = \binom{n}{r} / \binom{N_v}{r} \quad (5.25)$$

for sampling n vertices without replacement,

$$p_r = \sum_{m=0}^r (-1)^m \binom{r}{m} (1 - m/N_v)^n \quad (5.26)$$

for sampling with replacement, and $p_r = p^r$ for Bernoulli sampling with probability p .

The estimator (5.21) of N_e in this situation is just

$$\hat{N}_e = \sum_{(i,j) \in V^{*(2)}} \pi_{ij}^{-1} = p_2^{-1} N_e^* , \quad (5.27)$$

which simply scales up the empirically observed total N_e^* by a factor p_2^{-1} . The variance (5.22) takes the form

$$\mathbb{V}(\hat{N}_e) = \alpha_0 N_e^2 + \alpha_1 \mathcal{Q} + \alpha_2 N_e , \quad (5.28)$$

where $\mathcal{Q} = \sum_{i \in V} d_i^2$ is the sum of squares of the vertex degrees in G and

$$\begin{aligned}\alpha_0 &= (p_4 - p_2^2)/p_2^2 \\ \alpha_1 &= (p_3 - p_4)/p_2^2 \\ \alpha_2 &= (p_2 - 2p_3 + p_4)/p_2^2 .\end{aligned}\quad (5.29)$$

The variance estimator in (5.23) becomes

$$\widehat{\mathbb{V}}(\hat{N}_e) = \beta_0 N_e^{*2} + \beta_1 \mathcal{Q}^* + \beta_2 N_e^* , \quad (5.30)$$

where \mathcal{Q}^* is the analogue of \mathcal{Q} on G^* and

$$\begin{aligned}\beta_0 &= \frac{1}{p_2^2} - \frac{1}{p_4} \\ \beta_1 &= \frac{1}{p_4} - \frac{1}{p_3} \\ \beta_2 &= \frac{2}{p_3} - \frac{1}{p_2} - \frac{1}{p_4} .\end{aligned}\quad (5.31)$$

Figure 5.6 shows the results of a numerical simulation, in which the true graph G is the network of protein interactions in yeast from Section 4.2.1.1, as in Example 5.1. Induced subgraph sampling was simulated in each of 10,000 trials, using Bernoulli sampling of vertices with $p = 0.10, 0.20$, and 0.30 . Shown in the figure are histograms of the estimators \hat{N}_e in (5.27), for each choice of p , and of the estimated standard errors of these estimators, based on (5.30). The average of \hat{N}_e over the simulations was 31116, 31197, and 31203, for $p = 0.10, 0.20$, and 0.30 , respectively. Thus the unbiasedness of all three estimators in estimating $N_e = 31,201$ is well-supported by these results. The unbiasedness of the estimated variances (5.30) was similarly supported. Note that we do not expect unbiasedness of the estimated standard error, as the latter is a nonlinear function of the estimated variance. However, the average of these values is in fact reasonably close to the true underlying values. Note too, from the figure, that not only does the mean estimated standard error decrease with p , but so too does the variability of this quantity. \square

Frank [151] offers results similar to those above for the estimation of certain totals in directed graphs.

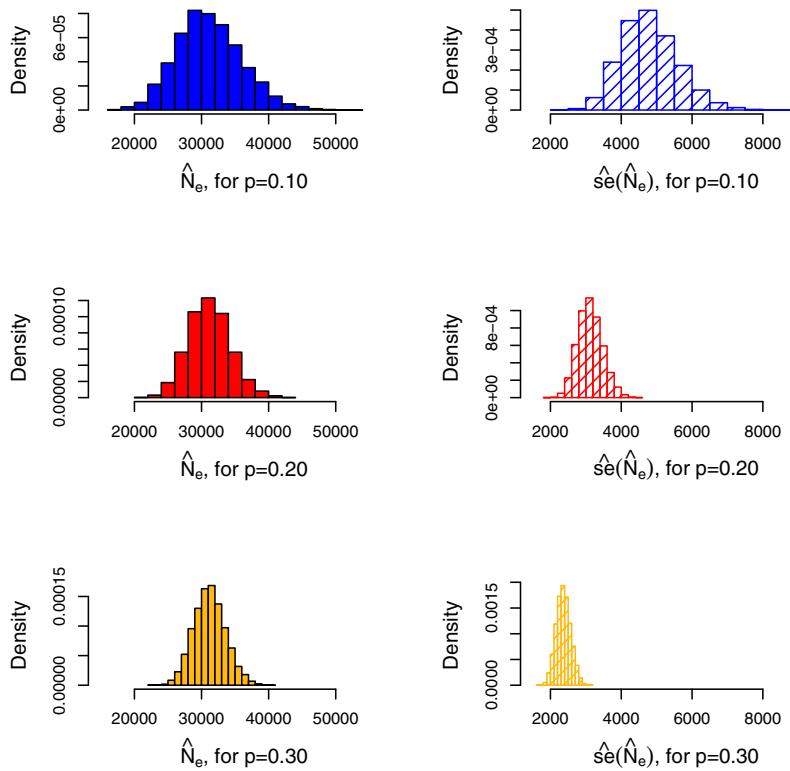


Fig. 5.6 Histograms of estimates \hat{N}_e (left) of $N_e = 31,201$, as well as estimated standard errors (right), in the yeast protein interaction network, under induced subgraph sampling, with Bernoulli sampling of vertices, using $p = 0.10$ (blue), 0.20 (red), and 0.30 (yellow). Results based on 10,000 trials.

5.4.3 Totals of Higher Order

Having considered estimation of totals on vertices and vertex pairs, it should be clear that we can in principle continue in this manner and tackle totals on subsets of vertices of arbitrarily higher order. In practice, however, the relevant expressions become quite complicated. Triples of vertices appear to be both the order of next most natural interest beyond vertex pairs and the only other order studied in any detail.

For the case of totals $\tau = \sum_{(i,j,k) \in V^{(3)}} y_{ijk}$ indexed by triples of vertices, similar reasoning to that above leads to the Horvitz-Thompson estimator

$$\hat{\tau}_\pi = \sum_{(i,j,k) \in V^{*(3)}} \frac{y_{ijk}}{\pi_{ijk}} . \quad (5.32)$$

Accompanying expressions for variance and estimated variance follow in a like manner, but we will not state them here. However, the following example provides some insight into these quantities.

Example 5.5 (Estimation of cl_T). Recall the definition of the clustering coefficient cl_T of a graph G , as given in (4.13). This quantity may be re-expressed in the form

$$\text{cl}_T(G) = \frac{3\tau_\Delta(G)}{\tau_3^\dagger(G) + 3\tau_\Delta(G)} , \quad (5.33)$$

where $\tau_3^\dagger(G) = \tau_3(G) - 3\tau_\Delta(G)$ is defined to be the number of vertex triples that are connected by exactly two edges. Thus $\text{cl}_T(G)$ is a function of two different totals of the form $\sum_{(i,j,k) \in V^{(3)}} y_{ijk}$, where

$$y_{ijk} = A_{ij}A_{jk}A_{ki} \quad (5.34)$$

in the case of $\tau_\Delta(G)$ and

$$y_{ijk} = A_{ij}A_{jk}(1 - A_{ki}) + A_{ij}(1 - A_{jk})A_{ki} + (1 - A_{ij})A_{jk}A_{ki} \quad (5.35)$$

in the case of $\tau_3^\dagger(G)$, and A is the adjacency matrix of G .

The network of protein interactions in yeast has $\tau_\Delta(G) = 44,858$ triangles, $\tau_3^\dagger(G) = 1,006,575$ triples connected by exactly two edges, and a clustering coefficient $\text{cl}_T(G) = 0.1179$. We simulated 10,000 trials of induced subgraph sampling, with Bernoulli sampling of vertices, using $p = 0.20$. Unbiased estimates of the two totals were obtained through the formulas $\hat{\tau}_\Delta(G) = p^{-3}\tau_\Delta(G^*)$ and $\hat{\tau}_3^\dagger(G) = p^{-3}\tau_3^\dagger(G^*)$. Histograms of the resulting values are shown in Figure 5.7. The average value for these estimators over the 10,000 trials was 44,681 and 1,004,963, respectively, while their standard errors were 12,425 and 245,051. The results therefore support the unbiasedness of the estimators, but suggest that both are rather variable.

A plug-in estimator of $\text{cl}_T(G)$ follows by the appropriate substitution of $\hat{\tau}_\Delta(G)$ and $\hat{\tau}_3^\dagger(G)$ in (5.33). Note that the resulting value $\hat{\text{cl}}_T$ is just $\text{cl}_T(G^*)$. We see that under the sampling design used here, this estimator appears to perform reasonably well, with an average value of 0.1191 and a standard error of 0.0251 over the 10,000 trials. \square

Results for estimation of analogous totals on vertex triples in directed graphs may be found in Frank [151], for the case of induced subgraph sampling, with simple random sampling of vertices without replacement.

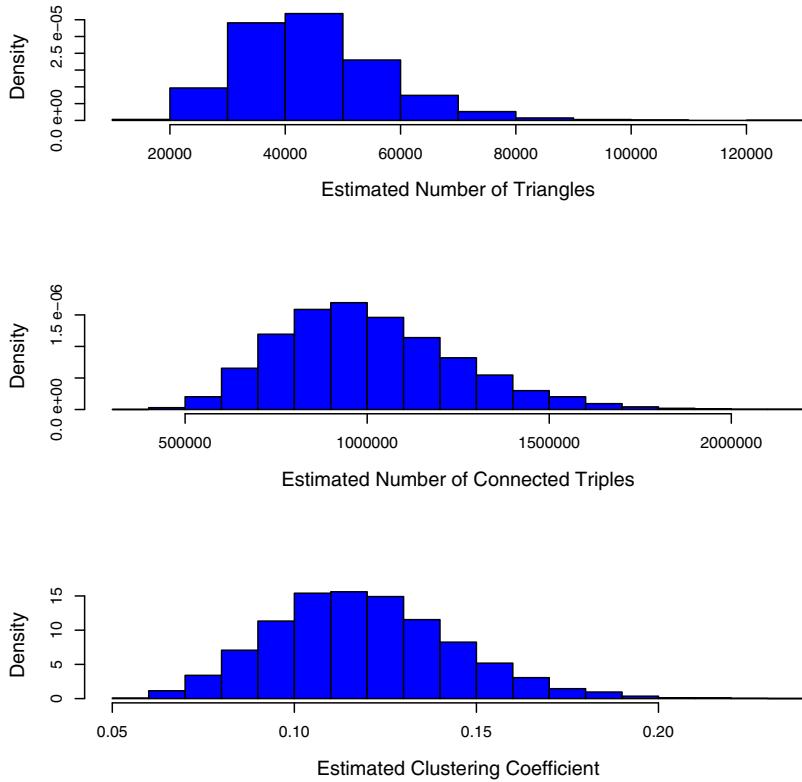


Fig. 5.7 Histograms of estimates $\hat{\tau}_\Delta(G)$ (top), $\hat{\tau}_3^\dagger(G)$ (middle), and $\hat{cl}_T(G)$ (bottom) in the yeast protein interaction network, under induced subgraph sampling, with Bernoulli sampling of vertices, using $p = 0.20$. True values being estimated were $\tau_\Delta(G) = 44,858$, $\tau_3^\dagger(G) = 1,006,575$, and $cl_T(G) = 0.1179$. Results based on 10,000 trials.

5.4.4 Effects of Design, Measurement, and Total

On a final note, although the above-described framework for estimation of various totals in network graphs is fairly straightforward in its essence, it is important to realize that the interaction of the sampling design, the measurements taken, and the total to be estimated has a direct effect on whether and how this framework may be used effectively. In particular, three elements must be present in the problem at hand. First, the network graph summary statistic $\eta(G)$ must, of course, be expressed in terms of a total(s). Second, the values y underlying the definition of this total must be either observed or obtainable from the available measurements. Third, the relevant inclusion probabilities π must be computable for the underlying network

sampling design. Unfortunately, it is not always the case that all three elements are present at the same time. The following example helps illustrate this point.

Example 5.6 (Estimating Average Degree (Continued)). We consider again the problem of estimating the average degree of a network graph G (i.e., $\mu = \tau/N_v$, where $\tau = \sum_{i \in V} d_i$). Also, as in Example 5.1, we again consider two possible network sampling designs – unlabeled star sampling and induced subgraph sampling. We write the resulting sampled graphs as $G_{St}^* = (V_{St}^*, E_{St}^*)$ and $G_{IS}^* = (V_{IS}^*, E_{IS}^*)$, respectively. Recall that under both designs vertices are selected through simple random sampling without replacement from V , but that all edges adjacent to vertices $i \in V_{St}^*$ are observed in forming E_{St}^* , whereas only edges between pairs of vertices $i, j \in V_{IS}^*$ are observed in forming E_{IS}^* .

The average degree is clearly a rescaling of a vertex total. Therefore, it seems natural to use the estimator in (5.19) here. Implicitly, this approach requires that $y_i = d_i$ be observed. Under the star sampling design this is indeed the case, and we arrive at the estimator

$$\hat{\mu}_{St} = \frac{\hat{\tau}_{St}}{N_v} , \quad \text{where} \quad \hat{\tau}_{St} = \sum_{i \in V_{St}^*} \frac{d_i}{n/N_v} . \quad (5.36)$$

On the other hand, under induced subgraph sampling, we do not observe d_i . Rather, as remarked in Example 5.1, we effectively observe a number $d_i^* \leq d_i$ for each vertex $i \in V_{IS}^*$. As a result, τ is not amenable to Horvitz-Thompson estimation methods as a vertex total under this design.

Note, however, the relation $\mu = 2N_e/N_v$ shows that estimation of μ may alternatively be approached as a variation on the problem of estimating the graph size N_e . And under induced subgraph sampling, as we saw above in Example 5.4, we can estimate N_e in an unbiased fashion by

$$\hat{N}_{e,IS} = \sum_{\{i,j\} \in E_{IS}^*} \frac{1}{n(n-1)/[N_v(N_v-1)]} = N_{e,IS}^* \cdot \frac{N_v(N_v-1)}{n(n-1)} , \quad (5.37)$$

which yields the unbiased estimator

$$\hat{\mu}_{IS} = \frac{2\hat{N}_{e,IS}}{N_v} \quad (5.38)$$

for μ .

The estimators (5.36) and (5.38) may be usefully compared by re-writing them as

$$\hat{\mu}_{St} = \frac{2N_{e,St}^*}{n} \quad \text{and} \quad \hat{\mu}_{IS} = \frac{2N_{e,IS}^*}{n} \cdot \frac{N_v - 1}{n - 1} . \quad (5.39)$$

We see that under star sampling our estimator simply results from the use of the formula $2N_e/N_v$ with the graph G_{St}^* , since $N_{St}^* = n$. In contrast, under induced subgraph sampling, the analogous quantity is inflated by the factor $(N_v - 1)/(n - 1)$, which

seeks to correct for the extent to which the degrees d_i are ‘thinned’ in observing the d_i^* . \square

5.5 Estimation of Network Group Size

Throughout most of the previous section, we implicitly assumed that N_v was known. In some settings, this may not be the case. Some groups, like many human and animal populations, are too mobile or elusive to count accurately. Others, like the collection of all web pages or all Internet routing devices, are too massive and dispersed to survey in their entirety. In these settings, estimation of N_v itself may be a prime objective.

If it is feasible to use a network sampling design that selects vertices through simple random sampling without replacement or Bernoulli sampling, then we may entertain the idea of doing so twice, after ‘marking’ the first sample, and using capture-recapture estimators like that in (5.9) directly ‘off the shelf.’ When it is not feasible to select vertices in this manner, or when the basic capture-recapture estimator performs poorly, it may be necessary or at least desirable to develop analogous estimators of N_v explicitly tailored to the sampling and network at hand. We illustrate with two examples.

Example 5.7 (Estimating the Size of a ‘Hidden Population’). The term ‘hidden population’ generally refers to one in which the individuals do not wish to expose themselves to view. For example, humans of socially sensitive status, such as the homeless, or who are involved in socially sensitive activities, such as illegal drug usage or prostitution, will typically not be inclined to disclose themselves. This tendency, coupled with the fact that such subpopulations often are also quite small, makes estimating their size a particularly challenging problem. Simply sampling the overall population and using the observed proportion of hidden individuals to create an estimate of their number is likely to require an unacceptably large sample size for the estimate to have an acceptably small level of error.

As an alternative, Frank and Snijders [154] describe how snowball sampling may be used for this problem. In essence, the successive waves of the snowball sample provide information not unlike the re-observed ‘marked’ individuals in standard capture-recapture methods. Various estimators can be defined to exploit this information. We describe one such estimator to illustrate the underlying principle.

Let V be the set of all members of the hidden population, and let $G = (V, E)$ be a directed graph associated with that population, in which an arc from vertex i to vertex j indicates that, if asked, individual i would mention individual j as a member of the hidden population.³ Let G^* be a subgraph of G , where the vertices $V^* = V_0^* \cup V_1^*$ are obtained through a one-wave snowball sample, with the initial sample V_0^* selected through Bernoulli sampling from V with probability p_0 .

³ The practical concerns of trust, veracity, etc. associated with such questioning are themselves non-trivial issues of survey methodology, but we will ignore them here.

Our estimator of N_v will be derived using the method-of-moments. Let N, M_1 , and M_2 be random variables defined such that $N = |V_0^*|$ is the size of the initial sample, M_1 , the number of arcs among individuals in V_0^* , and M_2 , the number of arcs pointing from individuals in V_0^* to individuals in V_1^* . The expectations of these three quantities, with respect to the randomness in the selection of the initial sample V_0^* , are given by

$$\mathbb{E}(N) = \mathbb{E}\left(\sum_i Z_i\right) = N_v p_0 \quad (5.40)$$

$$\mathbb{E}(M_1) = \mathbb{E}\left(\sum_{i \neq j} Z_i Z_j A_{ij}\right) = (N_e - N_v) p_0^2 \quad (5.41)$$

$$\mathbb{E}(M_2) = \mathbb{E}\left(\sum_{i \neq j} Z_i (1 - Z_j) A_{ij}\right) = (N_e - N_v) p_0 (1 - p_0) , \quad (5.42)$$

where Z_i is a binary variable indicating whether vertex i is in the initial sample, and A_{ij} is the (i, j) -th entry of the adjacency matrix for G . Setting the left-hand sides of these equations equal to their observed counterparts, say n, m_1 , and m_2 , the first resulting equation yields the expression $N_v = n/p_0$, while the second two together yield the estimate $\hat{p}_0 = m_1/(m_1 + m_2)$ for the unknown value p_0 . Upon substitution we obtain

$$\hat{N}_v = n \left(\frac{m_1 + m_2}{m_1} \right) . \quad (5.43)$$

In other words, the number of individuals observed initially is inflated by an estimate of the sampling rate, where that estimate reflects the relative number of arcs from individuals in the initial sample that point inwards among themselves. It is interesting to compare the above estimator to that in (5.9).

Frank and Snijders [154] offer a number of other estimators of N_v in this spirit, using both design-based and model-based approaches. We note that the estimator in (5.43), while derived from a design-based perspective here, can also be derived from a model-based perspective. See Example 6.3 in Chapter 6.

Also proposed by these authors is an approach to estimating the variance of these estimators, based on the jackknife principle. Specifically, let $\hat{N}_{v,(i)}$ be the estimate of N_v obtained when vertex i is removed from the original sample V_0^* , as well as any vertex $j \in V_1^*$ having only i pointing to it among those vertices in V_0^* . Then the suggested estimate of variance is of the form

$$\hat{\mathbb{V}}_J(\hat{N}_v) = \frac{n-2}{2n} \sum_{i \in V_0^*} (\hat{N}_{v,(i)} - \hat{N}_{v,(.)})^2 , \quad (5.44)$$

where $\hat{N}_{v,(.)}$ is the average of the estimates $\hat{N}_{v,(i)}$ over $i \in V_0^*$. It should be noted that the factor $(n-2)/(2n)$ in (5.44) differs from the factor $(n-1)/n$ typically found in jackknife estimates of variance for iid observations. The change can be justified as accounting for the effects of the fact that (i) there are essentially n^2 ‘observations’ in

the $n \times n$ adjacency matrix, instead of just n , and (ii) these are not iid observations. See Frank and Snijders [154] for details. \square

Example 5.8 (How Large Is the Internet?). The Internet is massive – and still growing. It is therefore of interest, from both practical and intellectual perspectives, to ask just how large the Internet actually is. Of course, there are various ways in which we might wish to define the ‘size’ of the Internet and various measurement techniques that might be used to obtain relevant data. Here we will simplify the problem to one of inferring the number of vertices N_v in an undirected graph G , based on a set of measurements taken according to the traceroute design described in Section 5.3. Our treatment follows that of Viger et al. [388].

Interestingly, the estimation of N_v from traceroute measurements can be viewed as a species problem. Recall that under our idealization of traceroute sampling the observed vertex set V^* in the sampled graph G^* is the set of all vertices discovered on paths from n_s sources $S = \{s_1, \dots, s_{n_s}\}$ to n_t targets $T = \{t_1, \dots, t_{n_t}\}$. A given vertex may be discovered on more than one path. Hence, the vertex may be thought of as a species, and each time that vertex is observed on a path, this may be viewed as having seen a member of that species.

Viger et al. [388] propose a solution to this particular species problem using a method based on principles of sample re-use. The basic idea is to ask whether a given target vertex $t_j \in T$, if dropped from the study, would have been discovered on paths to the other remaining target vertices. One can easily determine what fraction of targets would not have been discovered in this manner, and that fraction can be used to inflate the observed number of vertices N_v^* upward to create an estimate \hat{N}_v of N_v , as we describe now.

Let $V_{(-j)}^*$ denote the number of vertices discovered on sampled paths to targets other than t_j , and define $\delta_j = I\{t_j \notin V_{(-j)}^*\}$ to be the indicator of the event that target t_j is not ‘discovered’ on sampled paths to any other target. Write the total number of such targets as $X = \sum_j \delta_j$.

Given a set of pre-selected source nodes (chosen either randomly or not), if the target nodes in T are chosen by simple random sampling without replacement from $V \setminus S$, the probability that target t_j is not discovered on the paths to other targets is given by

$$\mathbb{P}(\delta_j = 1 | V_{(-j)}^*) = \frac{N_v - N_{(-j)}^*}{N_v - n_s - n_t + 1}, \quad (5.45)$$

where $N_{(-j)}^* = |V_{(-j)}^*|$. Note that, by symmetry under simple random sampling, the expectation $\mathbb{E}(N_{(-j)}^*)$ is the same for all j . We denote this quantity by $\mathbb{E}(N_{(-)}^*)$ and, as a result, we obtain

$$\mathbb{E}(X) = \sum_{j=1}^{n_t} \mathbb{P}(\delta_j = 1 | V_{(-j)}^*) = \frac{n_t [N_v - \mathbb{E}(N_{(-)}^*)]}{N_v - n_s - n_t + 1}. \quad (5.46)$$

Rewriting this equation to isolate N_v , we have

$$N_v = \frac{n_t \mathbb{E}(N_{(-)}^*) - (n_s + n_t - 1) \mathbb{E}(X)}{n_t - \mathbb{E}(X)}. \quad (5.47)$$

Estimation of N_v is thus reduced to estimation of $\mathbb{E}(X)$ and $\mathbb{E}(N_{(-)}^*)$. Unbiased estimators of these two quantities are the value of X itself and the average of the $N_{(-j)}^*$, respectively. Under a slight variation of this idea and ignoring trivial terms and factors, Viger et al. [388] arrive at an estimator of the form

$$\hat{N}_v = (n_s + n_t) + \frac{N_v^* - (n_s + n_t)}{1 - w^*}, \quad (5.48)$$

where $w^* = X/(n_t + 1)$. In other words, N_v is estimated by counting the source and target vertices, and then adjusting upwards the number of additional vertices discovered through traceroute sampling, by a factor reflecting the average tendency of targets to be discovered by other paths. This estimator has the advantage over the simple plug-in estimator suggested just above in that even if $X = n_t$ it remains well defined.⁴

A simple estimate of the variance of \hat{N}_v in (5.48) is provided by the expression

$$\hat{\mathbb{V}}(\hat{N}_v) \approx \frac{(N_v^* - n_s - n_t)^2 w^*}{(1 - w^*)^3 n_t}, \quad (5.49)$$

which follows using a delta-method argument. See Viger et al. [388] for details.

Figure 5.8 illustrates the effectiveness of the estimator \hat{N}_v , as applied to a part of the Internet inferred by the Internet mapping project Skitter.⁵ This network had $N_v = 624,324$ vertices and $N_e = 1,191,525$ edges. Sampling was simulated using $n_s = 10$ sources and a target sampling density $\rho_t = n_t/N_v$ that ranged from 0.001 to 1.0. The results show that the estimator can be quite accurate for even low target sampling rates and is certainly a vast improvement over the observed number of nodes N_v^* . \square

The above are just two examples of group-size estimation problems arising in network contexts. It would appear that species problems are encountered often in this setting. For example, in Example 5.6, if we were forced to estimate the average degree, using techniques for vertex totals, under induced subgraph sampling, the dilemma faced there is essentially a species problem, in not knowing how many edges there are for each observed vertex $i \in V^*$. Similarly, in Example 5.8, estimation of the number of edges N_e is also a species problem, in that a given edge may be observed on multiple paths. Similarly, the estimation of the degree of a given node

⁴ The derivation of the estimator requires an assumption equivalent to saying that the number of vertices uniquely discovered on paths to any one or any pair of target vertices is relatively small compared to the overall number of vertices N_v^* in the sampled graph V^* . This assumption appears to be well motivated by empirical findings in the literature.

⁵ See <http://www.skitter.org>.

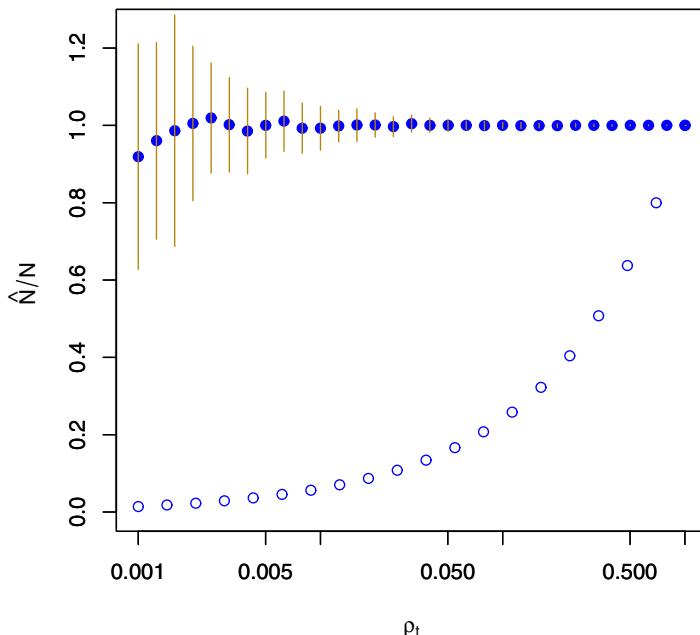


Fig. 5.8 Comparison of $\hat{N} = \hat{N}_v$ (filled circles) and $\hat{N} = N_v^*$ (open circles), as estimators of N_v , for various values of target sampling density ρ_t . Vertical bars indicate intervals of one standard deviation above and below each point, capturing the variation over multiple choices of source and target nodes. (Bars for the values N_v^* are too small to be visible.)

under traceroute sampling is a species problem too, as has been observed by Zhang [412].

5.6 Other Network Graph Estimation Problems

There are many other network graph estimation problems, some of which have been studied to an extent, and many of which have not. If a network graph G is sampled, rather than enumerated, and interest is in properties of G itself, then the computation of pretty much any characteristic $\eta(G)$ becomes a *de facto* estimation problem. However, not all are directly amenable to the types of methods we have described in this chapter, as the following example shows.

Example 5.9 (Estimation of Degree Frequency). A number of papers have demonstrated the effects that different network sampling designs can have in rendering

the observed degree distribution unrepresentative of the true underlying degree distribution, with particular interest in the issue of broad versus concentrated degree distributions. For instance, in Lakhina, Byers, Crovella, and Xie [243], results from numerical experiments are presented showing that extreme forms of traceroute sampling can actually induce a broad degree distribution in the sampled graph G^* when none exists in the true graph G . Analytical work (e.g., Clauset and Moore [91], Achlioptas, Clauset, Kempe, and Moore [3]) has confirmed and refined these findings, showing that even when G has an actual power-law degree distribution, the exponent α can be significantly under-estimated by exponents obtained from the degree distribution of G^* . Similar results have been demonstrated by Han et al. [192] in the case of sampling protein-protein interaction networks. See also Stumpf and Wiuf [373] and Stumpf, Wiuf, and May [374].

The degree distribution is susceptible to sampling effects in the same way that we have already seen the average degree to be; however, the design of accurate estimators for the average is substantially easier than for the full distribution. Let f_d and f_d^* be the true and observed frequencies of degree d nodes in G and G^* , respectively. Frank [153] shows using a combinatorial argument that, under induced subgraph sampling,

$$\mathbb{E}(f_d^*) = \sum_{d'=0}^{N_v-1} P(d, d') f_{d'} , \quad (5.50)$$

where

$$P(d, d') = \frac{\binom{d'}{d} \binom{N_v-1-d'}{n-1-d}}{\binom{N_v-1}{n-1}} . \quad (5.51)$$

In principle, pursuing a method-of-moments estimator, we can substitute f_d^* for $\mathbb{E}(f_d^*)$ on the left-hand side of (5.50), to obtain a system of equations for the vector (f_0, \dots, f_{N_v}) . But the resulting system will generally be under-determined, unless it is known *a priori* that the degree of G is bounded by a value $d_{max} \leq n$ and the equations are restricted accordingly. Furthermore, even in this setting, the solution to this system is not guaranteed to be non-negative. Finally, similar expressions for the variance of any such estimator would seem to be difficult to derive. \square

There are other estimation problems in network graph sampling that have received attention, such as work by Frank [150] on the estimation of the number of connected components in a graph. Overall, however, while the literature on sampling and estimation in network graphs is nontrivial in its size and breadth, much of this material pre-dates the explosion of interest in networks of the past 5 to 10 years, and as a result there are many open problems in this area. Examples include the analysis of non-traditional sampling designs (e.g., particularly adaptive designs), the estimation of quantities not easily expressed as totals (e.g., degree distribution exponents), and the incorporation of effects of sampling error and missingness.

On a final note, we point out that sampling and estimation are also being used in a proactive manner in the context of large network graphs, as a way of producing computationally efficient ‘approximations’ to quantities that, if computed for the full network graph, would be prohibitively expensive. Examples include the estimation

of centrality measures (e.g., Eppstein and Wang [133], Brandes and Pich [57]), and the detection of so-called ‘network motifs’ in work of Kashtan, Itzkovitz, Milo, and Alon [218]. We will encounter the latter topic in Section 6.2.4.3.

5.7 Additional Related Topics and Reading

The elements of the material presented here go back to foundations laid in classical sampling theory, for which there are many good references, such as Thompson [379]. In the specific context of network graphs, a substantial fraction of the work done in this area is due to Ove Frank, appearing in a series of papers throughout the 1970’s and 1980’s. See Frank [152] for a recent overview and extensive bibliography. For background on the species problem, see the survey of Bunge and Fitzpatrick [69].

Exercises

5.1. Recall that for estimating a total τ , the Horvitz-Thompson estimator $\hat{\tau}_\pi$ in (5.3) is unbiased, in the sense that its expected value over all possible samples S is equal to τ . Show that the corresponding variance estimate $\hat{\mathbb{V}}(\hat{\tau}_\pi)$ in (5.5) is an unbiased estimate of the true variance $\mathbb{V}(\hat{\tau}_\pi)$ in (5.5). That is, show that

$$\mathbb{E}(\hat{\mathbb{V}}(\hat{\tau}_\pi)) = \mathbb{V}(\hat{\tau}_\pi) .$$

(Hint: You will want to introduce binary random variables Z_1, \dots, Z_{N_u} , with $Z_i = 1$ indicating that unit i is in the sample S , as we did in Section 5.2 in showing the unbiasedness of $\hat{\tau}_\pi$ as an estimator of τ .)

5.2. Suppose that for every pair of vertices $i, j \in V$ in a graph G there is a unique shortest path, say $\mathcal{P}(i, j)$, between them. Then the betweenness centrality of a vertex $k \in V$ can be expressed as a total in the form

$$c_B(k) = \sum_{i \neq j \neq k \in V} I_{k \in \mathcal{P}(i, j)} .$$

Consider the problem of estimating $c_B(k)$, for fixed k .

- a. If you obtain a sample of n vertices V^* by simple random sampling without replacement from $V \setminus \{k\}$, show that for each pair $i \neq j \in V \setminus \{k\}$ the probability of inclusion takes the form

$$\pi_{ij} = \frac{n(n-1)}{(N_v - 1)(N_v - 2)} .$$

- b.** For a network graph G of your choice, conduct a simulation study to evaluate the performance of the estimator

$$\hat{c}_B(k) = \sum_{(i,j) \in V^{*(2)}} \frac{I_{k \in \mathcal{P}(i,j)}}{\pi_{ij}} ,$$

where the π_{ij} are as in part (a) and you assume the indicators $I_{k \in \mathcal{P}(i,j)}$ are observable for any pair $(i,j) \in V^{*(2)}$. Explore the characteristics of the estimators' performance as a function of the characteristics of the vertices k (e.g., degree, centrality c_B , etc.).

(Note: You may need to equip the edges e of your graph with weights w_e to enforce unique shortest paths. Weights that differ from one by small, random additive perturbations should be sufficient.)

- 5.3.** Verify the moment equations in (5.40), (5.41), and (5.42). Show that the method-of-moments estimator of the number of vertices N_v yielded by these equations is given by the expression in (5.43).

- 5.4.** Recall the discussion and references in the first paragraph of Example 5.9, regarding the extent to which the observed degree distributions $\{f_d^*\}$ of sampled graphs G^* have been found to deviate from the degree distributions $\{f_d\}$ of a true underlying network graph G .

- a.** Using a network graph G and a sampling design of your choice, explore this phenomenon using numerical simulation. To what extent do you find that the degree distributions of G^* and G differ? How sensitive are your results to tuneable aspects of your sampling design (e.g., the number n of vertices sampled)?
- b.** Under induced subgraph sampling, using your network graph G from part (a), explore the extent to which the proposed estimator deriving from (5.50) is feasible in your case. If it is not feasible – for example, if $d_{max} \gg n$ – can it be used to obtain accurate estimates of the distribution f_d up through some appropriately specified $d'_{max} \leq n$?

Chapter 6

Models for Network Graphs

In this chapter, we turn to the topic of modeling network graphs. We introduce a number of important classes of network graph models, and we illustrate some of the various statistical uses to which these models have been put, including the detection of network motifs, the evaluation of proposed network generative mechanisms, and the assessment of potential predictive factors of relational ties.

6.1 Introduction

So far in this book, the emphasis has been almost entirely focused upon methods, to the exclusion of modeling – methods for constructing network graphs, for analyzing their observed structure, and for obtaining sample-based estimates of unobserved or partially observed structure. For the remainder of this book, our focus will shift to the construction and use of models in the analysis of network data. And we begin, in this chapter, with the modeling of network graphs.

By a model for a network graph we mean effectively a collection

$$\{ \mathbb{P}_\theta(G), G \in \mathcal{G} : \theta \in \Theta \} , \quad (6.1)$$

where \mathcal{G} is a collection (or ‘ensemble’) of possible graphs, \mathbb{P}_θ is a probability distribution on \mathcal{G} , and θ is a vector of parameters, ranging over possible values in Θ . When convenient, we may often drop the explicit reference to θ and simply write \mathbb{P} for the probability function.¹

The richness of network graph modeling derives largely from how we choose to specify $\mathbb{P}(\cdot)$, with methods in the literature ranging from the simple to the complex. Some approaches, for example, simply let $\mathbb{P}(\cdot)$ be uniform on \mathcal{G} , but restrict \mathcal{G} to contain only those graphs G satisfying certain properties of interest. Other approaches induce $\mathbb{P}(\cdot)$ implicitly through the recurrent application of simple gen-

¹ More specifically, by a ‘model’ like that in (6.1) we have in mind the specification of a collection of probability spaces, indexed by θ ; but we will not work at that level of formality here.

erative mechanisms. Finally, still other approaches are designed to facilitate the inclusion of various endogenous and exogenous factors in the model and the statistical evaluation of their respective association with the topology of a graph G .

In practice, network graph models are used for a variety of purposes, many of which have been alluded to already in earlier chapters. These include the study of proposed mechanisms for the emergence of certain commonly observed properties in real-world networks (such as broad degree distributions or small-world effects) or the testing for ‘significance’ of a pre-defined characteristic(s) in a given network graph. Also, the use of network graph models allows for an alternative context to the design-based paradigm introduced in the previous chapter for defining and evaluating the accuracy of estimators of summaries of graph structure.

We will discuss four categories of network graph models in this chapter. First, in Section 6.2, we consider random graph models and several of their statistical applications. These models most literally capture the idea of a graph being drawn ‘at random’ from a collection \mathcal{G} . Second, in Section 6.3, we will present models for graphs possessing the small-world property observed in many real-world network graphs, as described in Section 4.3.2. Third, in Section 6.4, we will see a number of network growth models, wherein the focus is on the effect of simple, parameterized generative mechanisms. Fourth, and lastly, in Section 6.5, we turn our attention to exponential random graph models, which extend the idea of a statistical regression model to random graphs. A case study that examines the application of exponential random graph models to patterns of cooperation among corporate lawyers will be presented in Section 6.5.4. Some additional comments on the significant challenges remaining for network graph modeling are described briefly in Section 6.6.

6.2 Random Graph Models

The term *random graph model* typically is used to refer to a model specifying a collection \mathcal{G} and a uniform probability $\mathbb{P}(\cdot)$ over \mathcal{G} . Random graph models are arguably the most well-developed class of network graph models, mathematically speaking. This is no doubt due in part to the comparatively simpler nature of these models. However, it is precisely this nature that has allowed for, in contrast to many of the other classes of network graph models, the precise analytical characterization of many of the structural summary measures introduced in Chapter 4. And these characterizations in turn help provide insight into appropriate usage of such models for estimation and testing of structural characteristics in observed network graphs.

Example 6.1 (Model-Based Estimation in Network Graphs). In traditional statistical sampling theory, there are two main approaches to constructing estimates of population parameters from a sample: design-based and model-based. In the design-based approach, inference is based entirely on the random mechanism by which a subset of elements were selected from the population to create the sample. In the model-based approach, on the other hand, a model is given by the analyst that

specifies a relationship between the sample and the population. Model-based estimation strategies (e.g., least-squares, method-of-moments, maximum-likelihood, etc.) are then typically used for constructing estimators of population parameters. In more recent decades, the distinction between these two approaches has become more blurred, with elements of both design- and model-based strategies often used profitably in conjunction with each other. See the book by Särndal, Swensson, and Wretman [339], for example.

Consider the task of estimating a given characteristic $\eta(G)$ of a network graph G , based on a sampled version of that graph, G^* . In Chapter 5, we presented almost exclusively a design-based perspective on this problem. Alternatively, we might augment this perspective to include a model-based component as well, wherein G is assumed to have been generated uniformly at random from a collection \mathcal{G} , prior to our obtaining G^* . Inference on η then should incorporate both randomness due to selection of G from \mathcal{G} and randomness due to sampling G^* from G . We will illustrate this approach for estimating the size of a ‘hidden’ population, using one of the models introduced in this section, later in Section 6.2.4. \square

Example 6.2 (Assessing Significance in Network Graphs). Suppose that we have a graph derived from observations of some sort (i.e., not necessarily through a formal network sampling mechanism), which we will denote as G^{obs} here, and that we are interested in some structural characteristic, say $\eta(\cdot)$. In particular, suppose that we are interested in assessing whether the value $\eta(G^{obs})$ is ‘significant,’ in the sense of being somehow unusual or unexpected.

Of course, such notions of significance must be defined in comparison to an appropriate frame of reference. Random graph models are often used in setting up such comparisons. A collection of random graphs \mathcal{G} is defined, and the value $\eta(G^{obs})$ is compared to the collection of values $\{\eta(G) : G \in \mathcal{G}\}$. If $\eta(G^{obs})$ is judged to be extreme with respect to this collection, then that is taken as evidence that G^{obs} is unusual in having this value. More formally, a random graph model is used to create a reference distribution which, under the accompanying assumption of uniform likelihood of elements in \mathcal{G} , takes the form

$$\mathbb{P}_{\eta, \mathcal{G}}(t) = \frac{\#\{G \in \mathcal{G} : \eta(G) \leq t\}}{|\mathcal{G}|} . \quad (6.2)$$

If $\eta(G^{obs})$ is found to be sufficiently unlikely under this distribution, this is taken as evidence against the hypothesis that G^{obs} is a uniform draw from \mathcal{G} .

How best to choose \mathcal{G} is a practical issue of some importance, since it can have a direct impact on the relevance of the results of such procedures, as we illustrate later in Section 6.2.4. Also important is the fact that, outside of the contexts with both very small network graphs and very simple random graph models \mathcal{G} , we cannot generally hope to be able to explicitly enumerate all of the elements of \mathcal{G} , and therefore, we cannot expect to be able to calculate the probabilities in (6.2) exactly. Rather, approximations to these probabilities are used. There are analytical approximations for certain comparatively simple cases; see Wasserman and Faust [393, Ch.

13.5], for example. But for most other cases of interest it is necessary to employ Monte Carlo simulation methods – such as those we describe in Section 6.2.3 – to obtain numerical approximations. \square

In this section, we present classical random graph models and generalized random graph models, including an overview of various of the most relevant properties of each. We then describe algorithms for simulating random draws of such graphs from their underlying collections \mathcal{G} , which is critical for their use in applications. Finally, we illustrate the practical application of random graph models in the contexts of model-based estimation and assessment of significance, as introduced in Examples 6.1 and 6.2. Our focus here will be primarily on graphs G that are undirected and simple (i.e., no loops or multi-edges). Extensions of the ideas and results we discuss to, for example, directed graphs are generally straightforward and may be found in many of the references cited.

6.2.1 Classical Random Graph Models

The classical theory of random graph models, as established in a series of seminal papers by Erdős and Rényi [134, 135, 136], rests upon a simple model that places equal probability on all graphs of a given order and size. Specifically, their model specifies a collection \mathcal{G}_{N_v, N_e} of all graphs $G = (V, E)$ with $|V| = N_v$ and $|E| = N_e$, and assigns probability $\mathbb{P}(G) = \binom{N}{N_e}^{-1}$ to each $G \in \mathcal{G}_{N_v, N_e}$, where $N = \binom{N_v}{2}$ is the total number of distinct vertex pairs.² The key contribution of Erdős and Rényi was to develop a foundation of formal probabilistic results concerning the characteristics of graphs G drawn randomly from \mathcal{G}_{N_v, N_e} with respect to this $\mathbb{P}(\cdot)$.

It is more common, however, to see such results stated for a variant of \mathcal{G}_{N_v, N_e} , first suggested by Gilbert [173] at approximately the same time. In this formulation, a collection $\mathcal{G}_{N_v, p}$ is defined to consist of all graphs G of order N_v that may be obtained by assigning an edge independently to each pair of distinct vertices with probability $p \in (0, 1)$. When p is an appropriately defined function of N_v , and $N_e \sim pN_v$, these two classes of models are essentially equivalent for large N_v . We shall refer primarily to the version of Gilbert when discussing ‘classical random graphs’ in the remainder of this section.

Classical random graph models are by far the most thoroughly studied in the literature. We mention just a handful of key properties here – primarily those associated with certain of the characteristics mentioned in Chapter 4. A thorough coverage of results of this nature may be found in the book by Bollobás [43].

Note that we cannot expect that a random graph be connected *per se*. However, it is possible to make precise statements regarding the level of connectivity likely

² Following convention in the literature, we will frequently abuse notation by annotating our collection \mathcal{G} with additional parameters associated with the probability distribution \mathbb{P} accompanying \mathcal{G} , as in \mathcal{G}_{N_v, N_e} , and refer to that interchangeably as the ‘collection’ or the ‘model,’ as context demands, rather than writing out the full model specification corresponding to (6.1).

in a randomly generated $G \in \mathcal{G}_{N_v, p}$, as a function of the relation between N_v and p . For example, let $p = c/N_v$, for $c > 0$. Then, if $c > 1$, with high probability³ G will have a single connected component consisting of $\alpha_c N_v$ vertices, for some constant $\alpha_c > 0$ depending on c , with the remaining components having only on the order of $O(\log N_v)$ vertices. On the other hand, if $c < 1$, then all components will have on the order of $O(\log N_v)$ vertices, with high probability – that is, G will consist entirely of a large number of very small, separate components. In the case of $c > 1$, the larger component is typically called the *giant component*, as was mentioned in Section 4.3.2. Note too that, under the stated conditions, the expected density of the graph G is only $p \sim N_v^{-1}$, and so G is likely to be sparse for large N_v .

Regarding the degree distribution, it is known that classical random graphs have distributions that are concentrated, with exponentially decaying tails. More precisely, if we write $f_d(G)$ as the (random) proportion of vertices with degree d in G , then for $d = 0, 1, 2, \dots$ and $p = c/N_v$, with $c > 0$, it can be shown that with high probability

$$(1 - \varepsilon) \frac{c^d e^{-c}}{d!} \leq f_d(G) \leq (1 + \varepsilon) \frac{c^d e^{-c}}{d!}, \quad (6.3)$$

for any given $\varepsilon > 0$. This result states that, for large N_v , G will have a degree distribution that is like a Poisson distribution with mean c . That this should be true is somewhat intuitive, as it is easy to see that each vertex $v \in V$ has its degree d_v identically distributed as a binomial random variable,⁴ with parameters $N_v - 1$ and p , which suggests the relevance of a Poisson approximation to this binomial with parameter $\mathbb{E}[d_v] = p(N_v - 1) \sim c$.

So classical random graphs do not have the broad degree distribution observed in many large-scale real-world networks. Nor, it turns out, do they display much clustering, under the conditions corresponding to the results above. For example, recall that the clustering coefficient $\text{cl}_T(G)$ measures the fraction of transitive triples in G that close to form a triangle, which may be interpreted as the probability that two neighbors of a randomly chosen vertex v in G are themselves linked. But by construction, this probability is just p , and as we have seen above, the interesting cases involve $p \sim N_v^{-1}$, which will tend to zero as N_v grows.

On the other hand, graphs in $\mathcal{G}_{N_v, p}$ do possess the small-world property, under the conditions above, in that the diameter can be shown to vary like $O(\log N_v)$, with high probability, as $N_v \rightarrow \infty$.

³ For this result, and a number of others we summarize in this section, variants may be stated with assurances stronger than ‘with high probability,’ but at the cost of additional technical details and structure.

⁴ And in fact, it may be shown that these binomials are quite close to being independent. See McKay and Wormald [274].

6.2.2 Generalized Random Graph Models

The formulation of Erdős and Rényi can be generalized in a straightforward manner. Specifically, the basic recipe is to (a) define a collection of graphs \mathcal{G} consisting of all graphs of a fixed order N_v that possess a given characteristic(s), and then (b) assign equal probability to each of the graphs $G \in \mathcal{G}$. In the Erdős-Rényi model, for example, the condition is simply that the size of the graphs G be equal to some fixed N_e .

Beyond Erdős-Rényi, the most commonly chosen condition is that of a fixed degree sequence. That is, \mathcal{G} is defined to be the collection of all graphs G with a pre-specified degree sequence, which we will write here as $\{d_{(1)}, \dots, d_{(N_v)}\}$, in ordered form. Note that, with N_v specified as well, the number of edges N_e is fixed under this model, due to the relation $\bar{d} = 2N_e/N_v$. It is also useful to note that, conversely, for fixed N_v and N_e , and a chosen degree sequence with average degree \bar{d} matching this choice, the resulting collection \mathcal{G} is strictly contained within \mathcal{G}_{N_v, N_e} . So the addition of an assumed form for the degree sequence is in this case equivalent to specifying our model through a conditional distribution on the original collection \mathcal{G}_{N_v, N_e} . In either case, all other characteristics are free to vary to the extent allowed by the chosen degree sequence.

A number of authors have conducted mathematical studies of the properties of generalized random graph models with fixed degree sequence. There are minor differences in the manner in which these authors specify their random graph models, with different formulations better facilitating the use of different mathematical tools. Nonetheless, all of these models are essentially equivalent asymptotically.

Molloy and Reed [282, 283] study random graphs whose degree sequence conforms approximately to a pre-specified degree distribution $\{f_d\}$ and is sparse and well behaved in a certain well-defined sense. They provide a simple condition, on a function of the first two moments of the degree distribution, under which their generalized random graphs G have a giant component, with high probability. If the condition is not satisfied, then, analogous to the classical case, the graph consists of a large collection of small, separate subgraphs. In addition, they show that the giant component, when it exists, should contain a number of vertices growing like αN_v , for fixed constant α , with high probability, similar to the classical case.

Aiello, Chung, and Lu [5] have built upon these results, under a similar model formulation, in the special case where the degree distribution $\{f_d\}$ has a power-law form (i.e., $f_d = Cd^{-\alpha}$). In particular, they provide specific conditions for the emergence of a giant component and they characterize the size of that component and the remaining components, as a function of the parameters α and C . Additionally, Chung and Lu [87] have characterized the diameter and the average distance in a related power-law random graph model, with an eye towards the phenomena of small-world behavior. They have shown, for example, that for degree distribution exponents $\alpha \in (2, 3)$, a range cited often in practice,⁵ the diameter is of $O(\log N_v)$

⁵ Recall, for example, our analysis of the Internet router network in Example 4.2, and the evidence for an exponent α slightly under 3.0.

and the average distance is of order $O(\log \log N_v)$, with high probability under mild conditions.

Finally, using a slightly different modeling framework, Newman, Strogatz, and Watts [299] analyze much the same characteristics of generalized random graph models, using semi-formal arguments, and produce similar results. Later, in the same context, Newman [296, Sec. IV.B] argues that, under certain conditions, for power-law degree distributions with $\alpha \in (7/3, 3)$, the clustering coefficient cl_T should behave like $N_v^{-\beta}$, where $\beta = (3\alpha - 7)/(\alpha - 1)$. This result suggests that, for power-law generalized random graphs with exponent α in this range, the clustering coefficient tends to zero with increasing N_v , but slower than the rate N_v^{-1} observed in the classical setting.

6.2.3 Simulating Random Graph Models

The mathematical study of random graph models yields various fundamental insights. However, in order to use these models in statistical applications it is generally necessary to be able to efficiently simulate the sampling of such random graphs using a computer. For some models it is actually possible to produce samples in linear time; for others, it appears that Markov chain Monte Carlo (MCMC) methods are the only realistic alternative.

6.2.3.1 Simulating Classical Random Graphs

Consider the problem of simulating classical random graph models. In the model $\mathcal{G}_{N_v, p}$, for example, the definition of the model class itself suggests a simple algorithm. That is, in principle, we can simply generate $\binom{N_v}{2} \sim N_v^2/2$ independent Bernoulli random variables, each with probability of success p , and assign edges to those pairs of vertices whose corresponding variable yields a success. Recall, however, that when $p \sim N_v^{-1}$, the graphs emerging from $\mathcal{G}_{N_v, p}$ are expected to be sparse. Therefore, an overwhelming majority of these variables will not yield a success, and the algorithm is inefficient.

Batagelj and Brandes [23], exploiting this fact, offer an algorithm that runs in $O(N_v + N_e)$ time, which in the case of sparse graphs will be significantly faster than the nominal $O(N_v^2)$. The key insight underlying their algorithm is that it is possible to simply skip potential edges in a manner that is nevertheless consistent with the model. In particular, for each vertex $v \in V$, the other $N_v - 1$ vertices are ordered, and an index, say j , is drawn according to a geometric random variable⁶ with parameter p . The first $j - 1$ potential edges incident to this vertex are not formed, but the j -th one is. If $j < N_v - 1$, so that there remain potential edges, the process is repeated

⁶ The waiting time for a success in a sequence of independent and identically distributed Bernoulli(p) random variables is distributed as a geometric random variable with parameter p .

with those that remain. A formal statement of the algorithm is provided in Batagelj and Brandes [23].

For the model \mathcal{G}_{N_v, N_e} , recall that the goal is to sample graphs G uniformly from all those with N_v vertices and N_e edges. Batagelj and Brandes [23] observe that, while skipping can again be used, it will be more cumbersome and costly to do so, since the presence or absence of edges are now correlated with each other. More efficient is to simply draw candidate edges one by one, uniformly at random from the set of all $\binom{N_v}{2}$ possible edges, rejecting edges already chosen, until the pre-specified number of distinct edges N_e have been obtained. This algorithm has a running time that is $O(N_v + N_e)$ in expectation.

6.2.3.2 Simulating Generalized Random Graphs

Now let us consider sampling according to a generalized random graph model, where the problem is more challenging than in the classical setting, due to the increased complexity of the specified constraint(s). To illustrate, we will focus our discussion here upon the case in which the collection \mathcal{G} is determined by constraining the degree sequence to be fixed. We offer a few brief comments regarding more general cases at the end of this section.

Two standard approaches to simulating random graphs with fixed degree sequence are *matching algorithms* and *switching algorithms*. The basic matching algorithm is simple and can be described as follows. Given an ordered degree sequence $\{d_{(1)}, \dots, d_{(N_v)}\}$, create a list containing, for each $i = 1, \dots, N_v$, a total of $d_{(i)}$ copies of the corresponding vertex. Then, randomly choose pairs of elements from this list, removing each pair once chosen. The result will be a set of edges for a candidate graph G . If this set contains multi-edges or loops, the corresponding graph is a multi-graph, and so the set is discarded and the procedure repeated. Under appropriate conditions on the degree sequence, it can be argued that this algorithm will generate graphs from \mathcal{G} with equal probability. See Molloy and Reed [282].

Unfortunately, in discarding ineligible candidate graphs, this algorithm can be quite inefficient. Indeed, it can be especially inefficient when the specified degree sequence is highly skewed, as with some of the examples seen in Section 4.2.1.1, because we can expect fairly frequently to see vertices of high degree matched with one another more than once. In fact, the frequency of such matchings can prevent the algorithm from completing with any sort of regularity. A common solution to this problem is to monitor the pairs of vertices being selected and, if a candidate pair matches one already removed from the list, it is rejected and another candidate selected instead. Note that this modification will necessarily introduce bias into the sampling, and the graphs G thus generated will no longer correspond to a strictly uniform sampling. Milo et al. [279], however, report a number of examples where this bias seems to be sufficiently small for practical purposes.

Alternatively, we can instead sample so as to avoid repeating existing matches in the first place. Chen, Diaconis, Holmes, and Liu [81], for example, introduce a method that is guaranteed to avoid repeated matches, due to a sequential assign-

ment of edges to each vertex in turn. Moreover, their method assigns these edges in a particularly efficient manner, through the use of an appropriately chosen non-uniform sampling scheme. The overall algorithm appears to produce nearly uniform samples in practice. The applications in their paper, however, are relatively small in scale, and it remains to be seen the extent to which their algorithm can successfully scale to large, sparse graphs.⁷

In contrast to matching algorithms, switching algorithms (also called *rewiring algorithms*) begin with a graph that has the prescribed degree sequence, and proceed to modify the connectivity of that graph through a succession of simple changes applied to randomly chosen subsets of edges. The term ‘switching’ derives from the nature of the basic change. Specifically, at each step, a pair of edges in the current graph, say $e_1 = \{u_1, v_1\}$ and $e_2 = \{u_2, v_2\}$, are randomly selected and replaced by the new edges $\{u_1, v_2\}$ and $\{u_2, v_1\}$, assuming neither of the latter already exists. That is, at each step of the algorithm it is proposed that a pair of randomly selected edges be ‘switched’; but if either of the new edges that would result already exist, the proposed switch is abandoned.

This type of algorithm falls within the realm of MCMC algorithms. In practice, it is typical to let the algorithm run for some time before beginning to collect sample graphs G , so as to let the limiting behavior of the underlying Markov chain ‘kick in.’ But, as is often the case with MCMC methods, there is currently no well-developed theory to indicate just how long of a preliminary period is necessary. Milo et al. [279], however, cite empirical evidence to suggest a factor of $100N_e$ can be more than sufficient.

In order to ensure that the algorithm asymptotically yields strictly uniform sampling from \mathcal{G} , it is necessary that certain formal conditions be satisfied.⁸ Rao, Jana, and Bandyopadhyay [322] describe such conditions for the basic switching algorithm described above and show that, in addition to the switching of pairs of edges, it is also necessary to allow for the possibility of a small number of three-edge exchanges.

MCMC methods, extending the basic switching concept described above, are also popular for generating generalized random graphs uniformly from other types of collections \mathcal{G} , in which additional characteristics beyond the degree sequence are constrained. This is largely due no doubt to the generality of the MCMC formalism, and the relative ease with which it may be implemented. Snijders [359], Roberts [326], and more recently, McDonald, Smith, and Forster [273], for example, offer MCMC algorithms for a number of different cases, such as where the graphs G

⁷ The method of Chen, Diaconis, Holmes, and Liu [81] is actually developed for uniformly sampling $r \times c$ matrices \mathbf{M} of non-negative integers with fixed marginal totals, such as can be needed in the analysis of contingency tables. However, it is immediately applicable to the problem of uniformly sampling generalized random graphs of fixed degree sequence, since the collection \mathcal{G} can be linked in one-to-one correspondence with the collection \mathcal{A} of all $N_v \times N_v$ binary adjacency matrices A with row and column sums matching the specified degree sequence. This duality has been exploited by a number of authors in this area. See Chen, Diaconis, Holmes, and Liu [81] for citations.

⁸ In particular, the corresponding Markov chain should be irreducible on \mathcal{G} and ergodic, so that convergence to the limiting uniform stationary distribution is guaranteed by the theory.

are directed and constrained to have both a fixed pair of in- and out-degree sequences and a fixed number of mutual arcs. It should be noted, however, that development of the corresponding theory, verifying the assumptions underlying Markov chain convergence, currently appears to lag far behind the pace of algorithm development.

6.2.4 Statistical Application of Random Graph Models

There are numerous ways in which random graph models can be used in statistical practice. We illustrate the possibilities here by describing their usage for model-based estimation of and assessment of significance of network graph characteristics.

6.2.4.1 Model-Based Estimation in Network Graphs

Recall the discussion in Example 6.1, where we revisited the problem of estimating a characteristic $\eta(G)$ of a network graph G , based on a sampled graph G^* . There were two aspects to the framework we described, with one aspect specifying the sampling mechanism used in producing G^* from G and the other specifying that G be the result of a uniform draw from a collection \mathcal{G} . The following example illustrates the use of such a framework, in the context of the problem of estimating the size of a hidden network population, as introduced in Example 5.7 of Chapter 5.

Example 6.3 (Estimating the Size of a ‘Hidden Population’ (Continued)). We describe a model-based derivation of the estimator in (5.43), again using a method-of-moments approach, and also due to Frank and Snijders [154]. Recall that the population graph G was a digraph, with vertices corresponding to a population of hidden individuals, and arcs, to the pattern of references among individuals if asked to identify other hidden individuals than themselves. The goal was to estimate the size of this population (i.e., $N_v = |V|$).

Suppose now that we model G as being from a collection \mathcal{G} of random graphs, where each graph is constructed from an initial vertex set V by independently adding arcs between distinct pairs of vertices with probability $p = p_{\mathcal{G}}$. This is a directed version of the random graph model $\mathcal{G}_{N_v, p}$ introduced earlier in this chapter. Then, given G , we again assume that a subgraph G^* is obtained through a one-wave snowball sample, with the initial sample of vertices V_0^* selected through Bernoulli sampling on V with probability $p = p_0$.

As before, let $N = |V_0^*|$ be the size of the initial sample, M_1 the number of arcs among individuals in V_0^* , and M_2 the number of arcs pointing from individuals in V_0^* to individuals in V_1^* , where V_1^* is the set of individuals discovered in the first wave of sampling (i.e., the set of individuals discovered after the initial sample V_0^*). Under the assumptions on the model and sampling, it can be shown that

$$\mathbb{E}(N) = N_v p_0 \quad (6.4)$$

$$\mathbb{E}(M_1) = N_v (N_v - 1) p_0^2 p_{\mathcal{G}} \quad (6.5)$$

$$\mathbb{E}(M_2) = N_v(N_v - 1)p_0(1 - p_0)p_{\mathcal{G}}. \quad (6.6)$$

Compare these expressions to those in equations (5.40) through (5.42).

After setting the right-hand sides of each of the above equations equal to the observed values n, m_1 , and m_2 of the arguments of the left-hand sides, some algebra yields the method-of-moments solutions

$$\hat{p}_0 = m_1/(m_1 + m_2) \quad (6.7)$$

$$\hat{p}_{\mathcal{G}} = m_1(m_1 + m_2)/n[(n - 1)m_1 + nm_2] \quad (6.8)$$

$$\hat{N}_v = n(m_1 + m_2)/m_1 \quad (6.9)$$

for the parameters $p_0, p_{\mathcal{G}}$, and N_v . Note that these are the same estimates of p_0 and N_v as those in Example 5.7, derived from (5.40) through (5.42).

Frank and Snijders [154] also provide another estimator using this model, which differs from that just derived, based on maximization of a conditional likelihood. \square

Such usage of classical random graph models is traditionally not uncommon in the social networks literature. On a related note, we point out that there are other ways to incorporate models into the inference of population graph parameters η . For example, instead of specifying a model for the graph G itself, we might instead specify a model directly for η . We have in fact already touched briefly on one such use of models in Section 4.2.1.1, in discussing methods to characterize the shape of a degree distribution. There, for certain broad degree distributions we saw that we might entertain simple power-law models like that in (4.1), or in particular, the linearized version in (4.2). Recall, however, that it was found that the fitting of such models to estimate power-law exponents can potentially be a rather delicate affair. Additionally, such approaches do not themselves inherently address effects of network sampling bias, like those mentioned in Example 5.9.

6.2.4.2 Assessing Significance in Network Graphs

Recall the discussion in Example 6.2, where we proposed to assess the significance of the value $\eta(G^{obs})$ of a network characteristic η for an observed graph G^{obs} , by constructing a reference distribution $\mathbb{P}_{\eta, \mathcal{G}}$ and examining how likely $\eta(G^{obs})$ is under this distribution. As we remarked there, an important practical issue is determining which \mathcal{G} to use.

Suppose, for example, that the value $\eta(G^{obs})$ of interest is the number of distinct triangles in G^{obs} . If the graphs in \mathcal{G} are not restricted to have the same number of edges N_e^{obs} as G^{obs} , then an apparent significance in the number of triangles in G^{obs} could result simply from comparing against graphs G with too few or too many edges to even possibly generate triangles at a similar order of magnitude. While this particular flaw can be easily remedied in this case, the basic point obviously stands. In practice, it is common to control for the degree sequence observed in

G^{obs} . However, at times there may arise other factors for which we might wish to control. For example, if we knew that certain groups were present within the vertex set V^{obs} , we might wish to maintain the number of edges between and within each group. We illustrate some of these various issues through the following example.

Example 6.4 (Assessing Clustering in the Karate Network). We saw in Section 4.3.1 that the karate club network has a clustering coefficient $\text{cl}_T(G) = 0.2557$. With no other comparable sources of information, such as networks of similar clubs or dynamic snapshots of the same club, it is difficult to say *a priori* whether or not this value is in any sense unusual or unexpected. Therefore, we instead use random graphs to establish two abstract frames of reference, in the manner described above. For the first, we define \mathcal{G} to be the collection of random graphs of the same order $N_v = 34$ and size $N_e = 78$ as the karate club network, while for the second, we add the further restriction that the elements of \mathcal{G} have the same degree distribution as in the original.

For \mathcal{G} defined with respect to a fixed number of edges, there are $\binom{34}{2} \approx 8.4 \times 10^{96}$ elements G . When \mathcal{G} is instead defined through specifying the degree distribution, the total number of elements is much less, but still quite large. It is in fact a difficult problem in and of itself to evaluate this number, and essentially infeasible for graphs of order any larger than about $N_v = 10$. Discussion on this point may be found in Snijders [359] or Chen, Diaconis, Holmes, and Liu [81], for example. Therefore, we instead approximate the distribution in (6.2) by simulating⁹ the uniform sampling of 10,000 random graphs G from \mathcal{G} , for each of these two choices of \mathcal{G} , and calculating $\eta(G) = \text{cl}_T(G)$ for each graph sampled.

Histograms showing the resulting distributions may be found in Figure 6.1. We see that under both random graph models it is quite unlikely to see a clustering coefficient as high as the observed value of 0.2557. In the first case, when the number of edges was fixed, only 3 of our 10,000 samples resulted in a graph with higher clustering coefficient, whereas in the second case, when the degree distribution was fixed, all of the samples resulted in lower clustering coefficient. We therefore have strong evidence to reject the hypothesis that the karate club network can be viewed as a uniform sample under either random graph model. As a result, we conclude that the network graph shows markedly greater transitivity than random graphs of comparable magnitude (i.e., with respect to order and size) or connectivity (i.e., with respect to degree distribution).

We note that it is also of some interest to compare the two histograms in Figure 6.1. The distribution in the first case is noticeably more broad than that in the second, and shifted slightly more to the left. This indicates that further conditioning on degree distribution, rather than just the number of edges, has the effect of restricting to graphs for which the clustering coefficient is slightly higher and also more concentrated. In addition, note the suggestion of bimodal structure in the bottom histogram. Visual examination of sampled graphs from each of the two modes

⁹ Simulations were conducted using Monte Carlo algorithms, implemented in the R package `statnet`. See <http://csde.washington.edu/statnet>.

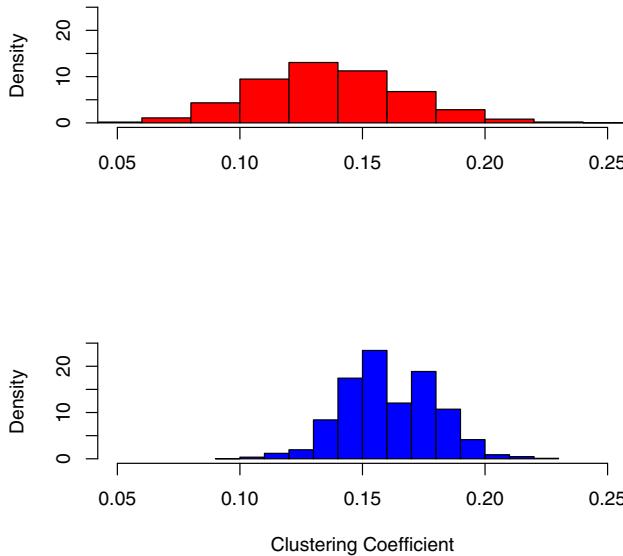


Fig. 6.1 Histograms of clustering coefficients $\text{cl}_T(G)$ for random graphs of order $N_v = 34$, generated uniformly from those with the same number of edges N_e (top, in red) and the same degree distribution (bottom, in blue) as in the karate club network.

showed that, roughly speaking, those in the right-hand mode tended to often be characterized by two large and somewhat distinct clusters, as in the original network (see Figure 1.2), while those in the left-hand mode appeared to have more diffuse clusters or even just one large cluster. Finally, also worth mentioning is the observation that, due to the restriction placed on the network through conditioning on degree, coupled with the invariance of cl_T under isomorphism, the effective size of the sample space becomes quite small. In the 10,000 trials run, there were only 25 different values of cl_T ; in fact, over 99% of the mass of the distribution rested on just 17 distinct values. □

6.2.4.3 Detecting Network Motifs

A related use of random graph models is in the detection of *network motifs*, defined by Alon and colleagues [218, 278] to be small subgraphs occurring far more frequently in a given network than in comparable random graphs. The motivation for this interest in subgraphs is the idea that many large, complex networks may perhaps be constructed – at least in part – of smaller, comparatively simple ‘building

blocks.' Network motif detection seeks to identify possible subgraph configurations of this nature.

In Milo et al. [278], each of the elements N_i of the vector (N_1, \dots, N_{L_k}) are calculated, where N_i is the number of occurrences of the i -th of L_k possible types of k -vertex subgraphs. Similarly, in Kashtan, Itzkovitz, Milo, and Alon [218] the relative proportions

$$F_i = \frac{N_i}{\sum_{i'=1}^{L_k} N_{i'}} \quad (6.10)$$

are used. Then, analogous to what was described in the previous section, each value of N_i (or F_i) is compared to an appropriate reference distribution $\mathbb{P}_{N_i, \mathcal{G}}$ (or $\mathbb{P}_{F_i, \mathcal{G}}$) resulting from calculation of the same subgraph count (or relative count) statistic for all G in some collection \mathcal{G} . Subgraphs for whom the value N_i (or F_i) is found to be extreme are declared to be network motifs.

Example 6.5 (Detecting Motifs in the AIDS Blog Network). Consider the AIDS blog network in Figure 1.4 of Chapter 1. This directed graph represents $N_e = 183$ web-link-based citations (excluding loops and multi-edges) among $N_v = 146$ bloggers in a group focused on topics relating to AIDS. Although the network is not large, a simple glance at the figure is sufficient to see that it would be non-trivial to detect small, recurrent subgraph patterns by eye alone. The network motif methodology just described allows for an automated and principled approach to this task.

A total of 10,000 random digraphs were generated using a switching algorithm, with in- and out-degree sequences, and also mutual edges, fixed to equal those of the original AIDS blog network.¹⁰ Evidence for motifs of size $k = 3$ and 4 was examined, based on the appropriate reference distributions. Subgraphs for which the probability, under this random graph model, of seeing N_i as large or larger (i.e., the p -value) was 0.01 or smaller were declared to be motifs. One three-vertex subgraph and four four-vertex subgraphs were discovered using this criterion. These are displayed in Figures 6.2 and 6.3, respectively.

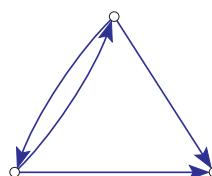


Fig. 6.2 Three-vertex motif discovered for the AIDS blog network.

¹⁰ Calculations were done using the mfinder1.2 software. See <http://www.weizmann.ac.il/mcb/UriAlon/groupNetworkMotifSW.html>.

There were a total of 2,442 three-vertex subgraphs found in our network graph. The three-vertex motif that we discovered occurred 15 times in this network, but only somewhat less than 7 times on average among the random graphs sampled (mean 6.7, with standard deviation 2.2). It represents two bloggers pointing to each other, with each in turn also pointing to a third blogger. That is, they share a mutual edge, and in addition both point to a common ‘authority.’ A total of 27,887 four-vertex subgraphs are contained in the network graph, and the four-vertex motifs were found to occur 704, 37, 13, and 5 times, respectively, based on the same criteria as described above for the three-vertex case. The first four-vertex motif represents one blogger pointing to another, who in turn points to two others. The other three four-vertex motifs are a bit more complicated in nature, but note that each of them contains within it one or more versions of the three-vertex motif and also the first four-vertex motif. So there is an argument to be made that the three-vertex motif and the first four-vertex motif are the most likely candidates to serve as elementary ‘building block’ structures.

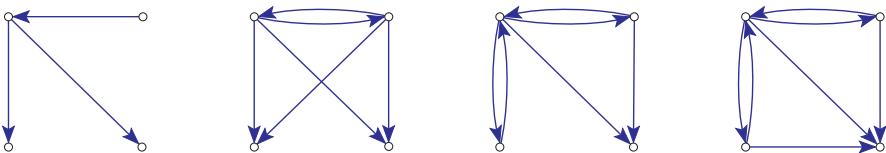


Fig. 6.3 Four-vertex motifs discovered for the AIDS blog network.

On a cautionary note, however, we point out that with the network graph as small as it is here, and the blog citations as inter-woven as they are, most of the individual copies of any of these motifs are overlapping with other versions of themselves. In larger networks, it is common to also require that a subgraph occur at some minimum number of disjoint locations in order to be declared a motif. \square

Network motif detection is based on the basic principles of assessing significance we already discussed earlier. The new wrinkle here is primarily in the decision to focus on arbitrary subgraphs and their frequency of occurrence,¹¹ coupled with the determination to do this in large networks (e.g., $N_e > 100,000$) or for motifs defined by relatively large subgraphs (e.g., $k \geq 5$). With these larger problems come substantial computational challenges. For a given choice of k , we are nominally required to count the number N_i of each possible k -vertex subgraph, in both the observed network G^{obs} and each (or at least a large random subset) of the networks G in the collection \mathcal{G} . But the number L_k of possible motifs grows quite quickly in k . For example, in a directed graph G , there are $L_3 = 13$ distinct types of connected,

¹¹ We note that a precursor in the social network literature is the use of the so-called *triad census* in testing for group structure. See Wasserman and Faust [393, Sec. 14.4].

three-vertex subgraphs, $L_4 = 199$ such four-vertex subgraphs, and a rather daunting $L_5 = 9364$ such five-vertex subgraphs.

Graph sampling techniques, like those encountered in Chapter 5, are appealing for use in overcoming this computational challenge. Specifically, if k -vertex subgraphs H are sampled in some fashion, then an unbiased estimate of the total number N_i of a given subgraph type is just

$$\hat{N}_i = \sum_{H \text{ of type } i} \pi_H^{-1} , \quad (6.11)$$

where π_H is the inclusion probability for H . Natural (although biased) estimates \hat{F}_i of the corresponding relative frequencies F_i are obtained through direct substitution of (6.11) in (6.10).

Kashtan, Itzkovitz, Milo, and Alon [218] propose the following sampling method. Given a network graph G , we first randomly select a single edge. Then, using the idea of link-tracing introduced in Section 5.3.3, we follow a new edge to a neighboring vertex, where that edge is selected randomly from among those in G incident to the two vertices defining the first edge. We then continue in this fashion, where at the m -th stage we follow an edge randomly selected from among those incident to the collection of vertices at hand at the $(m-1)$ -st stage, excluding those edges already encountered. When a total of k vertices have been sampled in this manner, we obtain the final sampled subgraph H by joining to these vertices and the edges encountered any remaining unencountered edges among these k vertices. That is, H is defined to be the induced subgraph on the set of discovered vertices.

Note that, using this sampling strategy, in order to sample a given k -vertex subgraph H , an ordered set of $k-1$ appropriate edges must be sampled. The probability of sampling H is therefore the sum of the probabilities of all such possible ordered edge sets. This probability is given by

$$p_H = \sum_{\sigma \in S} \prod_{e_j \in \sigma} \mathbb{P}(e_j | (e_1, \dots, e_{j-1})) , \quad (6.12)$$

where S is the set of all possible sets of $k-1$ ordered edges, σ is an element of that set, and e_j is the j -th edge in a specific element σ . The calculation of such probabilities is illustrated in Kashtan, Itzkovitz, Milo, and Alon [218, Fig. 2].

The probability that H is included in a collection of n subgraphs sampled in this fashion is just $\pi_H = 1 - (1 - p_H)^n$, following reasoning similar to that in Example 5.3. Kashtan et al. use the approximation $\pi_H \approx np_H$, which holds when a subgraph H is unlikely to be observed in the sample more than once at most. They also present an algorithm for the overall procedure, with an $O(k^2)$ complexity for sampling each subgraph H and an $O(C^{k-1}k^{k+1})$ complexity for calculating p_H , where C is a small constant correlated with the average degree \bar{d} of the graph G being sampled. Numerical results show that this approach can produce substantial savings in computation time, while maintaining good levels of accuracy, in graphs G with broad degree distributions. However, with concentrated degree distributions, it appears the method does not provide similar savings.

6.3 Small-World Models

Arguably one of the more important innovations in modern network graph modeling is a movement from traditional random graph models, like those in Section 6.2, to models explicitly designed to mimic certain observed ‘real-world’ properties, often through the incorporation of a simple mechanism(s). Work of this type received a good deal of its impetus from the seminal paper of Watts and Strogatz [396] and the ‘small-world’ network model introduced therein.¹²

From an applications perspective, small-world network graph models are of interest across a wide range of disciplines. The interest in these models is largely due to their particular relevance to the topic of communication – where ‘communication’ should be interpreted in the broadest sense. The fact that a network is a small-world network essentially means that its structure is such that a system of neighbor-to-neighbor exchanges is sufficient to ‘transmit’ information quickly across the network, on average. Hearkening back to the context of the original experiment of Milgram [277], it is evident that small worlds are relevant to the spread of news, gossip, rumors, and the like. But they also have been found to be of importance in the spread of diseases, both natural – as in the case of human infectious diseases – and virtual – as in the Internet. See the book by Watts [395], for example, for a broad-ranging non-technical overview and commentary on small-world networks and their relevance. We will consider such topics in slightly more detail in Chapter 8, when we look at models for processes on network graph. Here, in this section, we concentrate simply on presenting definitions and basic properties of small-world network models.

6.3.1 The Watts-Strogatz Model

Watts and Strogatz [396] were intrigued by the fact that many networks in the real world display high levels of clustering, but small distances between most nodes. To better appreciate that this should seem a surprising combination, recall the case of the classical random graph model, for which it was observed in Section 6.2.1 that while the diameter varied as $O(\log N_v)$, indicating ‘small-world’ behavior, the clustering coefficient cl_T varied as N_v^{-1} , which suggests very little clustering.

In order to create a network graph with both of these properties, Watts and Strogatz suggested instead beginning with a graph with lattice structure, and then randomly ‘rewiring’ a small percentage of the edges. More specifically, in this model we begin with a set of N_v vertices, arranged in a periodic fashion, and join each vertex to r of its neighbors to each side. Then, for each edge, independently and with probability p , one end of that edge will be moved to be incident to another vertex,

¹² Bollobás and Riordan [45, Sec. 1.4] point out that mathematical treatments of analogous models of small-world phenomena can be found somewhat earlier in the literature on classical random graphs.

where that new vertex is chosen uniformly, but with attention to avoid the construction of loops and multi-edges. An example of a small-world network graph of this sort is shown in Figure 6.4.

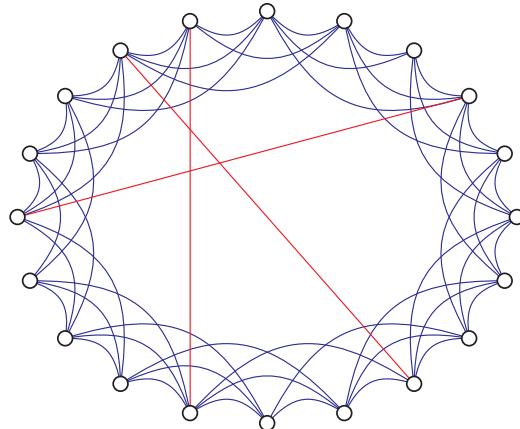


Fig. 6.4 Example of a Watts-Strogatz ‘small-world’ network graph. Blue edges pertain to the original underlying lattice; red edges are rewired.

For the lattice alone, it can be shown that the clustering coefficient $\text{cl}_T(G) = (3r - 3)/(4r - 2)$, or roughly $3/4$ for r large, which indicates a high level of clustering. On the other hand, the diameter clearly varies like $N_v/(2r)$, and the average distance \bar{l} behaves similarly as $N_v/(4r)$, for fixed r as N_v grows. So the distance between vertices on the lattice can be made to stay arbitrarily large, in defiance of the small-world property. But the addition of a few randomly rewired edges has the effect of producing ‘short-cuts’ in the graph. In numerical simulations, due to these short-cuts, Watts and Strogatz observed \bar{l} to drop down to a magnitude of $O(\log N_v)$, even while the graph maintained a clustering coefficient close to $3/4$. This effect may be achieved even with very small p .

To illustrate, consider Figure 6.5, in which we show results from simulating a particular Watts-Strogatz small-world network model, with $N_v = 1,000$ and constant lattice degree of $d = 12$. Re-wiring is then done with probability p , as p varies from 0 to 1. Both the clustering coefficient¹³ and the average distance \bar{l} have been normalized by their largest values, which occur at a re-wiring probability of $p = 0$. It is quite evident from the figure that, over a non-trivial range of p – from roughly 10^{-3} to 10^{-1} in this particular setting – the network exhibits small average distance while maintaining a high level of clustering.

¹³ Here we use cl , as in the original Watts-Strogatz analysis, rather than cl_T .

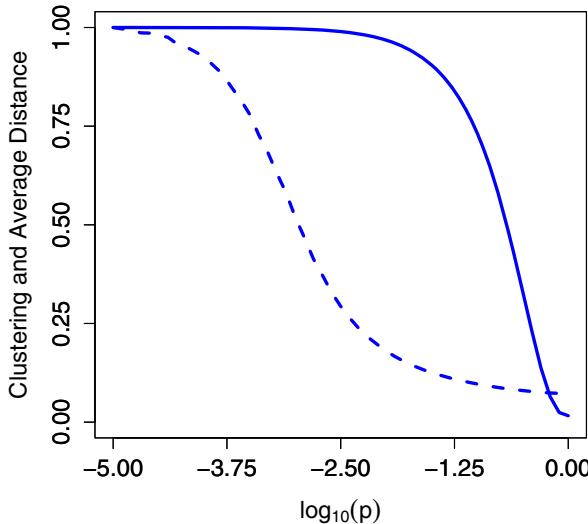


Fig. 6.5 Plot of the clustering coefficient $cl(G)$ (solid) and average geodesic distance \bar{l} (dashed), as a function of the rewiring probability p for a Watts-Strogatz small-world model. Results are averages based on 1,000 simulation trials.

Since the introduction of the model of Watts and Strogatz, a number of authors have focused upon better understanding its properties. However, the precise, formal characterization of these properties appears to still be an open problem. Barrat and Weigt [19] have argued that, for large N_v , the clustering coefficient cl_T should behave like

$$\frac{3r-3}{4r-2}(1-p)^3 \quad . \quad (6.13)$$

This expression differs from the value for the classical random graph only by the factor $(1-p)^3$, which will be relatively close to 1 for small p . Similarly, there has been progress on providing partial analytical characterizations of \bar{l} , in the form of partial series expansions and scaling properties. See Newman [296, Sec. VI] and Boccaletti et al. [41] for details and further references.

6.3.2 Other Small-World Network Models

Often it is slight variations of the original Watts-Strogatz model that are actually studied in the literature, being essentially equivalent and more amenable to analytical calculations. Perhaps most popular is the model where no edges are rewired, but

rather a small number of new edges are added to randomly selected pairs of vertices, as proposed by Monasson [284] and by Newman and Watts [298]. Note that in terms of simulation, any of these models are straightforward to generate. For example, the model in Newman and Watts [298] simply requires the generation of (i) a lattice, which may be done easily in linear time, and (ii) a classical random graph on top of that. Using the ‘skipping’ algorithm described in Section 6.2.3 for the second step, we obtain an overall $O(N_v + N_e)$ algorithm. Similarly, Batagelj and Brandes [23] provide an $O(N_v + N_e)$ algorithm for a version of the original small-world model in which edges are fully rewired (i.e., both vertices change).

Other mechanisms have also been proposed that successfully generate network graphs with the small-world property. For example, Kasturirangan [219] begins with a one-dimensional periodic lattice, but then, instead of rewiring or adding edges among vertices in that lattice, a small number of vertices are added to the center and each is connected randomly to a large number of vertices throughout the original lattice. Alternatively, Kleinberg [229, 231], studying small-world network graphs from an algorithmic perspective, proposes a model that begins with a two-dimensional lattice (i.e., a standard grid) and then adds short-cuts between vertices with a probability varying inversely proportional to the distance between them. That is, with $p \propto (\text{dist})^{-r}$, for some $r > 0$. Kleinberg shows that, whereas for the original Watts-Strogatz model no algorithm exists which can find shortest paths between vertices using only local information, for his model, when $r = 2$ (and, interestingly, only when $r = 2$), there exists a simple algorithm for doing so. The implication here is that, in order to mimic the types of small worlds that we navigate successfully on a daily basis, it is not sufficient simply to add random short-cuts. Rather, there must be some bias in the way they are laid, encouraging them to be – at an appropriate level – more local than not.

6.4 Network Growth Models

Many networks grow or otherwise evolve in time. The World Wide Web and scientific citation networks are two obvious examples. Similarly, many biological networks may be viewed as evolving as well, over appropriately defined time scales. Much energy has been invested in the development of models that mimic network growth.¹⁴ In this arena, typically a simple mechanism(s) is specified for how the network changes at any given point in time, based on concepts like vertex preference, fitness, copying, age, and the like. Usually interest then centers on what properties emerge for the network in the limit of a large number of consecutive time periods. If certain properties are found to match those observed in real-world networks, this is often taken as suggestive that the specified mechanism is perhaps a reasonable approximation to a similar, real-world mechanism. Although generally the reality of the forces behind the emergence of real-world networks is decidedly more subtle

¹⁴ What we call ‘network growth’ models here are commonly referred to as ‘evolutionary network’ models in the physics literature.

and involved than the comparatively simple mechanisms proposed in these models, work on mechanisms like these nevertheless has proven useful in thinking more deeply about real-world network growth and design.

In this section, we discuss models based on two fundamental and popular classes of growth mechanisms – preferential attachment and copying. We also briefly discuss the problem of fitting such models to network data.

6.4.1 Preferential Attachment Models

Preferential attachment mechanisms are designed to embody the principle that ‘the rich get richer.’ A driving motivation behind the introduction of these mechanisms was a desire to reproduce the types of broad degree distributions observed in many large, real-world networks. Simon [353], intrigued by the appearance of power-law distributions in various settings (e.g., the frequency of occurrence of words, or the relative size of city populations), proposed a class of models in 1955 that produced such broad, skewed distributions. Price [112] then took this idea a decade later and applied it in creating a model for the manner in which networks of citations for documents in the literature grow. But it is the work of Barabási and Albert [18], nearly 35 years after Price, that launched the present-day fascination with models of this type, and so it is with their model that we begin our exposition here.¹⁵

Barabási and Albert were motivated by the growth of the World Wide Web, noting that often web pages to which many other pages point will tend to accumulate increasingly greater numbers of links as time goes on. For example, think of how people increasingly added websites like Google to their list of ‘Favorites’ in their web browsers, as those sites grew in popularity. The Barabási-Albert (BA) model is a network growth model for undirected graphs, formulated originally as follows. Start with an initial graph $G^{(0)}$ of $N_v^{(0)}$ vertices and $N_e^{(0)}$ edges. Then, at stage $t = 1, 2, \dots$, the current graph $G^{(t-1)}$ is modified to create a new graph $G^{(t)}$ by adding a new vertex of degree $m \geq 1$, where the m new edges are attached to m different vertices in $G^{(t-1)}$, and the probability that the new vertex will be connected to a given vertex v is given by

$$\frac{d_v}{\sum_{v' \in V} d_{v'}} . \quad (6.14)$$

That is, at each stage, m existing vertices are connected to a new vertex in a manner preferential to those with higher degrees. After t iterations, the resulting graph $G^{(t)}$ will have $N_v^{(t)} = N_v^{(0)} + t$ vertices and $N_e^{(t)} = N_e^{(0)} + tm$ edges. And, because of the tendency towards preferential attachment, intuitively we would expect that a number of vertices of comparatively high degree should gradually emerge as t increases.

Note that in this formulation there is some latitude left for specifying precisely how to select a set of m vertices in proportion to their degree, since there is in

¹⁵ For an extensive history of power-law models, including their use in network modeling, see Mitzenmacher [280].

fact more than one way to do so.¹⁶ Bollobás and colleagues [44, 46], citing this ambiguity, have proposed a particular version of the BA preferential attachment model in which the nature of the sampling is made precise, called the *linearized-chord diagram* (LCD) model, which is defined as follows. For the case $m = 1$, we begin with a graph $G^{(1)}$ consisting of a single vertex with a loop. Then, for each $t = 2, 3, \dots$, the graph $G^{(t)}$ is created from $G^{(t-1)}$ by adding the vertex v_t to $G^{(t-1)}$ with an edge to a vertex v_s , for $1 \leq s \leq t$ chosen randomly with respect to the distribution

$$\mathbb{P}(s = j) = \begin{cases} d_{G^{(t-1)}}(v_j)/(2t-1), & \text{if } 1 \leq j \leq t-1, \\ 1/(2t-1), & \text{if } j = t, \end{cases} \quad (6.15)$$

where $d_{G^{(t-1)}}(v_j)$ denotes the degree of the vertex v_j at time $t-1$. For the case $m > 1$, the above process is simply run for m steps at a time, after which the m vertices just created are contracted into one vertex, with the m edges retained. Note that this formulation of the preferential attachment model allows for loops and multi-edges. However, these should occur relatively rarely, and the precision gained by this formulation is necessary for rigorously deriving mathematical results regarding model properties.

The most celebrated property of such preferential attachment models is that, in the limit as t tends to infinity, the graphs $G^{(t)}$ have degree distributions that tend to a power-law form $d^{-\alpha}$, with $\alpha = 3$. More precisely, formalizing a body of earlier results, such as Barabási and Albert [18], Dorogovtsev, Mendes, and Samukhin [124], and Krapivsky, Redner, and Leyvraz [240], it was shown in Bollobás, Riordan, Spencer, and Tusnady [46] that with high probability

$$(1 - \varepsilon) f_{d,m} \leq f_d(G^{(t)}) \leq (1 + \varepsilon) f_{d,m}, \quad (6.16)$$

for any $\varepsilon > 0$, and every d in the range $0 \leq d \leq \left(N_v^{(t)}\right)^{1/5}$, where

$$f_{d,m} = \frac{2m(m+1)}{(d+2)(d+1)d}, \quad (6.17)$$

which behaves like d^{-3} for d large relative to m . The expression in (6.16) may be compared to that in (6.3), describing the Poisson limiting behavior for classical random graphs.

A number of other properties of interest have been established for these preferential attachment models as well. Note that in the original formulation of the BA model, the graphs $G^{(t)}$ are connected, by definition, assuming that the initial graph $G^{(0)}$ was connected. But in the LCD model, these graphs are no longer necessarily connected. However, Bollobás and Riordan [44] show that they are connected with high probability. They also show that these random graphs will exhibit small-world behavior. In particular, with high probability, in the case $m = 1$, the diam-

¹⁶ See Bollobás and Riordan [45, Sec. 1.6] for a discussion of some of the issues that can arise due to this latitude.

eter of $G^{(t)}$ behaves like $O(\log N_v^{(t)})$; in the case $m > 1$, it actually behaves like $O(\log N_v^{(t)} / \log \log N_v^{(t)})$, which is a bit smaller still. On the other hand, with respect to clustering, these models are known to be somewhat unsatisfactory, in that the resulting graphs display decidedly less clustering than often observed in real-world networks of similar magnitude. For instance, Bollobás and Riordan [45] show that the expected value $\mathbb{E}[\text{cl}_T(G^{(t)})]$ behaves like

$$\frac{m-1}{8} \frac{(\log N_v^{(t)})^2}{N_v^{(t)}} , \quad (6.18)$$

which is only a little better than the N_v^{-1} behavior in the case of classical random graph models.

These preferential attachment models may be simulated in linear time. Batagelj and Brandes [23] describe an efficient algorithm for simulating the LCD model. Also see Newman [296, Sec. VII.B].

There have been many extensions and variations on the basic preferential attachment model that have been proposed in the literature – far too many to even summarize adequately here – for modeling both undirected graphs (as in the original BA model) and directed graphs. However, overall the common concern in offering these extensions has primarily been (i) whether or not a power-law limiting distribution is achieved, and (ii) if so, the manner in which the model mechanism(s) and parameters affect the power-law exponent. Some models modify the functional form of the preferential attachment probability in (6.14), and by doing so produce a richer collection of possible limiting behaviors for the degree distribution of the resulting random graphs than the original BA model. For example, even simply including an offset parameter $-m < d^* < \infty$ in the numerator and denominator, by changing d_v to $d_v + d^*$ throughout, yields models with limiting degree distributions that behave like power laws with exponent $\alpha = 3 + d^*/m$. That is, we can produce exponents anywhere in the range $(2, \infty)$. Or, replacing the linear role of the degrees d_v in (6.14) by powers of the form d_v^γ , it is possible to produce limiting degree distributions with a number of different functional forms, as γ varies, deviating to various extents from the strict power-law behavior arising in the case $\gamma = 1$. Models including some measure of the ‘fitness’ or inherent quality of vertices have also been proposed by many authors, as have models that supplement preferential attachment with some additional mechanism(s), often with an eye towards inducing a greater amount of clustering in the network. Finally, models have also been proposed that allow the number of new edges m to vary randomly, as well as models that allow for both addition and removal of edges.

See Albert and Barabási [6] for an extensive review of much of this literature, and their Table III for a concise summary. Also see Newman [296, Sec. VII.C] and Boccaletti et al. [41, Sec. 2.3.5], which offer summaries a bit more recent, albeit less extensive. Additional resources are also described in Section 6.7.

6.4.2 Copying Models

Another mechanism of fundamental interest, which is distinct from preferential attachment but can nevertheless also produce power-law degree distributions, is that of copying. Like preferential attachment, *copying mechanisms* were also originally proposed in the context of the World Wide Web – in this case by Kleinberg et al. [232]. However, they arguably have generated the most interest in the context of biochemical networks where, unlike preferential attachment, copying is felt to be a tenable mechanism from the perspective of basic biology. For example, it is commonly held that gene duplication is at the heart of nature’s observed tendency to ‘re-use’ biological information in evolving the genomes of living organisms (e.g., Ohno [305]).

Chung, Lu, Dewey, and Galas [89] offer a network growth model based on the following simple, straightforward notion of copying (or ‘duplication,’ in their terminology). Beginning with an initial graph $G^{(0)}$, graphs $G^{(t)}$ are constructed from their immediate predecessors, $G^{(t-1)}$, by the addition of a new vertex, say v , that is connected to some randomly chosen subset of neighbors of a randomly chosen existing vertex, say u . More precisely, a vertex u is chosen from $G^{(t-1)}$ uniformly at random, and then the new vertex v is joined with each of the neighbors of u in $G^{(t-1)}$ independently with probability p .

Chung et al. show that, with high probability, the degree distribution $\{f_d(G^{(t)})\}$ will tend to a power-law form $d^{-\alpha}$, with exponent α satisfying the equation

$$p(\alpha - 1) = 1 - p^{\alpha-1} . \quad (6.19)$$

This equation will have two solutions α for any given p , but only one will be stable. For $p > 0.5671\dots$, the stable solution is simply $\alpha = 1$; for $p = 1/2$, it is $\alpha = 2$. Figure 6.6 shows a plot of the solutions, as a function of p .

Note that when $p = 1$, each new vertex v is connected to $G^{(t-1)}$ by fully duplicating the edges of the randomly selected vertex u . Citing previous numerical work of theirs showing that power-law behavior will not result when $p = 1$, Chung et al. prove that to achieve power-law behavior with full duplication it is sufficient to allow partial duplication to occur some fraction $q \in (0, 1)$ of the times that a new vertex is added.

There have been other copying models suggested as well. For example, prior to Chung, Lu, Dewey, and Galas [89], growth models were proposed by Kleinberg et al. [232] and by Kumar et al. [241] in which, when creating $G^{(t)}$ from $G^{(t-1)}$, a new vertex v receives $m \geq 1$ new edges to vertices in $G^{(t-1)}$ by a combination of either (i) copying edges from a randomly chosen vertex u or (ii) simply establishing new edges. More precisely, for each new vertex, a ‘prototype’ vertex u is chosen from $G^{(t-1)}$, as in the approach of Chung, Lu, Dewey, and Galas [89]. Then, for $i = 1, \dots, m$, the i -th edge is chosen with probability β to be from v to some vertex selected uniformly at random from $G^{(t-1)}$, while with probability $1 - \beta$, it is chosen instead to be to the i -th neighbor of u .

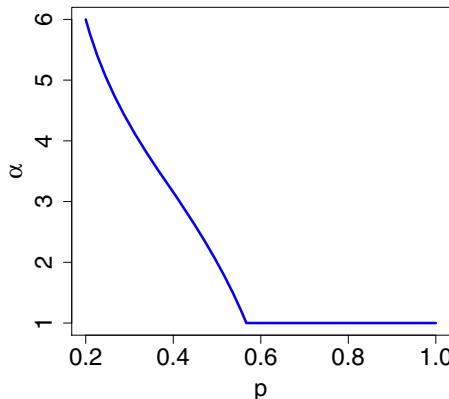


Fig. 6.6 Power-law exponent α , as a function of p , for the copying model of Chung, Lu, Dewey, and Galas [89], as given by the expression in equation (6.19).

The original conceptual image behind this model is that of an individual setting up a new web page. On the one hand, the web page is likely being set up in the context of some established topic(s), and thus will probably include a number of established links in that topic. On the other hand, an individual also brings their own perspective to the topic, and will be expected to generate some previously unseen links as well. Alternatively, as described in Kim, Krapivsky, Kahng, and Redner [226], where a very similar model is proposed for protein interaction networks, we can envision the structure of genes associated with the proteins as typically evolving through duplication, as above, but periodically arising instead through mutation.

For this model, the degree distribution $\{f_d(G^{(t)})\}$ also tends to a power law. Specifically, Kumar et al. [241] show that, for each $d > 0$, the sequence of values $f_d(G^{(t)})$ tends – as t tends to infinity – to

$$f_d = f_0 \prod_{j=1}^d \frac{1 + \beta / (j(1 - \beta))}{1 + 2 / (j(1 - \beta))} . \quad (6.20)$$

The value f_d in (6.20) in turn can be shown to behave like $d^{-\alpha}$, for $\alpha = (2 - \beta) / (1 - \beta)$. Kumar et al. [241] also show that this model generates many more dense bipartite subgraphs than are found in comparable classical random graphs. This characteristic is encountered in real web graphs, and was one of the primary motivations behind their work.

On a final note, we mention that models have also been proposed that combine both copying and preferential attachment mechanisms. Cooper and Frieze [99], for example, introduce a particularly general network growth model of this sort and

establish a set of precise conditions for power-law behavior to occur. Their results, however, are somewhat complicated to state, and are omitted here.

6.4.3 Fitting Network Growth Models

By far the most common practical usage of network growth models (i.e., in conjunction with analyzing observed network graphs) has been predictive, in the sense of making informal comparisons between certain characteristics of an observed network and the graphs $G^{(t)}$ resulting from such models. Both theoretical calculations, like those described above, and numerical simulations are generally used in making comparisons of this sort. We note, however, that there has been little attempt to date to actually fit standard network growth models to network data. This is to be expected, on the one hand, given the comparative (and, generally, intentional) simplicity of such models. On the other hand, it can be argued that fitting even these simple models may be useful in some cases, in that estimates of parameters, like the duplication probability p in the copying model of Chung, Lu, Dewey, and Galas [89], can provide at least a rough characterization of basic rates and proportions of some fundamental interest.

Ideally, in order to fit a network growth model we would like to have a dynamic sequence of ‘snap-shots’ of the graph as it has changed over consecutive periods of time. In practice, however, as mentioned earlier in this book, such data still remain fairly elusive in most settings. Instead, we effectively find ourselves faced in most situations with a single snap-shot, which we may conveniently picture as the end-product of some sort of process of evolution. For this situation, Wiuf, Brameier, Hagberg, and Stumpf [407] have proposed a methodology for fitting network growth models that makes clever usage of their recursive nature to get around this lack of multiple snap-shots.

Suppose that the class of network growth models to be fit is defined so that a single vertex is added to $G^{(t-1)}$ in creating $G^{(t)}$ and, importantly, that the manner in which this vertex is added depends only on the vertices and edges present in $G^{(t-1)}$ and no earlier.¹⁷ Such is the case, for example, in all of the models described above. Next, define $\delta(G^{(t)}, v)$ to be the graph obtained by deleting the vertex v and all edges incident to it from $G^{(t)}$. We will call v *removable* if $G^{(t)}$ can be created from $\delta(G^{(t)}, v)$ through copying. If $G^{(t)}$ contains no removable vertices, we will call it *irreducible*; otherwise, $G^{(t)}$ will be said to be *reducible*.

Wiuf et al. consider a class of models they call *duplication-attachment* (DA) models, parameterized by a vector θ , that includes a number of the models already described above as special cases. The key properties satisfied by these models are

- (i) the initial graph $G^{(0)}$ is irreducible, and
- (ii) $\mathbb{P}_\theta(G^{(t)}|G^{(t-1)}) > 0$ if and only if $G^{(t)}$ can be obtained by copying one vertex in $G^{(t-1)}$.

¹⁷ That is, it is assumed that $\{G^{(t)}\}_{t=0}^\infty$ forms a Markov chain.

Assuming that $G^{(t)}$ represents the observed network graph, they define a likelihood function¹⁸ for θ recursively as

$$\mathcal{L}(\theta; G^{(t)}) = \frac{1}{t} \sum_{v \in \mathcal{R}(G^{(t)})} \mathbb{P}_\theta(G^{(t)} | \delta(G^{(t)}, v)) \mathcal{L}(\theta; \delta(G^{(t)}, v)) , \quad (6.21)$$

where $\mathcal{R}(G^{(t)})$ is the set of all removable nodes in $G^{(t)}$. A maximum likelihood estimate $\hat{\theta}$ is then defined as the maximizer of (6.21) in θ .

Unfortunately, explicit calculation of the likelihood function is non-trivial, even for network graphs of modest size, due to the recursion involved. For the extreme case of a complete graph, the complexity of this recursion varies like $O(t2^t)$; but in practice, even with sparser network graphs, Wiuf et al. observed the complexity to still scale quite poorly.

But Monte Carlo methods (i.e., so-called sequential importance sampling) may be used profitably to produce stochastic approximations to the likelihood. Wiuf et al. show how (6.21) may be re-expressed as an expectation in the form

$$\mathcal{L}(\theta; G^{(t)}) = \mathbb{E}_{\theta_0} \left[\prod_{s=0}^t S(\theta_0, \theta, G^{(s)}, v) \right] , \quad (6.22)$$

for arbitrary θ_0 , where $S(\cdot)$ is defined to be 1 at $s = 0$ and is a function of the probabilities $\mathbb{P}_{\theta'}(G^{(s)} | \delta(G^{(s)}, v))$ for $1 \leq s \leq t$ and $\theta' = \theta_0$ or θ . Sequences $G^{(1)}, \dots, G^{(t)}$ are simulated according to the specified DA model, under θ_0 , and $G^{(0)}$ is taken to be the irreducible graph underlying the observed graph. The product in (6.22) is evaluated for each simulated sequence. Averaging a sufficient number of these products yields an approximation to the expectation in (6.22), and hence to the likelihood function evaluated at θ .

Wiuf et al. illustrate the potential of this approach by calculating a univariate likelihood function for the probability p in an extended version of the copying model of Chung, Lu, Dewey, and Galas [89], for a network of interactions among 2,368 proteins in the worm organism *Caenorhabditis elegans*. However, as pointed out by these authors themselves, there are a number of issues that remain to be solved for the methodology to be scaled up effectively to more complicated contexts, such as those involving multivariate parameters, larger networks, and more realistic network growth models. It is also an open problem to establish an appropriate theory for confidence intervals and testing in this context.

¹⁸ In principle, to specify a full likelihood function we should model the initial graph $G^{(0)}$. Wiuf et al. show that the set of irreducible graphs obtainable from repeatedly removing removable vertices forms an equivalence class, in the sense that all such graphs are isomorphic to each other, and so argue that the additional model structure is unnecessary. Alternatively, (6.21) may be viewed as defining a conditional likelihood.

6.5 Exponential Random Graph Models

The models discussed so far in this chapter serve a variety of useful purposes. Yet for the purpose of statistical model building, in the sense of proposing, fitting, and evaluating a model(s) in a manner informed intimately by observed data, they come up short. Indeed, as Robins and Morris [328] write, “A good [statistical network graph] model needs to be both estimable from data and a reasonable representation of that data, to be theoretically plausible about the type of effects that might have produced the network, and to be amenable to examining which competing effects might be the best explanation of the data.” None of the models we have seen up until this point are really intended to meet such criteria.

In contrast, *exponential random graph models* (ERGMs)¹⁹ are designed precisely with these criteria in mind and are formulated in a manner that facilitates the adaptation and extension of well-established statistical principles and methods for the construction, fitting, and comparison of models. As a result, they have the potential to come substantially closer to satisfying the criteria stated above. On the other hand, the highly complex patterns of dependency underlying the relational ties in typical observed network graphs mean that the necessary adaptations and extensions are far from trivial.

We discuss here the key aspects of this approach to network graph modeling and point out some of its current limitations as well. An illustration of the methodology will be presented in the case study that follows this discussion, in Section 6.5.4.

6.5.1 Model Specification

An arbitrary (discrete) random vector \mathbf{Z} is said to belong to an *exponential family* if its probability mass function may be expressed in the form

$$\mathbb{P}_\theta(\mathbf{Z} = \mathbf{z}) = \exp\{\theta^T \mathbf{g}(\mathbf{z}) - \psi(\theta)\}, \quad (6.23)$$

where $\theta \in \mathbb{R}^p$ is a $p \times 1$ vector of parameters, $\mathbf{g}(\cdot)$ is a p -dimensional function of \mathbf{z} , and $\psi(\theta)$ is a normalization term, ensuring that $\mathbb{P}_\theta(\cdot)$ sums to one over its range. The class of discrete exponential families includes many familiar distributions, such as the binomial, geometric, and Poisson. In the case of continuous exponential families, where an analogous form of (6.23) holds for probability density functions, examples include the Gaussian and chi-square distributions. Exponential families all have a variety of common useful algebraic and geometric properties, mak-

¹⁹ These models are also referred to as p^* models, particularly in the social network literature, where they are seen as one of the later examples of a series of model classes introduced in succession over a roughly 20-year period covering the late 1970’s, 1980’s, and early 1990’s. See the review of Wasserman and Pattison [394], for example. Our use of the term ‘exponential random graph models’ reflects current practice, which emphasizes the connection of these models with traditional exponential family models in classical statistics, as we shall see below.

ing this class of distributions mathematically convenient for purposes of inference and simulation.

Consider $G = (V, E)$ as a random graph. Let $Y_{ij} = Y_{ji}$ be a binary random variable indicating the presence or absence of an edge $e \in E$ between the two vertices i and j in V . The matrix $\mathbf{Y} = [Y_{ij}]$ is thus the (random) adjacency matrix for G . Denote by $\mathbf{y} = [y_{ij}]$ a particular realization of \mathbf{Y} . An exponential random graph model is a model specified in exponential family form for the joint distribution of the elements in \mathbf{Y} . More precisely, an ERGM takes the form

$$\mathbb{P}_\theta(\mathbf{Y} = \mathbf{y}) = \left(\frac{1}{\kappa} \right) \exp \left\{ \sum_H \theta_H g_H(\mathbf{y}) \right\}, \quad (6.24)$$

where

- (i) each H is a *configuration*, which is defined to be a set of possible edges among a subset of the vertices in G ;
- (ii) $g_H(\mathbf{y}) = \prod_{y_{ij} \in H} y_{ij}$, and is therefore either one if the configuration H occurs in \mathbf{y} , or zero, otherwise;
- (iii) a non-zero value for θ_H means that the Y_{ij} are dependent for all pairs of vertices $\{i, j\}$ in H , conditional upon the rest of the graph; and
- (iv) $\kappa = \kappa(\theta)$ is a normalization constant,

$$\kappa(\theta) = \sum_{\mathbf{y}} \exp \left\{ \sum_H \theta_H g_H(\mathbf{y}) \right\}. \quad (6.25)$$

The summation in (6.24) is over all possible configurations H . Note that this model implies a certain (in)dependency structure among the elements in \mathbf{Y} . Generally speaking, such assumptions specify that the random variables $\{Y_{ij}\}_{(i,j) \in \mathcal{A}}$ are independent of $\{Y_{i'j'}\}_{(i',j') \in \mathcal{B}}$, conditional on the values of $\{Y_{i''j''}\}_{(i'',j'') \in \mathcal{C}}$, for given index sets \mathcal{A} , \mathcal{B} , and \mathcal{C} . Conversely, we can begin with a collection of (in)dependence relations among subsets of elements in \mathbf{Y} and try to develop a model. However, it is important to realize that it is *not* the case that simply any collection of (in)dependence relations among the elements of \mathbf{Y} yields a proper joint distribution on \mathbf{Y} . Rather, certain conditions must be satisfied, as formalized in the celebrated Hammersley-Clifford theorem (e.g., Besag [36]).

We postpone a somewhat more detailed discussion of these issues until Chapter 10, and instead content ourselves here with illustrating them through examples.

Example 6.6 (Bernoulli Random Graphs). Suppose we specify that, for any given pair of vertices, the presence or absence of an edge between that pair is independent of the status of possible edges between any other pairs of vertices. That is, for each pair $\{i, j\}$, we assume that Y_{ij} is independent of $Y_{i'j'}$, for any $\{i', j'\} \neq \{i, j\}$. This assumption implies that $\theta_H = 0$ for all configurations H involving three or more vertices. In this case, therefore, the only relevant functions g_H are those of the form $g_H(\mathbf{y}) = g_{ij}(\mathbf{y}) = y_{ij}$, and the ERGM in (6.24) thus reduces to

$$\mathbb{P}_\theta(\mathbf{Y} = \mathbf{y}) = \left(\frac{1}{\kappa} \right) \exp \left\{ \sum_{i,j} \theta_{ij} y_{ij} \right\} . \quad (6.26)$$

Note that (6.26) is just another way of writing that each edge $\{i, j\}$ is present in the graph independently with probability

$$p_{ij} = \exp(\theta_{ij}) / [1 + \exp(\theta_{ij})] . \quad (6.27)$$

However, this entails a model with N_v^2 parameters, which is likely far too over-parameterized. In order to reduce the total number of parameters, it is common to impose an assumption of homogeneity across certain vertex pairs. For example, assuming homogeneity across all of G (i.e., $\theta_{ij} \equiv \theta$, for all $\{i, j\}$) yields

$$\mathbb{P}_\theta(\mathbf{Y} = \mathbf{y}) = \left(\frac{1}{\kappa} \right) \exp \{ \theta L(\mathbf{y}) \} , \quad (6.28)$$

where $L(\mathbf{y}) = \sum_{i,j} y_{ij} = N_e$ is the number of edges in the graph. In this case, the Bernoulli random graph model of Section 6.2.1 is recovered, with $p = \exp(\theta) / [1 + \exp(\theta)]$.

Alternatively, suppose that vertices are known *a priori* to fall within either of two sets, say S_1 and S_2 . If we impose homogeneity within and between sets, we arrive at a model of the form

$$\mathbb{P}_\theta(\mathbf{Y} = \mathbf{y}) = \left(\frac{1}{\kappa} \right) \exp \{ \theta_{11} L_{11}(\mathbf{y}) + \theta_{12} L_{12}(\mathbf{y}) + \theta_{22} L_{22}(\mathbf{y}) \} , \quad (6.29)$$

where $L_{11}(\mathbf{y})$ and $L_{22}(\mathbf{y})$ are the number of edges within sets S_1 and S_2 , respectively, and $L_{12}(\mathbf{y})$ is the number of edges between S_1 and S_2 . \square

Unfortunately, assumptions of complete independence among possible edges are largely untenable in practice. In addition, we saw ample evidence earlier, in Section 6.2.1, to indicate that Bernoulli-like random graphs lack the ability to reproduce many of the most basic structural characteristics observed in most real-world networks. But certain simple conditional independence assumptions may be used profitably to create a much richer class of models.

Example 6.7 (Markov Random Graphs). Frank and Strauss [155] introduced the notion of *Markov dependence* for network graph models, which specifies that two possible edges are dependent whenever they share a vertex, conditional on all other possible edges. That is, the presence or absence of $\{i, j\}$ in the graph will depend upon that of $\{i, k\}$, for a given $k \neq j$, even given information on the status of all other possible edges in the network. A random graph G arising under Markov dependence conditions is called a *Markov graph*.

Under an assumption of homogeneity, using the Hammersley-Clifford theorem, Frank and Strauss show that G is a Markov graph if and only if $\mathbb{P}_\theta(\cdot)$ may be expressed as

$$\mathbb{P}_\theta(\mathbf{Y} = \mathbf{y}) = \left(\frac{1}{\kappa} \right) \exp \left\{ \sum_{k=1}^{N_v-1} \theta_k S_k(\mathbf{y}) + \theta_\tau T(\mathbf{y}) \right\} , \quad (6.30)$$

where $S_1(\mathbf{y}) = N_e$ is the number of edges, $S_k(\mathbf{y})$ is the number of k -stars,²⁰ for $2 \leq k \leq N_v - 1$, and $T(\mathbf{y})$ is the number of triangles.

Note that the statistics S_k in (6.30), and also T , can be expected to often be correlated. For example, more edges in G clearly allows for the possibility of more k -stars of certain orders. Similarly, more k -stars, for a given k , means more k' -stars, for $k' < k$. In this sense, we may view the θ_k , for increasingly larger k , as successively higher-order effects in the model. And the inclusion of lower-order effects $\theta_{k'}$ means that θ_k is the effect due to k -stars, adjusted for the quantity of lower-order stars. Additionally, more k -stars means the potential for more triangles, and the value θ_τ is therefore the effect of triangles, adjusted for the levels of all k -stars.

Interestingly, we see from the nature of the statistics S_k and T that the transition from the independence assumptions underlying Bernoulli models to Markov dependence results in a model explicitly parameterized to account for some effects of transitivity, something lacking in the models introduced previously in this chapter.

□

In using Markov graph models, it has traditionally been common practice to include star counts S_k no higher than $k = 2$, or at most $k = 3$, by setting $\theta_4 = \dots = \theta_{N_v-1} = 0$. But experience has shown this practice to frequently produce models that fit quite poorly to real data. Investigation of this phenomena has found it to be intimately related to the issue of model degeneracy, which we define and discuss below in Section 6.5.3. Unfortunately, the alternative – including a sufficiently large number of higher order terms – is problematic as well, from the perspective of model fitting.

A solution to this dilemma, proposed by Snijders, Pattison, Robins, and Handcock [360], is to impose a parametric constraint of the form $\theta_k \propto (-1)^k \lambda^{2-k}$ upon the star parameters, for all $k \geq 2$, for some $\lambda \geq 1$. This tactic has the effect of combining all of the k -star statistics $S_k(\mathbf{y})$ in (6.30), for $k \geq 2$, into a single *alternating k-star statistic* of the form

$$AKS_\lambda(\mathbf{y}) = \sum_{k=2}^{N_v-1} (-1)^k \frac{S_k(\mathbf{y})}{\lambda^{k-2}} , \quad (6.31)$$

and weighting that statistic by a single parameter θ_{AKS} that takes into account the star effects of all orders simultaneously.

One may think of the alternating signs in (6.31) as allowing the counts of k -stars of successively greater order to balance each other, rather than simply ballooning. Alternatively, it may be shown that the statistic $AKS_\lambda(\mathbf{y})$ is a linear function of the *geometrically weighted degree count*, which is defined as

²⁰ A k -star is a tree, with one vertex of degree k , and k vertices of degree 1.

$$GWD_\gamma(\mathbf{y}) = \sum_{d=0}^{N_v-1} e^{-\gamma d} N_d(\mathbf{y}) , \quad (6.32)$$

where $N_d(\mathbf{y})$ is the number of vertices of degree d and $\gamma > 0$ is related to λ through the expression $\gamma = \log[\lambda/(\lambda - 1)]$. So this approach in a sense attempts to model the degree distribution, with choice of γ influencing the extent to which higher-degree vertices are likely to occur in the graph G .

Snijders, Pattison, Robins, and Handcock [360] discuss a number of other similar statistics, including a generalization of triadic structures based on alternating sums of k -triangles, which takes the form²¹

$$AKT_\lambda(\mathbf{y}) = 3T_1 + \sum_{k=2}^{N_v-2} (-1)^{k+1} \frac{T_k(\mathbf{y})}{\lambda^{k-1}} . \quad (6.33)$$

Here T_k is the number of k -triangles, where a k -triangle is defined to be a set of k individual triangles sharing a common base. These authors show that the inclusion of k -triangles in a model leads to a distribution $\mathbb{P}_\theta(\cdot)$ in which two potential edges are dependent, conditional upon the rest of the graph, if either (i) they share a common vertex, or (ii) their presence would create a four-cycle. As such, these models satisfy the so-called *partial conditional dependence* condition of Pattison and Robins [311], a non-Markov notion of dependence designed to allow more complex forms of inter-dependence than Markov dependence (e.g., such as through the status of ‘third party’ edges).

In fact, the ERGM framework is generally quite versatile and can easily allow for a number of additional generalizations. For example, directed versions of ERGMs are also available. The paper by Holland and Leinhardt [203] is seminal in this area, in moving from independent edges $Y_{ij} = Y_{ji}$ to independent dyads (Y_{ij}, Y_{ji}) . The Markov models of Frank and Strauss [155] apply to this case as well, and similarly allow for a departure from this independence assumption. ERGMs also extend to bipartite graphs and multivariate networks (i.e., where more than one network is considered simultaneously on the same set of vertices). Finally, in defining ERGMs for either undirected or directed graphs, it is straightforward to include, if desired, additional information on vertices beyond their connectivity, such as actor attributes in a social network or known functionalities of proteins in a network of protein interactions. Given a realization \mathbf{x} of a random vector \mathbf{X} on the vertices in G , we simply specify an exponential form for the conditional distribution $\mathbb{P}_\theta(\mathbf{Y} = \mathbf{y} | \mathbf{X} = \mathbf{x})$ that involves additional statistics $g(\cdot)$ that are functions of both \mathbf{y} and \mathbf{x} . An illustration may be found in Section 6.5.4.

²¹ Hunter [208] offers an equivalent formulation of this definition, in terms of geometrically weighted counts of the neighbors common to adjacent vertices.

6.5.2 Fitting Exponential Random Graph Models

In standard settings, with independent and identically distributed realizations, exponential family models like that in (6.23) are generally fit using the method of maximum likelihood, and the resulting parameter estimates $\hat{\theta}$ are accompanied by asymptotically justified confidence intervals and test statistics. In the context of the ERGMs in (6.24), however, traditionally the corresponding inferential framework has been less well developed. The maximum likelihood estimators (MLEs) $\hat{\theta}_H$ of the parameters θ_H are well defined, but their calculation is non-trivial. Furthermore, an appropriate asymptotic theory for confidence intervals and testing, taking into account the highly dependent nature of observations in a network graph, has yet to be established. Research in this area has concentrated primarily on the problem of fitting MLEs for ERGMs; we review some of the key methods emerging from this work here.

Consider the general definition of an ERGM in (6.24). The MLE for the vector $\theta = (\theta_H)$ is defined as $\hat{\theta} = \arg \max_{\theta} \ell(\theta)$, where $\ell(\theta)$ is the log-likelihood, which has the particularly simple form common to exponential families,

$$\ell(\theta) = \theta^T \mathbf{g}(\mathbf{y}) - \psi(\theta) . \quad (6.34)$$

Here \mathbf{g} denotes the vector of functions g_H and $\psi(\theta) = \log \kappa(\theta)$. Alternatively, taking derivatives on each side and using the fact that $\mathbb{E}_{\theta}[\mathbf{g}(\mathbf{Y})] = \partial \psi(\theta) / \partial \theta$, the MLE can also be expressed as the solution to the system of equations

$$\mathbb{E}_{\hat{\theta}}[\mathbf{g}(\mathbf{Y})] = \mathbf{g}(\mathbf{y}) . \quad (6.35)$$

Unfortunately, the function $\psi(\theta)$, occurring in both (6.34) and (6.35), cannot be evaluated explicitly in any but the most trivial of settings, as it involves the summation in (6.25) over $2^{\binom{N_y}{2}}$ possible choices of \mathbf{y} , for each candidate θ . Therefore, it is necessary to use numerical methods to compute approximate values for $\hat{\theta}$. Two Monte Carlo approaches are commonly used for this purpose, one based on the stochastic approximation of the log-likelihood in (6.34), and the other using a method for the stochastic approximation to solutions of systems of equations, applied to those in (6.35).

The first method, Markov chain Monte Carlo maximum likelihood estimation, derives from fundamental work of Geyer and Thompson [170]. Note that optimization of the log-likelihood in (6.34) is equivalent to optimization of the logarithm of the likelihood ratio

$$\begin{aligned} r(\theta, \theta^{(0)}) &= \ell(\theta) - \ell(\theta^{(0)}) \\ &= (\theta - \theta^{(0)})^T \mathbf{g}(\mathbf{y}) - (\psi(\theta) - \psi(\theta^{(0)})) , \end{aligned} \quad (6.36)$$

for arbitrary, fixed $\theta^{(0)}$. Furthermore, note that

$$\begin{aligned} \exp \left\{ \psi(\theta) - \psi(\theta^{(0)}) \right\} &= \sum_{\mathbf{y}} \exp \left\{ (\theta - \theta^{(0)})^T \mathbf{g}(\mathbf{y}) \right\} \left(\frac{\exp \left\{ (\theta^{(0)})^T \mathbf{g}(\mathbf{y}) \right\}}{\exp \left\{ \psi(\theta^{(0)}) \right\}} \right) \\ &= \mathbb{E}_{\theta^{(0)}} \left[\exp \left\{ (\theta - \theta^{(0)})^T \mathbf{g}(\mathbf{Y}) \right\} \right] . \end{aligned} \quad (6.37)$$

Therefore, an approximation to the term $\psi(\theta) - \psi(\theta^{(0)})$ in (6.36) can be produced by (i) generating a Markov chain Monte Carlo sample $\mathbf{Y}_1, \dots, \mathbf{Y}_n$ from the ERGM (6.24), under $\theta^{(0)}$, (ii) approximating the expectation in (6.37) by the corresponding average based on this sample, and (iii) taking a logarithm of that average. The resulting approximation to the log-likelihood ratio will converge to its target as n tends to infinity, and hence the optimum of this approximate log-likelihood ratio will approximate the MLE $\hat{\theta}$. See Hunter and Handcock [209], for example, for additional details and references.

The second method utilizes the algorithm of Robbins and Monro [324], which may be viewed as a stochastic version of the Newton-Raphson algorithm. Given a random vector \mathbf{Z} , with distribution parameterized by a vector θ , the Robbins-Monro algorithm allows for the solution in θ of the system of equations $\mathbb{E}_{\theta}[\mathbf{Z}] = 0$, through a sequence of iterations of the form

$$\hat{\theta}^{(i+1)} = \hat{\theta}^{(i)} - a_i D_i^{-1} \mathbf{Z}_i . \quad (6.38)$$

Here the \mathbf{Z}_i are a sequence of random vectors for which the distribution of \mathbf{Z}_i , conditional on $\mathbf{Z}_1, \dots, \mathbf{Z}_{i-1}$, is that of \mathbf{Z} at $\theta = \hat{\theta}^{(i)}$; the a_i are a sequence of positive numbers tending to zero; and D_i is a matrix playing the role of the Hessian in the traditional Newton-Raphson algorithm. In the case of $a_i = 1/i$ and \mathbf{Z} distributed according to an exponential family, as in (6.23), the optimal choice of D_i is $\text{Cov}_{\theta}(\mathbf{Z})$, although in practice this quantity is usually approximated. For the task of estimating the MLE $\hat{\theta}$ that solves the system of equations in (6.35) deriving from the ERGM in (6.24), we set $\mathbf{Z} = \mathbf{g}(\mathbf{Y}) - \mathbf{g}(\mathbf{y})$, where \mathbf{Y} is distributed according to (6.24). See Snijders [357] for additional details.

Note that for both of these methods of approximating the MLE, it is necessary to be able to simulate draws of \mathbf{Y} from the ERGM in (6.24). One natural approach to this task is to use the Gibbs sampler, a general Markov chain Monte Carlo method for simulating from the joint distribution of a vector \mathbf{Z} that utilizes only the univariate conditional distributions of each element given all of the others.²² For ERGMs, such conditional distributions have a particularly simple form. Writing $\mathbf{Y}_{(-ij)}$ to be all of the elements of \mathbf{Y} except Y_{ij} , the distribution of Y_{ij} conditional on $\mathbf{Y}_{(-ij)}$ is Bernoulli and satisfies the expression

$$\log \left[\frac{\mathbb{P}_{\theta}(Y_{ij} = 1 | \mathbf{Y}_{(-ij)} = \mathbf{y}_{(-ij)})}{\mathbb{P}_{\theta}(Y_{ij} = 0 | \mathbf{Y}_{(-ij)} = \mathbf{y}_{(-ij)})} \right] = \theta^T \Delta_{ij}(\mathbf{y}) , \quad (6.39)$$

²² We describe Gibbs sampling in somewhat greater detail in Section 8.3.2.2.

where $\Delta_{ij}(\mathbf{y})$ is the *change statistic*, denoting the difference between $\mathbf{g}(\mathbf{y})$ when $y_{ij} = 1$ and when $y_{ij} = 0$, which may be calculated in an efficient manner.²³ Various other MCMC methods are also possible. See Snijders [357] for some discussion.

A disadvantage of both of the methods above is their computationally intensive nature. To date they have been applied to networks with at most a few thousand vertices. An alternative is to estimate θ by maximizing not the actual log-likelihood (6.34), but rather the *log-pseudo-likelihood*

$$\sum_{\{i,j\}} \log \mathbb{P}_\theta(Y_{ij} = 1 | \mathbf{Y}_{(-ij)} = \mathbf{y}_{(-ij)}) . \quad (6.40)$$

This approach, originally proposed by Besag [37] in the context of spatial data analysis, and adapted for ERGMs by Strauss and Ikeda [372], will work best (i.e., in the sense of producing an estimate that reasonably approximates the MLE $\hat{\theta}$) when dependencies among the elements of \mathbf{Y} are relatively weak. Unfortunately, in many network contexts this is not likely to be the case. Nevertheless, pseudo-likelihood estimation is a computationally expedient²⁴ method for obtaining at least some rough sense of the value of $\hat{\theta}$. Historically, this approach has in fact been the primary method of estimation, until the comparatively more recent development of reliable MCMC methods. See Robins et al. [329] for a comparison of results from pseudo-likelihood and MCMC methods, when applied to some commonly studied social networks.

6.5.3 Goodness-of-Fit and Model Degeneracy

In any sort of modeling problem, the best fit chosen from among a class of models need not necessarily be a *good* fit to the data if the model class itself does not contain a sufficiently rich set of models from which to choose. The concept of model *goodness-of-fit* is therefore important. But, while this concept is fairly well developed in standard modeling contexts, such as linear modeling, it is arguably still in its infancy as far as network graph modeling is concerned.

For ERGMs, the current practice in assessing goodness-of-fit is to first simulate numerous random graphs from the fitted model and then compare high-level characteristics of these graphs with those of the originally observed graph. Examples of such characteristics include the distribution of any number of the various summaries of network structure encountered in Chapter 4, such as degree, centrality, and geodesic distance. If the characteristics of the observed network graph are too poor of a match to the typical values arising from realizations of the fitted random graph model (e.g., as viewed by appropriate plots, such as those shown in Section 6.5.4),

²³ In particular, its calculation does not involve $\psi(\theta)$.

²⁴ Maximum pseudo-likelihood estimation here is equivalent to logistic regression of the elements of \mathbf{y} on the design matrix formed by the vectors $\Delta_{ij}(\mathbf{y})$, and may thus be carried out using standard software.

then this suggests systematic differences between the specified class of models and the data, and therefore a lack of goodness-of-fit.²⁵

Goodness-of-fit has been found to be particularly important where ERGMs are concerned, due in large part to the issue of *model degeneracy*. In this context the term is used to refer to a probability distribution that places a disproportionately large amount of its mass on a correspondingly small set of outcomes. See Handcock [193]. A number of simple but originally popular Markov graph models have been shown to be degenerate. A common case is where the ERGM places most of its mass on either the empty graph, the complete graph, or a mixture of the two, depending on the value of θ . None of these, of course, is likely to be especially appropriate for modeling real data of any interest.

In addition to its defining lack of richness, model degeneracy also can lead to difficulties in fitting ERGMs. This is due to the fact that, for degenerate models, the transitions among the limited outcomes (e.g., from null, to mixture, to complete), as θ changes, can be quite sharp. As a result, for poorly specified ERGMs, numerical algorithms like MCMC can have substantial difficulty converging. See Handcock [193], for example, for a careful study of the relationship between model degeneracy and model fitting.

In summary, model specification for ERGMs is clearly a nontrivial task and one that should be approached in a manner closely informed by the above issues, with goodness-of-fit diagnostics playing an important facilitating role. These issues were in fact a major motivation for the proposal of statistics like the alternating k -stars and alternating k -triangles mentioned above. See Snijders, Pattison, Robins, and Handcock [360] for an extensive discussion in this context, including examples and further references.

6.5.4 Case Study: Modeling Collaboration Among Lawyers

In this case study we illustrate the application of exponential random graph models using a dataset on collaborative working relationships among members of a New England law firm, collected by Lazega [251]. Our analysis draws upon original analyses in Hunter and Handcock [209] and Snijders, Pattison, Robins, and Handcock [360].

Lazega's data were collected for the purpose of studying cooperation among social actors in an organization, through the exchange of various types of resources among them. The organization observed was a corporate law firm, consisting of over 70 lawyers (roughly half partners and the other half associates) in three offices located in three different cities. Relational data reflecting resource exchange were collected, and additional attribute information was recorded for each lawyer. See Lazega and Pattison [252] for additional details.

²⁵ This simulation-based approach to evaluating models is actually used more broadly, such as when parameters are simply ‘hand-tuned’ rather than fitted by formal statistical methods.

Here we use only a portion of Lazega's dataset. First, we restrict our attention to only the $N_v = 36$ partners in the firm, among whom the administrative structure was said to be 'relatively decentralized.' Second, we focus upon the network of collaboration among partners, where an edge is present in our network graph if and only if both partners indicated in a survey that they worked together in a substantive manner. Third, we include data on seniority (i.e., the rank entry of the partner's entrance into the law firm), gender, office location, and type of practice (i.e., litigation or corporate law). Figure 6.7 shows a visual representation of this data.

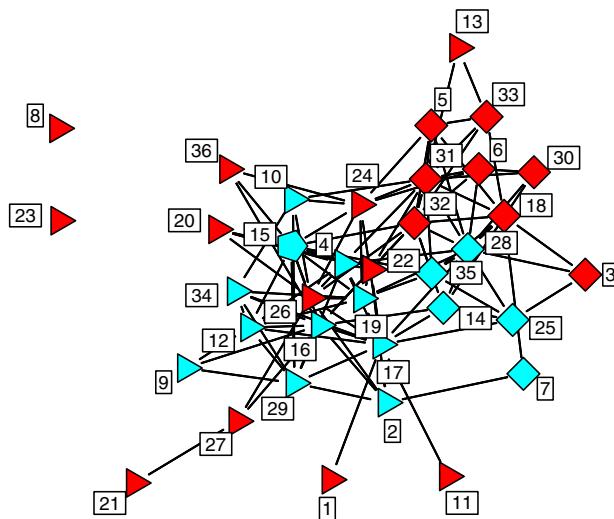


Fig. 6.7 Visualization of Lazega's network of collaborative working relationships among lawyers. Vertices represent partners and are labeled according to their seniority. Vertex shapes (i.e., triangle, square, or pentagon) indicate three different office locations, while vertex colors correspond to the type of practice (i.e., litigation (red) or corporate (cyan)). Edges indicate collaboration between partners. There are three female partners (i.e., those with seniority labels 27, 29, and 34); the rest are male. Data courtesy of Emmanuel Lazega.

It is of interest in a study like this to assess the effects on collaboration of the exogenous attribute variables, while controlling for certain endogenous effects of network structure that might be anticipated, and vice versa. Towards this end, we specify an ERGM like that in (6.24) with two network statistics and five statistics involving attributes. Drawing on the analyses in Hunter and Handcock [209] and Snijders, Pattison, Robins, and Handcock [360] to discard uninformative statistics, the network statistics we chose to incorporate were the number of edges $S_1(\mathbf{y}) = N_e$ and the alternating k -triangles statistic $AKT_k(\mathbf{y})$ defined in (6.33). These allow us

to control for the density of the network and some effects of transitivity. Following Snijders, Pattison, Robins, and Handcock [360], we use a value of $\lambda = 3.0$ in computing the AKT_λ statistic.²⁶

The five variables relating to attributes all are of the form

$$g(\mathbf{y}, \mathbf{x}) = \sum_{1 \leq i < j \leq N_v} y_{ij} h(\mathbf{x}_i, \mathbf{x}_j) , \quad (6.41)$$

where h is a symmetric function of \mathbf{x}_i and \mathbf{x}_j , and \mathbf{x}_i (or \mathbf{x}_j) is the vector of observed attributes for the i -th (or j -th) vertex. Our choice of h falls into two categories, producing ‘main effects’ and ‘second-order effects’ (or similarity or homophily effects) of certain attributes. Main effects are defined for the seniority and practice attributes using a simple additive form:

$$h(\mathbf{x}_i, \mathbf{x}_j) = \text{seniority}_i + \text{seniority}_j \quad (6.42)$$

for seniority and analogously for practice. Second-order effects are defined for practice, gender, and office location, and in each case h is simply an indicator for equivalence of the respective attribute between two vertices, such as $h(\mathbf{x}_i, \mathbf{x}_j) = I\{\text{gender}_i = \text{gender}_j\}$.

The overall model is therefore of the form

$$\mathbb{P}_{\theta, \beta}(\mathbf{Y} = \mathbf{y} | \mathbf{X} = \mathbf{x}) = \left(\frac{1}{\kappa(\theta, \beta)} \right) \exp \left\{ \theta_1 S_1(\mathbf{y}) + \theta_2 AKT_\lambda(\mathbf{y}) + \beta^T \mathbf{g}(\mathbf{y}, \mathbf{x}) \right\} , \quad (6.43)$$

where \mathbf{g} is the vector of five attribute statistics and β is the corresponding vector of parameters. The model was fit using the `statnet` package,²⁷ which implements the MCMC maximum likelihood procedure described in Section 6.5.2, and in addition contains various features to facilitate monitoring of convergence of the underlying Markov chains.

Table 6.1 shows the fitted values for the parameters in our model and an accompanying set of ‘standard errors.’ These standard error values are obtained by applying classical asymptotic theory, which, as was already discussed in Section 6.5.2, is strictly speaking unjustified in the face of the dependencies in our data. As a result, such values and any corresponding statements of significance are generally to be taken as heuristics. Here we see that the evidence of the regression suggests that all effects are potentially ‘significant,’ in the sense of having a non-trivial ability to

²⁶ Alternatively, we might consider treating λ in statistics like (6.31) and (6.33) as an additional parameter in the model, to be estimated from the data. But in this case, the model $\mathbb{P}_\theta(\mathbf{Y} = \mathbf{y})$ no longer satisfies the exponential family form in (6.23). Rather, it is a *curved exponential family*. Hunter and Handcock [209] present a modified form of the MCMC maximum likelihood algorithm described in Section 6.5.2 to fit such network graph models. For the analysis of this particular dataset, they report an estimate $\hat{\lambda}$ that does not appear to differ in a statistically significant manner from the choice $\lambda = 3.0$. However, this is not entirely surprising, given that this λ was chosen in Snijders, Pattison, Robins, and Handcock [360] after comparing results for several different values.

²⁷ Available at <http://csde.washington.edu/statnet>.

Parameter	Estimate	'Standard Error'
Density (θ_1)	-6.2073	0.5697
Alternating k -triangles (θ_2)	0.5909	0.0882
Seniority Main Effect (β_1)	0.0245	0.0064
Practice Main Effect (β_2)	0.3945	0.1103
Same Practice (β_3)	0.7721	0.1973
Same Gender (β_4)	0.7302	0.2495
Same Office (β_5)	1.1614	0.1952

Table 6.1 Results from fit of model (6.43) to the Lazega lawyer data.

explain some of the variation in network structure, since the ratio of each estimate to its standard error is on the order of 3 or more.

Recalling the expression in (6.39), note that the estimated coefficient of each attribute statistic may be interpreted as a conditional log-odds ratio for cooperation. For example, practicing corporate law, rather than litigation, increases the odds of cooperation by a factor of $\exp(0.3945) \approx 1.484$, or nearly 50%. Similarly, being of the same gender more than doubles the odds of cooperation, since $\exp(0.7302) \approx 2.076$. In all cases, such statements hold in the sense of ‘all else being equal’ (i.e., given no change among values of the other statistics).

In terms of network structure, the magnitude of the coefficient $\hat{\theta}_2 \approx 0.5909$ for the alternating k -triangle statistic and the comparatively quite small corresponding standard error indicate that there is strong evidence for a substantial transitivity effect. Note that, given the inclusion of our second-order attribute statistics in the model, our quantification of this effect naturally controls for basic homophily on these attributes. So there is likely something other than similarity of gender, practice, and office at work here – possibly additional attributes we have not controlled for, or possibly social processes of team formation.

In order to assess the goodness-of-fit of our model, we simulated realizations from the model (6.43) with the parameters in Table 6.1. The structure of the resulting random network graphs and the original network graph were compared in terms of the distribution of degree, geodesic length, and edge-wise shared partners (i.e., the number of neighbors shared by a pair of vertices defining an edge). The results are shown in Figure 6.8, and indicate that – on these particular characteristics – the fit of the model is quite good overall.

6.6 Challenges in Modeling Network Graphs

In designing network graph models, it is crucial to maintain sight of the fact that the network topology structure arises in conjunction with the workings of an underlying complex system. A fundamental challenge in this area is to adequately incorporate our understanding of such workings into network graph models that are nevertheless still at least computationally, if not also analytically, tractable.

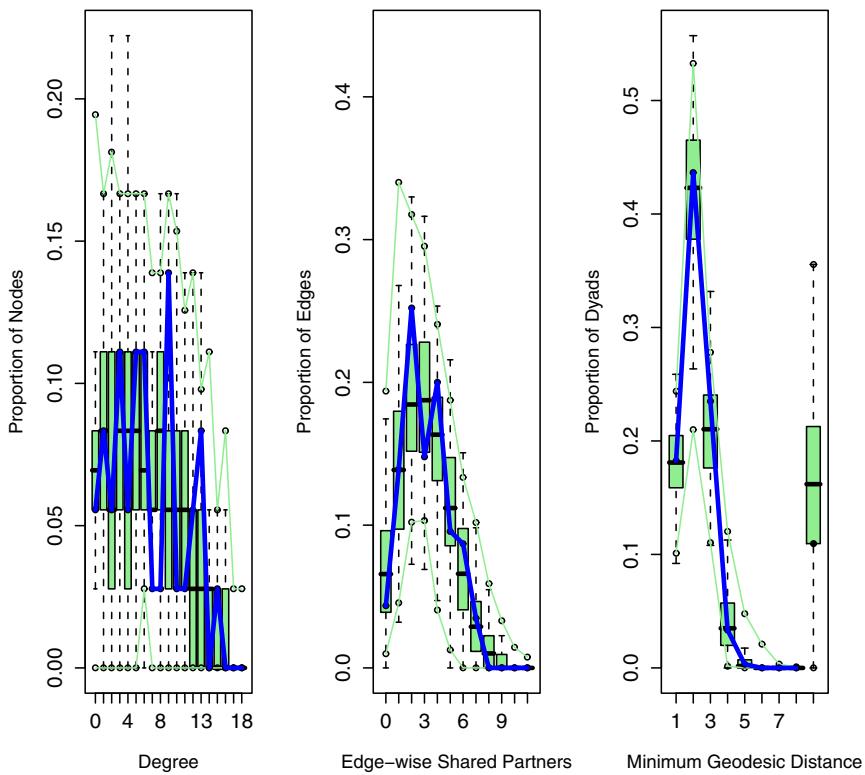


Fig. 6.8 Goodness-of-fit plots comparing original Lazega lawyer network and 100 realizations from the model in (6.43), with the parameters in Table 6.1. Comparisons are made based on the distribution of degree, edge-wise shared partners, and geodesic distance over the 100 realizations, represented by box-plots and curves showing 10-th and 90-th quantiles – both in green. Values for the Lazega network itself are shown with solid blue lines. In the distribution of geodesic distances between pairs, the rightmost box-plot is separate and corresponds to the proportion of nonreachable pairs.

Preferential attachment and copying, for example, are two simple, generic mechanisms that have been somewhat successful in reproducing certain gross structural characteristics of the degree distribution in various real-world networks. Nevertheless, as a result of their generic nature, they arguably capture the internal workings of the systems underlying these networks in only the crudest of manners. Work in this area continues to try to refine mechanisms of this sort.

Similarly, ERGMs too allow for the inclusion of effects of certain simple mechanisms, in their own way, such as transitivity, but they also can incorporate the effects of potentially important vertex attribute information. And, notably, they allow for

some assessment of the strength of association between the observed network topology and the proposed mechanisms and attributes. However, there remains much to be done in this area, especially with respect to model specification and inference.

Alternatively, in contrast to the modeling approaches discussed here, Li, Alderson, Willinger, and Doyle [256] have advocated a more holistic approach to network graph modeling, in which network growth is seen as influenced by various types of constraints and trade-offs. Their ‘first principles’ approach to constructing network graphs, illustrated in the context of modeling the router-level Internet (i.e., the Internet as described in Section 3.5.2), uses a blend of standard statistical data analysis tools and graph theory and, most importantly, various technology constraints and economic factors, to generate networks with topologies much more similar to those observed than produced by other models. However, their overall modeling approach – *heuristically optimal topology* (HOT) – a heuristically driven, non-random rewiring algorithm, necessarily must be taken more as an important demonstration and proof of concept than a formal methodology.

A dynamical systems perspective, which we have not touched upon here in this chapter, has – in principle – much to offer in this direction. The complex systems studied using network methods usually are indeed dynamic and a network graph G frequently is just a snap-shot of system behavior, either at a given point in time or perhaps cumulatively. But the sophisticated use of stochastic dynamical systems models for network formation, particularly in contexts with networks of non-trivial size, is still arguably in its infancy. There remain numerous challenges to be faced, ranging from improved understanding of the dynamics of such models, to how to fit them, to obtaining sufficient data to do so. We will return briefly to this topic area in Chapter 8, when we explore the related issue of modeling dynamic processes indexed on network graphs, in which the graph is assumed fixed and known.

One related area in which some progress has been made is that of agent-based modeling, which has its roots in the intersection of the more quantitative branches of economics, psychology, and sociology. In this approach, a game-theoretic perspective drives the study of network formation, with a particular eye towards how individual incentives balance with the global ‘greater good’ when agents (e.g., social actors) are allowed choice in establishing relationships. See the survey by Jackson [212], for example.

Developments in these various areas of network graph modeling increasingly continue to inform each other, as researchers in these diverse fields interact more and more. Presumably it is in such cross-fertilization that there lies the greatest potential for breakthroughs.

6.7 Additional Related Topics and Reading

There are a number of books devoted entirely to subsets of the topics in network graph modeling discussed in this chapter. The volume by Bollobás [43] is the standard reference for classical random graph theory. The book of Chung and Lu [88]

contains a unified exposition of numerous formal results on generalized random graph, network growth, and small-world models. The book by Dorogovstev and Mendes [123] presents a physics perspective on network growth (a.k.a. ‘evolutionary’) models.

A number of general literature surveys have been written on network graph modeling. Two that we have found particularly useful are the reviews by Newman [296] and by Boccaletti et al. [41]. Both are written from the perspective of the statistical mechanics of complex networks, but with a fairly general audience in mind. For an excellent overview and illustration of ERGMs, with an emphasis on more recent advances, see the special section in the May 2007 issue of the journal *Social Networks*.

While we have attempted to do reasonable justice to the topic of network graph modeling in this chapter – in a not unreasonable amount of space – by concentrating on what to date have been the main model classes of interest, there are of course other classes that call for attention as well. One example is the class of *weighted networks*, wherein the strength of the connection between vertices (i.e., the edge weight) is included in the model. See Boccaletti et al. [41, Sec. 2.4.3] for a short survey of work in this area. Another example is the class of latent variable models, as developed by Nowicki and Snijders [304], Hoff [199], and Hoff, Raftery, and Handcock [202], which augment the basic Bernoulli model in (6.26) with a second layer of modeling for the parameters θ_{ij} . We will see an example of such models in Chapter 7 in the context of predicting edges missing from a partially observed graph. Also, there are dynamic extensions of ERGMs and related models, in which a sequence of observed network graphs is modeled, for example through the addition of a Markov component in time. See Snijders [358], for example, for a review, and also Koskinen and Snijders [236].

Finally, on the topic of ERGMs, being as they are probabilistic models with conditional dependency relationships defined in terms of graph structure, we would be remiss not to point out here that they are, in fact, one example of the more general class of models known as *graphical models*. Technical tools like the Hammersley-Clifford theorem cited earlier in this chapter are fundamental to this area. We will encounter a handful of additional specific instances of graphical models over the course of the next three chapters, and in Chapter 10 we will then look briefly at the general topic of graphical models in its own right.

Exercises

6.1. Recall the problem of model-based estimation of the size of a ‘hidden population,’ as described in Example 6.3.

- a. Verify the expressions for the expectations in (6.4), (6.5), and (6.6).
- b. Show that the corresponding method-of-moments equations

$$\begin{aligned} n &= N_v p_0 \\ m_1 &= N_v (N_v - 1) p_0^2 p_{\mathcal{G}} \\ m_2 &= N_v (N_v - 1) p_0 (1 - p_0) p_{\mathcal{G}} \end{aligned}$$

yield the estimates \hat{p}_0 , $\hat{p}_{\mathcal{G}}$, and \hat{N}_v of p_0 , $p_{\mathcal{G}}$, and N_v given in equations (6.7), (6.8), and (6.9), respectively.

6.2. A semi-rigorous proof of the limiting power-law behavior of the degree distribution for the Barabási-Albert preferential attachment model rests upon the use of so-called ‘rate equations.’

- a. By considering the expected change in the number of vertices of degree d , from iteration t to iteration $t + 1$, argue that for $d > m$,

$$(t+1) f_d^{(t+1)} - t f_d^{(t)} = \frac{(d-1) f_{d-1}^{(t)}}{2} - \frac{d f_d^{(t)}}{2} ,$$

and for $d = m$,

$$(t+1) f_m^{(t+1)} = 1 - \frac{m f_m^{(t)}}{2} .$$

Here $f_d^{(t)} \equiv f_d(G^{(t)})$.

- b. Under the assumption that $f_d^{(t)}$ tends to the value f_d (i.e., $\lim_{t \rightarrow \infty} f_d^{(t)} = f_d$), for each d , with respect to some limiting distribution $\{f_d\}$, use part (a) to argue that

$$f_d = \begin{cases} \frac{1}{2}(d-1) f_{d-1} - \frac{1}{2} d f_d, & \text{if } d > m, \\ 1 - \frac{1}{2} m f_m, & \text{if } d = m . \end{cases}$$

- c. Use this result to conclude that $f_d = f_{d,m}$ is given by the expression in (6.17).

6.3. We consider here the problem of estimating the exponent of a power-law degree distribution, as described in Section 4.2.1.1.

- a. Implement one of the network growth models described in Section 6.4, so as to be able to simulate realizations of your model using a computer. For example, the LCD preferential attachment model can be implemented using the algorithm of

Batagelj and Brandes [23]. For various choices of t , compare the average degree distribution obtained with the distribution of the limiting power law.

- b.** Explore the accuracy with which you can estimate the power-law exponent from a single realization of your model, for sufficiently large t , using the Hill estimator (4.4). Compare the performance of this estimator with that of the naive regression estimator motivated by the relation in (4.2) and the logarithm of the relation in (4.3).

6.4. Recall the important role that the log-odds ratio plays in the fitting and interpretation of ERGMs.

- a.** For the exponential random graph model in (6.24), verify that the log-odds ratio

$$\log \left[\frac{\mathbb{P}_\theta(Y_{ij} = 1 | \mathbf{Y}_{(-ij)} = \mathbf{y}_{(-ij)})}{\mathbb{P}_\theta(Y_{ij} = 0 | \mathbf{Y}_{(-ij)} = \mathbf{y}_{(-ij)})} \right] = \boldsymbol{\theta}^T \Delta_{ij}(\mathbf{y}) ,$$

as stated in equation (6.39), where $\Delta_{ij}(\mathbf{y})$ is the change statistic.

- b.** In the case of the Markov graph model (6.30), with terms $S_1(\mathbf{y})$, $S_2(\mathbf{y})$, and $T(\mathbf{y})$ in the model, show that the change statistic is given by

$$\Delta_{ij}(\mathbf{y}) = \begin{pmatrix} 1 \\ y_{i+} + y_{j+} - 2y_{ij} \\ \sum_{k \neq i,j} y_{ik}y_{kj} \end{pmatrix} .$$

6.5. We saw two methods – in Example 6.4 and in the case study of Section 6.5.4 – for assessing the evidence for transitivity in a given network while controlling for certain other factors. For a network of your choice, and using appropriate software, assess the evidence for transitivity in your network using one or both of these methods. Report on your findings and discuss any implications of these findings on the system underlying your network.

Chapter 7

Network Topology Inference

In contrast to the previous chapter, with its focus on modeling and inference of observed network graphs, in this chapter we consider the problem of network topology inference, wherein the graph or some portion thereof is unobserved and we wish to infer it from measurements. There are a number of variations on this problem; we examine three particular forms in some depth.

7.1 Introduction

In Chapter 3, as part of our study of network mapping, we discussed the process of constructing a network graph representation for a system of interest based on a set of measurements from that system. Recall that this process, as described, was largely informal, with user-specified decisions and rules (presumably informed by expert knowledge) dictating the definition of what should constitute a vertex and an edge in the resulting graph. Such practice is typical – in fact, it is arguably the manner in which the vast majority of network graph representations are created.

As a result of its inherent informality, however, this manner of network graph construction distinctly lacks an element of validation, which can potentially be quite important depending on the context and the intended usage of the network graph. See the discussion in Section 3.7, for example. By ‘validation’ we have in mind issues like whether the network graph representation is ‘accurate,’ in terms of capturing some well-defined but unobservable relational structure in the underlying system; what accuracy ideally can be expected, given the information in the available set of measurements; whether there are other similar representations that are equally or almost as accurate; how robust the representation is to small changes in the measurements; and how useful the representation is as a basis for other, perhaps higher-level, purposes.

One natural way to endow the process of network graph construction with a greater sense of formality, and to facilitate a more precise formulation and study of such issues of validation, is to consider the task of constructing a network graph

representation from data as one of statistical inference. To be more precise, suppose that, broadly speaking, we have a set of measurements from a system of interest, such as vertex attributes $\mathbf{x} = (x_1, \dots, x_{N_v})^T$ or binary indicators $\mathbf{y} = [y_{ij}]$ of certain edges but not others, or both \mathbf{x} and \mathbf{y} , and we have a collection \mathcal{G} of potential network graphs G . We might then take as our goal to select an appropriate member of \mathcal{G} that best captures the underlying state of the system, based upon the information in the data as well as any other prior information, using techniques of statistical modeling and inference. That is, we might pose the problem as one of *network topology inference*. In doing so, we position ourselves to potentially bring to bear a collection of accompanying concepts and tools – such as statistical consistency, efficiency, and robustness – on the problem of validation.

Unfortunately, there is at present no single coherent body of formal results on inference problems of this type, and certainly not at the level of generality we have posed. Rather, such problems have been pursued to varying extents in a handful of different areas. And the frameworks developed in these settings naturally take various forms, as dictated by context, with differences driven primarily by the nature of the topology to be inferred and the type of data available.

In this chapter, we will focus on discussing three particular classes of problems that have emerged. Each class of problems is somewhat ‘canonical’ in nature, being easily posed in more than just a single specific network context. More specifically, in Section 7.2 we consider the problem of inferring whether or not a pair of vertices does or does not have an edge between them (i.e., inference of ‘edge’ or ‘non-edge’ status), using measurements that include a subset of vertex pairs whose edge/non-edge status is already observed. This problem is commonly referred to as *link prediction*. Next, in Section 7.3, we discuss the inference of association networks. Here the relation defining edges is taken by definition to be a nontrivial level of association (e.g., correlation) between certain characteristics of the vertices, but is itself unobserved and must be inferred from measurements reflecting these characteristics. Finally, we examine problems of tomographic network inference in Section 7.4. These problems are distinguished by the fact that measurements are available only at vertices that are somehow at the ‘perimeter’ of the network, and it is necessary to infer the presence or absence of both edges and vertices in the ‘interior.’

The ordering of these three classes of problems has a general progression. The first assumes knowledge of all of the vertices and the status of some of the edges/non-edges of the network graph, and seeks to infer the rest of the edges/non-edges. The second starts with no knowledge of edge status anywhere in the network graph, but assumes relevant measurements at all of the vertices and seeks to infer edge status throughout the network using these measurements. The third involves measurements at only a particular subset of vertices, which nevertheless indirectly provide some information useful for inferring the unknown topology of the rest of the network graph. A visual characterization of these three types of problems is shown in Figure 7.1.

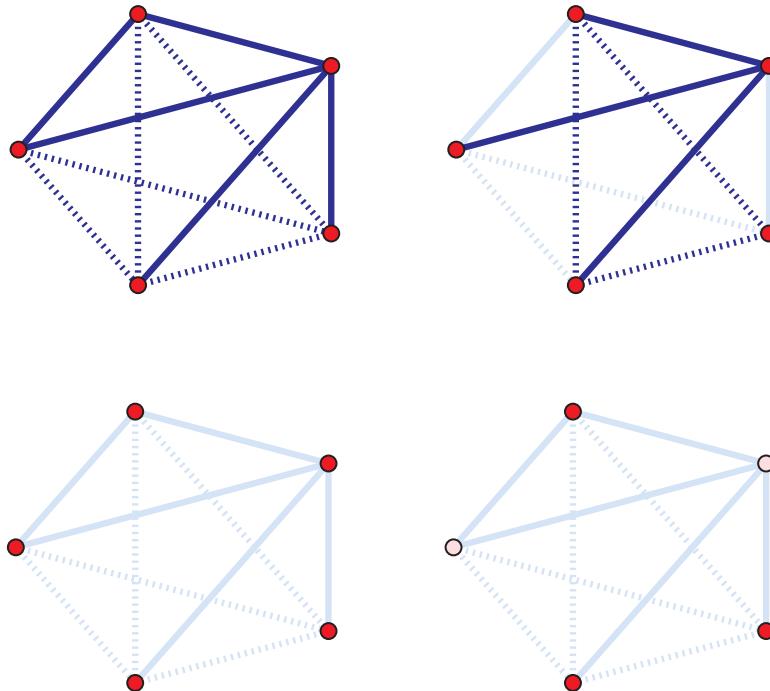


Fig. 7.1 Visual characterization of three types of network topology inference problems, for a toy network graph G . Edges shown in solid; non-edges, dotted. Observed vertices and edges shown in dark (i.e., red and blue, respectively); un-observed vertices and edges, in light (i.e., pink and light blue). Top left: True underlying graph G . Top right: Link prediction. Bottom left: Association graph inference. Bottom right: Tomographic network inference.

7.2 Link Prediction

Let $G = (V, E)$ be an undirected random network graph, and assume for simplicity that all vertices $v \in V$ are known. Recall that $V^{(2)}$ is the set of distinct unordered pairs of vertices. As such, it is (trivially) the union of (i) the set E of edges in G , and (ii) the set $V^{(2)} \setminus E$ of non-edges in G . In this section, we will suppose that among the vertex pairs in $V^{(2)}$, the presence or absence of an edge is observed only for some subset of pairs, say $V_{\text{obs}}^{(2)}$. For the remaining pairs, say $V_{\text{miss}}^{(2)} = V^{(2)} \setminus V_{\text{obs}}^{(2)}$, this information is missing.

Let \mathbf{Y} be the random $N_v \times N_v$ binary adjacency matrix for the full network graph G , and denote by \mathbf{Y}^{obs} and \mathbf{Y}^{miss} the entries of \mathbf{Y} corresponding to the vertex pairs in $V_{\text{obs}}^{(2)}$ and $V_{\text{miss}}^{(2)}$, respectively. The problem of *link prediction* is to predict the entries in \mathbf{Y}^{miss} , given the values $\mathbf{Y}^{\text{obs}} = \mathbf{y}^{\text{obs}}$ and possibly various vertex attribute variables $\mathbf{X} = \mathbf{x} \in \mathbb{R}^{N_v}$. In other words, we wish to predict whether ‘potential edges’ between

pairs of vertices in a network graph are present or absent using information provided by a subset of observed edges/non-edges and, if available, vertex attributes.

There are a number of variants of the link prediction problem that have been formulated, arising in settings ranging from information networks (e.g., Liben-Nowell and Kleinberg [261], Popescul and Ungar [314], Taskar, Wong, Abbeel, and Koller [376]), to social networks (e.g., Hoff [200]), to biomolecular networks (e.g., Goldberg and Roth [178], Bader, Chaudhuri, Rothberg, and Chant [16]). Besides differing in context, these variants of the problem can also differ, importantly, in why and how the values in \mathbf{Y}^{miss} are missing. Sometimes there is simply a temporal component to the problem and, for example, edges may be ‘missing’ only in the sense that they are absent up to a certain point in time and then become present, as in Liben-Nowell and Kleinberg [261] and their study of the growth of the World Wide Web network graph. In many cases, however, all edges and non-edges effectively coincide in time, but the status of potential edges is missing due to issues of sampling. In this latter case the underlying mechanism of missingness can be important.

A common assumption (e.g., Hoff [200], Taskar, Wong, Abbeel, and Koller [376]), and one we shall make here as well, is that the missing information on edge presence/absence is *missing at random*. This assumption means, essentially, that the probability that an edge variable Y_{ij} is observed depends only on the values of those other edge variables observed and not, for example, on its own value.¹ If edge variables Y_{ij} are missing with probability depending upon themselves, this is called *informative missingness*. One example of informative missingness is arguably in the context of certain biological networks, where edges are sometimes derived from databases summarizing the known results in the literature (e.g., gene 1 strongly influences gene 2, protein a has a strong affinity for protein b). Because of the well-documented bias towards ‘positive’ results in scientific literatures, there is a clear dependency of whether Y_{ij} is observed on whether or not it indicates a true edge. Informative missingness can render methods like those discussed below less effective, and calls for the extension of such methods to include, for example, additional modeling for the mechanism(s) of missingness. We will not pursue such extensions here.

In principle, given an appropriate model for \mathbf{X} and $(\mathbf{Y}^{obs}, \mathbf{Y}^{miss})$, we might aim to jointly predict the elements of \mathbf{Y}^{miss} based on a model for

$$\mathbb{P}\left(\mathbf{Y}^{miss} \mid \mathbf{Y}^{obs} = \mathbf{y}^{obs}, \mathbf{X} = \mathbf{x}\right). \quad (7.1)$$

But serious pursuit of this strategy entails, in part, successfully meeting the various challenges of modeling network graphs already described in Chapter 6, and in addition potentially modeling the missingness in an appropriate fashion. Perhaps not surprisingly, therefore, to date most efforts in this area have instead focused upon the comparatively more manageable task of predicting the variables Y_{ij}^{miss} individually.

¹ See Little and Rubin [263], for example, for a more formal definition and a general introduction to such missing data concepts.

Not only does this approach simplify matters considerably in many ways, but it also appears to in fact be a strong competitor to methods that predict the variables Y_{ij}^{miss} jointly. See Taskar, Wong, Abbeel, and Koller [376], for example. The methods we consider here in this section are designed for the prediction of individual Y_{ij}^{miss} .

7.2.1 Informal Scoring Methods

Scoring methods – methods based on the use of score functions – although somewhat less formal than the model-based methods we will discuss momentarily, are nevertheless popular and can be quite effective. Their essential element, a score function, can also be incorporated into the model-based methods as explanatory variables. Therefore, scoring methods serve as a useful starting point for our discussion.

With scoring methods, for each pair of vertices i and j whose edge status is unknown (i.e., for each potential edge $\{i, j\} \in V_{miss}^{(2)}$), a score $s(i, j)$ is computed. A set of predicted edges may then be returned either by applying a threshold s^* to these scores, for some fixed s^* , or by ordering them and keeping those pairs with the top n^* values, for some fixed n^* .

There are many types of scores that have been proposed in the literature. They generally are designed to assess certain structural characteristics of a network graph $G^{obs} = (V^{obs}, E^{obs})$ associated with $\mathbf{Y}^{obs} = \mathbf{y}^{obs}$. A simple score, inspired by the ‘small-world’ principle, is negative the shortest-path distance between i and j ,

$$s(i, j) = -\text{dist}_{G^{obs}}(i, j) . \quad (7.2)$$

The negative sign in (7.2) is present so as to have larger score values indicate vertex pairs more likely to share an edge. There are also a number of scores based on comparison of the observed neighborhoods \mathcal{N}_i^{obs} and \mathcal{N}_j^{obs} of i and j in G^{obs} , including the number of common neighbors

$$s(i, j) = |\mathcal{N}_i^{obs} \cap \mathcal{N}_j^{obs}| , \quad (7.3)$$

a standardized version of this value, called the *Jaccard coefficient*,

$$s(i, j) = \frac{|\mathcal{N}_i^{obs} \cap \mathcal{N}_j^{obs}|}{|\mathcal{N}_i^{obs} \cup \mathcal{N}_j^{obs}|} , \quad (7.4)$$

and a variation on this idea due to Liben-Nowell and Kleinberg [261], extending a more general idea of Adamic and Adar [4], of the form

$$s(i, j) = \sum_{k \in \mathcal{N}_i^{obs} \cap \mathcal{N}_j^{obs}} \frac{1}{\log |\mathcal{N}_k^{obs}|} . \quad (7.5)$$

This last score in (7.5) has the effect of weighting more heavily those common neighbors of i and j that are themselves not highly connected.

Most of the scores above assess only relatively local structure in G^{obs} . Continuing this line of thought, however, it is easy to conceive of higher-order summaries to use. For example, a variety of methods of scoring may be defined through different spectral characteristics of G^{obs} , analogous to many of those we have seen in Chapter 4. See Liben-Nowell and Kleinberg [261], for example.

Note the relevance that the mechanism of missingness can have here, with respect to the fact that the topology of G^{obs} is assumed to be somehow predictive. For example, if edges are ‘missing’ only because they can arise in the future, there is the implicit assumption in using the score (7.2) that vertices closer in the original period of observation will be more likely themselves to form a tie in the future, while the neighborhood-based scores (7.3) and (7.4) assume that two vertices originally having more neighbors in common makes the formation of an edge between them later more likely. The impact of different mechanisms of missingness on score function methods has yet to be explored in depth in the literature.

7.2.2 Probabilistic Classification Methods

As an alternative to informal scoring methods, it is appealing to approach link prediction as a classification problem. On the surface at least, to do so is relatively straightforward. The entries of $\mathbf{Y}^{obs} = \mathbf{y}^{obs}$ are taken as training examples of cases of two types (i.e., 0’s and 1’s), and the goal is to use the information in \mathbf{y}^{obs} and any vertex attributes to build a classifier, which in turn is then used to predict the entries in \mathbf{Y}^{miss} . Effectively, we aim to ‘learn’ an approximation to the rule dictating the presence and absence of edges in the network, and to use that approximation to predict those edges/non-edges we have not yet observed. Although in principle any type of standard classification method can be applied, classifiers based on logistic regression are common and we will build our discussion around this particular choice.

7.2.2.1 Logistic Regression Classifiers

Logistic regression classifiers in this context are based on models of the form

$$\log \left[\frac{\mathbb{P}_\beta(Y_{ij} = 1 | \mathbf{Z}_{ij} = \mathbf{z})}{\mathbb{P}_\beta(Y_{ij} = 0 | \mathbf{Z}_{ij} = \mathbf{z})} \right] = \beta^T \mathbf{z} , \quad (7.6)$$

where \mathbf{Z}_{ij} is a vector of explanatory variables indexed in the unordered pairs $\{i, j\}$, and β is a vector of regression coefficients, assumed common to all pairs. Standard techniques, based on an efficient iteratively re-weighted least-squares algorithm, may be used to produce maximum likelihood estimates $\hat{\beta}$ of β . Details may be

found in Hastie, Tibshirani, and Friedman [194, Ch. 4.4], for example. Potential edges $\{i, j\}$ are then classified as being present or absent according to whether or not the estimated classification probabilities

$$\mathbb{P}_{\hat{\beta}}(Y_{ij}^{miss} = 1 | \mathbf{Z}_{ij} = \mathbf{z}) = \frac{\exp(\hat{\beta}^T \mathbf{z})}{1 + \exp(\hat{\beta}^T \mathbf{z})} \quad (7.7)$$

exceed a threshold (e.g., 0.5). If more than one model is entertained, standard techniques for model selection may be used to select an appropriate model. See Hastie, Tibshirani, and Friedman [194, Ch. 7], for example, for a description of such techniques in general regression and classification problems.

The \mathbf{Z}_{ij} here are vectors of functions of the form

$$\mathbf{Z}_{ij} = \left(g_1(\mathbf{Y}_{(-ij)}^{obs}, \mathbf{X}), \dots, g_K(\mathbf{Y}_{(-ij)}^{obs}, \mathbf{X}) \right)^T, \quad (7.8)$$

where $\mathbf{Y}_{(-ij)}^{obs}$ is defined to be all elements of \mathbf{Y}^{obs} other than Y_{ij} . Note that the exclusion of Y_{ij} is only relevant to the training stage of our classification process (i.e., when the pair $\{i, j\}$ is among those observed). Such functions should be chosen to encode useful predictive information in $\mathbf{Y}^{obs} = \mathbf{y}^{obs}$ and $\mathbf{X} = \mathbf{x}$. For example, they may extract the vertex-specific attributes x_i and x_j from \mathbf{x} , or simple transformations thereof. Or they may summarize structural information in a graph G^{obs} derived from \mathbf{y}^{obs} , in the manner of the scores described in Section 7.2.1. Or, finally, they may combine information in $\mathbf{y}_{(-ij)}^{obs}$ and \mathbf{x} , in analogy to (6.41). We provide some specific examples below in the illustration of Section 7.2.3.

Two issues are important to note in regard to the procedure just described. First, the standard logistic regression framework assumes that the observations in the training data are independent, conditional on the explanatory variables. However, as we have already discussed in Chapter 6, the Y_{ij} can typically be expected to be dependent upon each other in a nontrivial fashion, beyond what is specified in a simple logistic model. In fact, the use of logistic regression is essentially equivalent to the pseudo-likelihood approach associated with (6.40). However, there is no formal work to date exploring the implications on prediction accuracy of ignoring possible dependencies in this manner. Second, as with the informal scoring methods of Section 7.2.1, the nature of the underlying mechanism of missingness is relevant. The use of classification in the link prediction problem is reminiscent of cross-validation, and the assumption that data are missing at random is fundamental to the predictive accuracy of cross-validation procedures.² Therefore, presumably, if the missingness

² *Cross-validation* is a simple and general method for estimating prediction error in regression and classification problems. It seeks to mimic the ideal prediction setup, where we have available separate training and testing datasets of independent and identically distributed cases, in the context where we cannot or do not wish to separate the data in such fashion. It does so by dividing the data into K subsets – called ‘folds’ – and alternately using one fold to validate predictions made by a model fit to the other folds. The prediction errors on the K folds can then be used to produce an estimate of the overall prediction error of the underlying methodology, which in turn can be used

underlying \mathbf{Y}^{miss} is not at random, the accuracy of the classification approach will suffer, although we might hope that the classification probabilities still provide a useful ordering of potential edges.

The issue of missingness is a topic on which, in the particular context of link prediction, little work has been done to date. However, the issue of dependence among the Y_{ij} can in principle be addressed by appropriate use of additional model structure, as we describe next.

7.2.2.2 Logistic Regression with Latent Variables

The use of latent variables is an intuitively appealing way to indirectly model unobserved factors driving the formation of network structure. We describe a very general class of latent variable models for this purpose, due to Hoff [200, 201], that extends the logistic regression framework above.

Let \mathbf{M} be an unknown random, symmetric $N_v \times N_v$ matrix of the form

$$\mathbf{M} = \mathbf{U}^T \Lambda \mathbf{U} + \mathbf{E} , \quad (7.9)$$

where $\mathbf{U} = [\mathbf{u}_1, \dots, \mathbf{u}_{N_v}]$ is a random orthonormal matrix, Λ is an $N_v \times N_v$ random diagonal matrix, and $\mathbf{E} = [\varepsilon_{ij}]$ is a matrix of independent and identically distributed noise variables, with $\varepsilon_{ij} = \varepsilon_{ji}$, so as to enforce symmetry. Each element of \mathbf{M} therefore has the form

$$M_{ij} | \mathbf{U}, \Lambda, \mathbf{E} = \mathbf{u}_i^T \Lambda \mathbf{u}_j + \varepsilon_{ij} . \quad (7.10)$$

Now, augment each observed explanatory variable \mathbf{Z}_{ij} by the unobserved variable M_{ij} , so that (7.6) becomes

$$\log \left[\frac{\mathbb{P}_\beta (Y_{ij} = 1 | \mathbf{Z}_{ij} = \mathbf{z}, M_{ij} = m)}{\mathbb{P}_\beta (Y_{ij} = 0 | \mathbf{Z}_{ij} = \mathbf{z}, M_{ij} = m)} \right] = \beta^T \mathbf{z} + m . \quad (7.11)$$

Although the Y_{ij} are still assumed conditionally independent, given the \mathbf{Z}_{ij} and M_{ij} , they are now conditionally *dependent* given only the \mathbf{Z}_{ij} , in contrast to the model in (7.6). This property follows as a result of the hierarchical nature of this model. The latent variable matrix \mathbf{M} is intended to capture effects of network structural characteristics or processes not already described by the observed explanatory variables \mathbf{Z}_{ij} . Hoff [201] shows that the form of \mathbf{M} in (7.9) includes other previously proposed latent variable models, such as those aimed at capturing effects of homophily (e.g., Hoff, Raftery, and Handcock [202]) or stochastic notions of equivalence (e.g., Nowicki and Snijders [304]).

In order to produce statistical predictions using this model it is necessary to additionally specify distributions for \mathbf{U} , Λ , and \mathbf{E} . If a Bayesian approach to inference is to be used, which is natural here, a prior distribution for β is also needed. An intuitive choice for modeling \mathbf{U} is the uniform distribution on the space of all $N_v \times N_v$

for tasks like variable selection and setting tuning parameters. For additional details, see Hastie, Tibshirani, and Friedman [194, Ch. 7.10], for example.

orthonormal matrices. For the other variables, multivariate Gaussian distributions are convenient, due to the manner in which they facilitate MCMC sampling of the relevant posterior distribution through conjugacy properties. See Hoff [200, 2011], for example. In predicting the values of elements in \mathbf{Y}^{miss} , it is natural to compare the values

$$\mathbb{E} \left(\frac{\exp\{\beta^T \mathbf{Z}_{ij} + M_{ij}\}}{1 + \exp\{\beta^T \mathbf{Z}_{ij} + M_{ij}\}} \mid \mathbf{Y}^{obs} = \mathbf{y}^{obs}, \mathbf{Z}_{ij} = \mathbf{z} \right) \quad (7.12)$$

to a given threshold, in analogy to our use of (7.7) previously. These values may be approximated numerically to any desired accuracy by the corresponding sample average of draws from the posterior indicated.

Note that the use of MCMC sampling noticeably increases the computational cost over the standard logistic regression method, which restricts the utility of this approach to networks of small to moderate size. This cost is primarily driven by the need to generate draws of the N_v^2 unobserved variables $\{U_{ij}\}$ in the process of sampling to approximate (7.12). But a significant reduction in cost can be obtained by specifying that \mathbf{U} have only K non-zero column vectors, for some $K \ll N_v$, which constrains \mathbf{M} to have an effective rank of K . Hoff [200, 2011] has found values of K as small as 2 or 3 to work well in practice, which is of some interest in its own right.

7.2.3 Case Study: Predicting Lawyer Collaboration

Recall the network of collaborative working relationships among lawyers analyzed in Section 6.5.4. We will use the same data to illustrate the relative performance of some of the link prediction methods we have discussed through the device of a cross-validation study.

The set of $36(36 - 1)/2 = 630$ distinct vertex pairs in the network graph in Figure 6.7 were randomly divided into five separate subsets. For each subset in turn, the presence or absence of its 126 potential edges were taken to constitute the variables in \mathbf{Y}^{miss} , while the status of potential edges in the remaining four subsets was considered observed and defined \mathbf{Y}^{obs} . For each link prediction method examined here, predictions were generated for the elements of \mathbf{Y}^{miss} , using the information in \mathbf{Y}^{obs} and, in some cases, the vertex attribute information on seniority, practice, gender, and office.

Four different methods were implemented for this illustration. Method 1 is the standard logistic regression in (7.6), where the explanatory variables for Y_{ij} were defined, in analogy to those in Section 6.5.4, to be

$$\begin{aligned} Z_{ij}^{(1)} &= \text{seniority}_i + \text{seniority}_j \\ Z_{ij}^{(2)} &= \text{practice}_i + \text{practice}_j \\ Z_{ij}^{(3)} &= \text{Indicator of } \{\text{practice}_i = \text{practice}_j\} \\ Z_{ij}^{(4)} &= \text{Indicator of } \{\text{gender}_i = \text{gender}_j\} \end{aligned}$$

$$Z_{ij}^{(5)} = \text{Indicator of } \{\text{office}_i = \text{office}_j\} .$$

Method 2 is also standard logistic regression, but with the above set of explanatory variables augmented by the additional variable

$$Z_{ij}^{(6)} = |\mathcal{N}_i^{obs} \cap \mathcal{N}_j^{obs}| . \quad (7.13)$$

This variable counts the number of neighbors common to vertices i and j in the graph G^{obs} induced by the vertex set V and the observations in \mathbf{Y}^{obs} . Method 3 is an informal scoring method, in which predictions were made based on the ranking of potential edges according to the number of common neighbors of their incident edges, as in (7.3), that is, based on their scores $s(i, j) = Z_{ij}^{(6)}$. Finally, method 4 implements the model of Hoff³ in the form described above, using the same set of explanatory variables \mathbf{Z}_{ij} as in method 1. The number of nontrivial columns in \mathbf{U} was set to be $K = 3$.

Figure 7.2 shows receiver operating characteristic (ROC) curves⁴ summarizing the predictive performance of the four methods as a function of an underlying threshold value. For methods 1, 2, and 4, this threshold is simply a value t ranging from 0 to 1, while for method 3, the threshold is the number n^* of vertex pairs predicted as having an edge. Each curve is an average over the five cross-validation folds. Since the network density was slightly under 0.2, in each fold the corresponding \mathbf{Y}^{miss} had roughly 25 true edges and 100 true non-edges to be predicted.

All four methods can be seen to have performed noticeably better than random guessing (indicated by the gray line). The worst performance is observed for method 1, based on logistic regression using vertex attributes alone. The other three methods are quite similar in their performance, with the scoring method doing arguably just slightly worse than the other two. Comparison of the average area under the curve⁵ for each of the four methods, as shown in Table 7.1, confirms this ordering. The improvement of methods 2, 3, and 4 over method 1 is indicative of the benefit derived from incorporating information on network structure. The fact that method 3 uses only network structure, whereas methods 2 and 4 use both network structure and vertex attributes, suggests that network structure alone is the more powerful explanatory factor here. This is confirmed by fitting models 2 and 4 without vertex attributes, which yields curves similar to that of method 3, though we do not show the results here.

³ Computations were performed using the R package *eigenmodel*, available at <http://cran.r-project.org>, through the Comprehensive R Archive Network.

⁴ An ROC curve relates the fraction of true positive predictions (y-axis) to the fraction of false positive predictions (x-axis) made by a binary classifier, as a parameter controlling the discrimination between positives and negatives is varied. Classifiers with curves pushing more into the upper left-hand corner are generally considered more desirable.

⁵ The area under the curve (AUC) is a standard method for comparing ROC curves. It is maximized at 1 in the case of a perfect classifier, and is expected to be 0.5 in the case of random guessing.

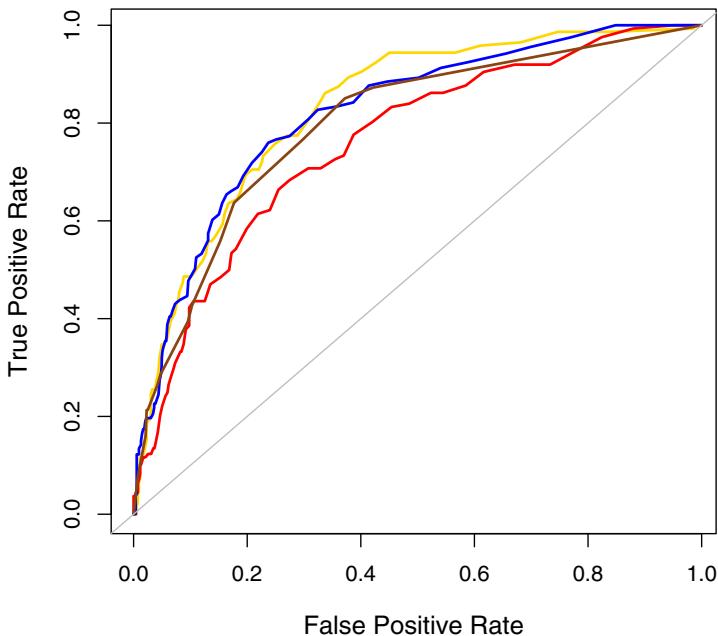


Fig. 7.2 ROC curves summarizing the capabilities to predict collaborative working relationships in the lawyer dataset of Section 6.5.4, for method 1 (red), based on logistic regression, with the explanatory variables $Z^{(1)}$ through $Z^{(5)}$, method 2 (blue), which is method 1 augmented with the variable $Z^{(6)}$, method 3 (brown), an informal scoring method based on scores $s(i, j) = Z_{ij}^{(6)}$, and method 4 (yellow), the method of Hoff [200, 201], using the same variables as in method 1.

	Method 1	Method 2	Method 3	Method 4
AUC	0.7638	0.8229	0.7963	0.8204

Table 7.1 Values of the area under the curve (AUC) for the ROC curves in Figure 7.2, averaged over five folds.

7.3 Inference of Association Networks

In Chapter 3 we saw that often the rule used for defining edges in the network graph representation of a given data set is that there be a sufficient level of ‘association’ between certain attributes of the two incident vertices. Such *association networks*

are found in many domains and include networks of citation patterns across scientific articles, networks of actors co-starring in movies, and networks of regulatory influence among genes. Frequently the rules defining edges in such networks are specified without statistics necessarily playing an explicit role. For example, recall the case study of Section 3.5.1, where the network of ‘Science’ shown in Figure 3.6 derives from use of the Jaccard measure of similarity in (3.1) and a combination of *ad hoc* thresholding and expert-guided clustering rules applied thereto.

While such rule-based approaches are appropriate in many contexts, in other contexts, where issues of sampling or measurement error are of potential concern, it may be necessary to incorporate statistical principles and methods into the process of constructing an association network graph. Generally speaking, we may suppose we have a collection of elements represented as vertices $v \in V$. Furthermore, suppose that each such vertex v has corresponding to it a vector $\mathbf{x} \in \mathbb{R}^m$ of observed vertex attributes, yielding a collection $\{\mathbf{x}_1, \dots, \mathbf{x}_{N_v}\}$ of attribute vectors. Let $\text{sim}(i, j)$ be a user-specified quantification of the inherent similarity between the pair of vertices $i, j \in V$, and assume that it is accompanied by a corresponding notion of what value(s) of $\text{sim}(i, j)$ constitute a ‘non-trivial’ level of association between i and j . We are interested here in the case where sim is not itself directly observable, but nevertheless the attributes $\{\mathbf{x}_i\}$ contain sufficiently useful information to make inference on sim conceivable.

Example 7.1 (Inference of Gene Regulation from Microarray Data). Regulatory interactions among genes are fundamental to the workings of biological organisms. Networks are a popular and useful tool for describing the patterns of known regulatory interactions, as we have already observed in the discussions surrounding Figures 1.3 and 3.2. Recall that, in this context, vertices represent genes and edges typically represent some notion of a regulatory relationship (e.g., activation or repression). Unfortunately, it is infeasible to construct such networks on a large-scale basis using traditional experimental techniques. But with the modern wealth of high-throughput genomic measurements, it is appealing to approach the construction of gene regulatory networks as a problem of network inference, given measurements sufficiently reflective of gene regulatory activity. See Gardner and Faith [166], for example, for an overview of various versions of this problem and some of the many approaches proposed for its solution.

Genes are sets of segments of DNA that encode information necessary to the proper functioning of a cell. The central dogma of biology says that such information is utilized in the ‘expression’ of genes, whereby biochemical products, in the form of RNA or proteins, are created. The ‘regulation’ of a gene refers to the control of its expression. One important stage at which such control is exerted is at the level of transcription, wherein DNA is copied to RNA. A gene that plays a role in controlling gene expression at this stage is called a ‘transcription factor’ (TF), and the genes that are controlled by it, gene ‘targets.’ The problem of inferring regulatory interactions among genes in this context refers to the identification of TF/target gene pairs.

The relative levels of RNA expression of genes in a cell, under a given set of conditions, can be measured efficiently on a genome-wide scale using microarray technologies. Microarray measurements are therefore a common type of data used in methods of inference of gene regulatory interactions. In particular, for each gene i , the vertex attribute vector $\mathbf{x}_i \in \mathbb{R}^m$ typically consists of RNA relative expression levels measured for that gene over a compendium of m experiments. We provide further details later in the case study of Section 7.3.4. Given such measurements, various techniques are then used in this context to summarize their similarity for each TF/target pair. We will describe a number of such techniques in additional examples below and in the case study. \square

There are of course countless choices for sim in practice. In this section we will concentrate on the use of certain basic linear measures of correlation between univariate random variables $X \in \mathbb{R}$. This choice is both popular in practice and enables us to clearly highlight a number of the fundamental issues that arise in this problem – regardless of how sim is defined. Some comments on alternative choices of similarity may be found in Section 7.5.

7.3.1 Correlation Networks

Let $X \in \mathbb{R}$ be a (continuous) random variable of interest corresponding to the vertices in V . A standard measure of similarity between vertex pairs is $\text{sim}(i, j) = \rho_{ij}$, where

$$\rho_{ij} = \text{corr}_{X_i, X_j} = \frac{\sigma_{ij}}{\sqrt{\sigma_{ii}\sigma_{jj}}} \quad (7.14)$$

is the *Pearson product-moment correlation* between X_i and X_j , expressed in terms of the entries of the covariance matrix $\Sigma = \{\sigma_{ij}\}$ of the random vector $(X_1, \dots, X_{N_v})^T$ of vertex attributes. Given this choice, a natural criterion defining association between i and j is that ρ_{ij} be non-zero. The corresponding association graph G is then just the graph (V, E) with edge set

$$E = \left\{ \{i, j\} \in V^{(2)} : \rho_{ij} \neq 0 \right\} . \quad (7.15)$$

Therefore, given a set of observations of the X_i , the task of inferring this association network graph can be taken as equivalent to that of inferring the set of non-zero correlations.

This task can be approached through testing the hypotheses

$$H_0 : \rho_{ij} = 0 \quad \text{versus} \quad H_1 : \rho_{ij} \neq 0 . \quad (7.16)$$

However, there are at least three important issues to be faced in doing so. First, there is the choice of test statistic to be made. Second, given a test statistic, an appropriate null distribution must be determined for the evaluation of statistical significance.

And third, there is the fact that a large number of tests are to be conducted simultaneously (i.e., for all $N_v(N_v - 1)/2$ potential edges), which implies that the problem of multiple testing must be addressed.

Suppose that for each vertex $i \in V$, we have n independent observations x_{i1}, \dots, x_{in} of X_i . As test statistics, the *empirical correlations*

$$\hat{\rho}_{ij} = \frac{\hat{\sigma}_{ij}}{\sqrt{\hat{\sigma}_{ii}\hat{\sigma}_{jj}}} , \quad (7.17)$$

are a common choice, where the $\hat{\sigma}_{ij}$ are the entries of the unbiased covariance matrix estimate

$$\hat{\Sigma} = \frac{1}{n-1} (\mathbf{X} - \bar{\mathbf{X}})^T (\mathbf{X} - \bar{\mathbf{X}}) , \quad (7.18)$$

with \mathbf{X} here denoting the $n \times N_v$ matrix of observations x_{ik} and $\bar{\mathbf{X}}$, the matrix whose columns contain copies of the sample means \bar{x}_i . If the pair of variables (X_i, X_j) has a bivariate Gaussian distribution, the density of $\hat{\rho}_{ij}$ under $H_0 : \rho_{ij} = 0$ has a concise closed-form expression, but it is somewhat complicated and requires numerical integration or tables to produce p -values. See Hotelling [206]. Therefore, transformed versions of the sample correlation $\hat{\rho}_{ij}$ may be preferable.

For example, under the same conditions as above, the statistic

$$z_{ij} = \frac{\hat{\rho}_{ij}\sqrt{n-2}}{\sqrt{1-\hat{\rho}_{ij}^2}} \quad (7.19)$$

can be shown to have a t -distribution on $n - 2$ degrees of freedom. Furthermore, under the null H_0 , this distribution is relatively robust to departures of the X_i from Gaussianity. Also commonly used is Fisher's transformation,

$$z_{ij} = \tanh^{-1}(\hat{\rho}_{ij}) = \frac{1}{2} \log \left[\frac{(1+\hat{\rho}_{ij})}{(1-\hat{\rho}_{ij})} \right] . \quad (7.20)$$

In this case, again for bivariate Gaussian pairs, the distribution of z_{ij} does not have a simple exact form. But under H_0 this distribution is well approximated by that of a Gaussian random variable with mean zero and variance $1/(n-3)$. The justification for this approximation is asymptotic in n , but has been observed to be quite accurate even for only moderately large n .

Example 7.2 (Correlation Among Gene Expression Levels). Recall our discussion of gene regulatory network inference in Example 7.1. In Section 7.3.4, we will examine the use of microarray measurements for inference of gene regulatory interactions on a large scale, using data for the bacteria *Escherichia coli* (*E. coli*). Here we use a small subset of that data to illustrate our above discussion of correlations.

Figure 7.3 shows a plot of microarray measurements across $n = 445$ experiments for two TFs, *tyrR* and *lrp*, and a potential target gene, *aroG*. It is known that *aroG* is regulated by *tyrR* but not *lrp*. Also shown in each plot is a least-squares regression line and the empirical correlation $\hat{\rho}_{ij}$ in (7.17). The correlation of $\hat{\rho} = 0.85$ between

aroG and *lrp* is decidedly stronger than that between *aroG* and *tyrR*, at $\hat{\rho} = 0.43$. These values lead to Fisher *z*-scores (7.20) of $z = 1.2562$ and 0.4599 , respectively, and comparison to a Gaussian distribution with mean zero and variance $1/(445 - 3) \approx 0.0023$ yields *p*-values of 4.27×10^{-152} and 7.69×10^{-22} , respectively. These results therefore suggest that, based on the use of correlation, the expression levels of the gene *aroG* are strongly associated with those of both *tyrR* and *lrp*. \square

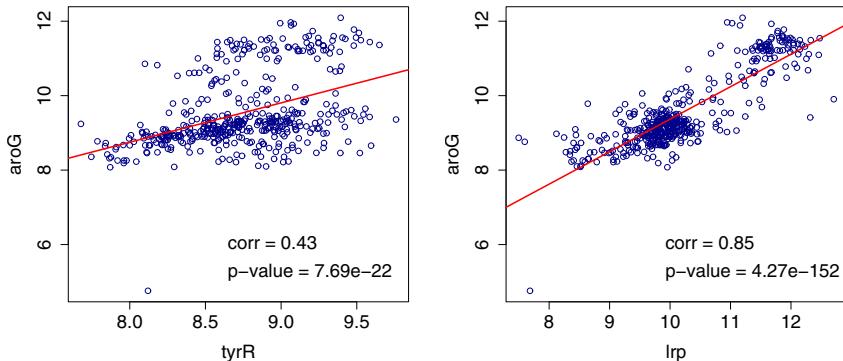


Fig. 7.3 Scatterplots of microarray measurements, in units of log-relative RNA expression levels, for the gene *aroG* and the transcription factors *tyrR* (left) and *lrp* (right), in the organism *E. coli*.

Alternatively, permutation methods may be used to construct an appropriate null distribution. In this case, it can still be advantageous to use a transformed version of the empirical correlations $\hat{\rho}_{ij}$, such as the z_{ij} in (7.19) or (7.20). For each pair of vertices $i, j \in V$, the labels “*i*” and “*j*” are randomly permuted on the $2n$ variables $(x_{i1}, \dots, x_{in}, x_{j1}, \dots, x_{jn})$ some large number B times. The value $z_{ij}^{(b)}$ is then computed on the ‘new’ data each time, for $b = 1, \dots, B$. The significance of the original test statistic z_{ij} may then be assessed through comparison with the distribution of the B values $\{z_{ij}^{(b)}\}$. Note, however, that this approach can become computationally intensive for large N_v .

Given a choice of test statistic and the ability to compute *p*-values, false discovery rate (FDR) principles may be used to address the problem of multiple testing. For example, using the original method proposed by Benjamini and Hochberg [33], we would sort the *p*-values from our $N = N_v(N_v - 1)/2$ tests, yielding a sequence $p_{(1)} \leq p_{(2)} \leq \dots \leq p_{(N)}$, and reject the null hypothesis for all potential edges for which

$$p_{(k)} \leq (k/N) \gamma . \quad (7.21)$$

Ideally, this approach will control the FDR for edges in the network graph at the level γ (i.e., it guarantees that $\text{FDR} \leq \gamma$, where FDR is defined in (2.24)). Alternatively, we could employ the method of Storey [370] and declare edges to be present

using a particular q -value, say q^* . Our expectation is then that only q^*N of those edges will be included erroneously.

In implementing and interpreting such FDR procedures in this manner to construct an association network graph, two points should be kept in mind. First, it is assumed that the p -values serving as input to the procedure are calculated correctly; that is, that they are calculated under an accurate representation of the null distribution. So issues of robustness, like those discussed above, are relevant. Second, for the standard FDR procedures, like those two just mentioned, the stated levels of control are most valid when the tests are independent. Under dependency of tests, the actual control levels can vary from those stated, although how much they vary depends upon both the extent and type of dependency involved.

In our present setting, dependency clearly will be found among the N tests, given the all-pairs nature of the comparisons being made. If the dependency among the test statistics is essentially positive in nature, work of Benjamini and Yekutieli [34] suggests that the control guarantee offered by the standard FDR procedure for independent tests still continues to hold. In addition, these authors offer a simple extension of FDR for the case of more general types of dependency, which in our context entails replacing the value γ in (7.21) with $\gamma/\sum_{k=1}^N(1/k)$. Alternatively, we could use more sophisticated, adaptive FDR procedures, which try to estimate a certain parameter(s) reflective of the dependency and use the estimate to adjust the FDR. See, for example, Efron [130] and other references therein.

7.3.2 Partial Correlation Networks

The oft-cited dictum ‘*correlation does not imply causation*’ should be kept in mind when constructing association networks based on Pearson’s correlation and the like. Two vertices $i, j \in V$ may have highly correlated attributes X_i and X_j because the vertices somehow strongly ‘influence’ each other in a direct fashion. Alternatively, however, their correlation may be high primarily because, for example, they each are strongly influenced by a third vertex, say $k \in V$, and hence X_i and X_j are each highly correlated with X_k . The extent to which this issue is problematic or not will of course depend in no small part on the intended usage of G . But if it is felt desirable to construct a graph G where the inferred edges are more reflective of direct influence among vertices, rather than indirect influence, the notion of *partial correlation* becomes relevant.

In words, the partial correlation of attributes X_i and X_j of vertices $i, j \in V$, defined with respect to the attributes X_{k_1}, \dots, X_{k_m} of vertices $k_1, \dots, k_m \in V \setminus \{i, j\}$, is the correlation between X_i and X_j left over after adjusting for those effects of X_{k_1}, \dots, X_{k_m} common to both. Formally, letting $S_m = \{k_1, \dots, k_m\}$, we define the partial correlation of X_i and X_j , adjusting for $\mathbf{X}_{S_m} = (X_{k_1}, \dots, X_{k_m})^T$, as

$$\rho_{ij|S_m} = \frac{\sigma_{ij|S_m}}{\sqrt{\sigma_{ii|S_m} \sigma_{jj|S_m}}} . \quad (7.22)$$

Here $\sigma_{ii|S_m}$, $\sigma_{jj|S_m}$, and $\sigma_{ij|S_m} = \sigma_{ji|S_m}$ are the diagonal and off-diagonal elements, respectively, of the 2×2 partial covariance matrix

$$\Sigma_{11|2} = \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21} , \quad (7.23)$$

where Σ_{11} , Σ_{22} , and $\Sigma_{12} = \Sigma_{21}^T$ are defined through the partitioned covariance matrix

$$\text{Cov} \begin{pmatrix} \mathbf{W}_1 \\ \mathbf{W}_2 \end{pmatrix} = \begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{bmatrix} , \quad (7.24)$$

for $\mathbf{W}_1 = (X_i, X_j)^T$ and $\mathbf{W}_2 = \mathbf{X}_{S_m}$. Note that in the case of $m = 0$, the partial correlation in (7.22) reduces to the Pearson correlation in (7.14). If $(X_i, X_j, X_{k_1}, \dots, X_{k_m})^T$ has a multivariate Gaussian distribution, then $\rho_{ij|S_m} = 0$ if and only if X_i and X_j are independent conditional on \mathbf{X}_{S_m} . For more general distributions, however, zero partial correlation will not necessarily imply independence (the converse, of course, is still true).

Partial correlations can be used in various ways for defining an association network graph G , with respect to vertex attributes X_1, \dots, X_{N_v} . For example, for a given choice of m , we may dictate that an edge be present only when there is correlation between X_i and X_j regardless of which m other vertices are conditioned upon. That is,

$$E = \left\{ \{i, j\} \in V^{(2)} : \rho_{ij|S_m} \neq 0, \text{ for all } S_m \in V_{\setminus\{i,j\}}^{(m)} \right\} , \quad (7.25)$$

where $V_{\setminus\{i,j\}}^{(m)}$ is the collection of all unordered subsets of m (distinct) vertices from $V \setminus \{i, j\}$. Other choices are clearly possible as well.

Under the definition of edges in (7.25), the problem of determining the presence or absence of a potential edge $\{i, j\}$ in G can be represented as one of testing

$$H_0 : \rho_{ij|S_m} = 0 \quad \text{for some } S_m \in V_{\setminus\{i,j\}}^{(m)}$$

versus

$$H_1 : \rho_{ij|S_m} \neq 0 \quad \text{for all } S_m \in V_{\setminus\{i,j\}}^{(m)} . \quad (7.26)$$

In order to infer an association network graph in this context, given measurements x_{i1}, \dots, x_{in} for each vertex $i \in V$, we must again select a test statistic, construct an appropriate null distribution, and adjust for multiple testing, as above.

With respect to the issue of a test statistic, it is useful to first note that the analogue of the empirical correlation coefficient $\hat{\rho}_{ij}$ in (7.17) is the empirical partial correlation coefficient, $\hat{\rho}_{ij|S_m}$. An algorithmic definition of this value is that it is the result of (i) performing separate multiple linear regressions of the observations of X_i and X_j , respectively, on the observed values of \mathbf{X}_{S_m} , and then (ii) computing the empirical correlation (7.17) between the two resulting sets of residuals. In practice, however, when computing many such coefficients, it is more computationally efficient to use recursive expressions, which relate each m -th order empirical partial correlation coefficient to three $(m-1)$ -th order coefficients. See, for example, Anderson [11, Ch. 2.5.3]. Exploiting redundancies in these recursions, all m -th or-

der coefficients for a given pair $i, j \in V$ may be computed in $O(m^3)$ time from the $(m - 1)$ -th order coefficients.

In approaching the testing problem in (7.26), it can be convenient to consider it as a collection of smaller testing sub-problems of the form

$$H'_0 : \rho_{ij|S_m} = 0 \quad \text{versus} \quad H'_1 : \rho_{ij|S_m} \neq 0 , \quad (7.27)$$

for which $\hat{\rho}_{ij|S_m}$ is a natural test statistic. Under the assumption that the joint distribution of $(X_i, X_j, X_{k_1}, \dots, X_{k_m})^T$ is a multivariate Gaussian distribution, the empirical partial correlation coefficient $\hat{\rho}_{ij|S_m}$ has a null distribution that is known but, again, whose functional form is somewhat complicated, requiring numerical integration or tables for practical usage (Hotelling [206]). Fisher's transformation

$$z_{ij|S_m} = \frac{1}{2} \log \left[\frac{(1 + \hat{\rho}_{ij|S_m})}{(1 - \hat{\rho}_{ij|S_m})} \right] \quad (7.28)$$

is therefore often used instead as a test statistic. It has an asymptotically Gaussian null distribution, with mean 0 and variance $1/(n - m - 3)$.

Example 7.3 (Partial Correlation Among Gene Expression Levels). Recall the microarray data shown in Figure 7.3 of Example 7.2. It is known that a TF can actually be a target of another TF. And so direct correlation between measurements of a TF and a gene target may actually just be a reflection of the regulation of that TF by another TF. Partial correlation provides a tool for attempting to sort out such confounding among variables.

In Figure 7.4 are shown scatterplots of the data for the gene target *aroG* against that of the TFs *tyrR* and *lrp*, each adjusted for the effects of the other. More precisely, the measurements of *aroG* were regressed upon those of *lrp*, and similarly those of *tyrR* were regressed upon those of *lrp*, and the residuals from the two regressions were plotted. The analogous analysis was done with the roles of *tyrR* and *lrp* reversed. We can see that the strength of the association between *aroG* and *lrp* is little changed, as measured now by the empirical partial correlation coefficients, while that between *aroG* and *tyrR* has dropped by nearly a third. Note that the association between *aroG* and *tyrR* would now be judged not to be at all significant at any reasonable level, based on the *p*-value 0.92, which derives from comparison of the Fisher *z*-value in (7.28) to a Gaussian distribution with mean zero and variance $1/(445 - 1 - 3) \approx 0.0023$.

Also shown in Figure 7.4 are similar scatterplots based on adjusting for the effects of $m = 152$ TFs rather than just one. So, for example, in comparing *aroG* and *tyrR* we adjust for the effects of not only *lrp* but also 151 other TFs in *E. coli*. Similarly, in comparing *aroG* and *lrp* we adjust for *tyrR* and the same 151 other TFs. In both cases, we adjust by regressing the expression levels of *aroG* and the TF of interest on those of the other 152 TFs. From the plots we can see that there is moderately strong evidence of association left in the residuals, as indicated by the *p*-values corresponding to the empirical partial correlation coefficients. Again the *z*-value in (7.28) was used, but now with a variance of $1/(445 - 152 - 3) \approx 0.00345$

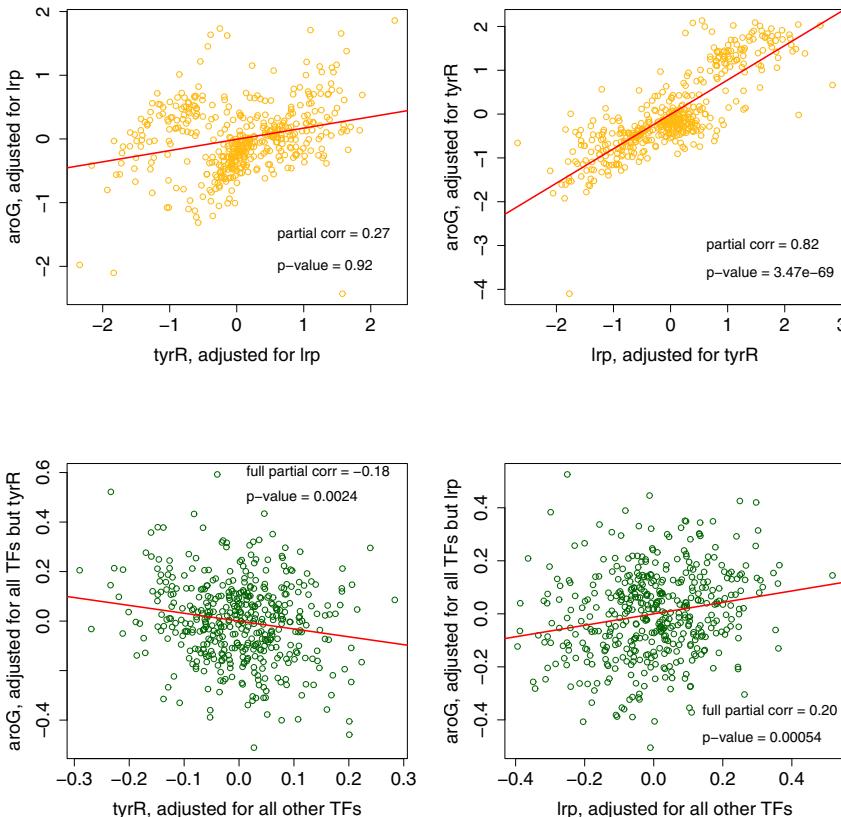


Fig. 7.4 Scatterplots of microarray measurements, in units of log-relative RNA expression levels, for the gene *aroG* and the transcription factors *tyrR* (left column) and *lrp* (right column). Top row (yellow) shows the data adjusted for *lrp* (left) and *tyrR* (right); bottom row (green) shows the data adjusted for all other transcription factors.

for the corresponding Gaussian under the null hypothesis. These results suggest that, beyond the effects of the other TFs, in each case the expression levels of *aroG* have a non-trivial association with those of the remaining TF (i.e., *tyrR* or *lrp*, respectively). Interestingly, the sign of the association between *aroG* and *tyrR* has in principle changed, from positive to negative, although recall that in the former case the *p*-value did not suggest a partial correlation significantly different from zero. This negative association suggests a repressive role of *tyrR* in regulating *aroG*. \square

A test of (7.26) may be constructed from the tests of the sub-problems (7.27) through aggregation. For example, Wille et al. [406] suggest combining the information in *p*-values $p_{ij|S_m}$, over all S_m by defining

$$p_{ij,max} = \max \left\{ p_{ij|S_m} : S_m \in V_{\setminus \{i,j\}}^{(m)} \right\} \quad (7.29)$$

to be the p -value for the test in (7.26), where the $p_{ij|S_m}$ are the p -values for the tests of (7.27). Given the full collection of such p -values $\{p_{ij,max}\}$, over all potential edges $\{i,j\}$, an FDR procedure may be applied to this collection to choose an appropriate testing threshold, analogous to the manner described above. See Wille and Bühlmann [405] for justification.

Note that some caution is called for in choosing the order(s) m defining the underlying association network graph. For example, if $n \leq m$ the empirical partial correlation coefficients $\hat{\rho}_{ij|S_m}$ are not well defined. And even when $n > m$, these values are likely to be poor estimates of $\rho_{ij|S_m}$ if m is still large with respect to n . In addition, with increasing m the computational costs grow exponentially fast. Citing a worst case total cost of $O(n^{m+2})$ if coefficients are to be computed for all vertex pairs, de la Fuente, Bing, Hoeschele, and Mendes [110] advocate using only partial correlation coefficients up to order $m = 2$ in the context of inference of biochemical networks.

7.3.3 Gaussian Graphical Model Networks

A special – and, indeed, popular – case of the use of partial correlation coefficients is when $m = N_v - 2$ and the attributes are assumed to have a multivariate Gaussian joint distribution. Here the partial correlation between attributes of two vertices is defined conditional upon the attribute information at *all* other vertices. Denoting these coefficients as $\rho_{ij|V \setminus \{i,j\}}$, under the Gaussian assumption the vertices $i, j \in V$ have partial correlation $\rho_{ij|V \setminus \{i,j\}} = 0$ if and only if X_i and X_j are conditionally independent given all of the other attributes. The graph $G = (V, E)$ with edge set

$$E = \left\{ \{i, j\} \in V^{(2)} : \rho_{ij|V \setminus \{i,j\}} \neq 0 \right\} \quad (7.30)$$

is called a *conditional independence graph*.⁶ The overall model, combining the multivariate Gaussian distribution with the graph G , is called a *Gaussian graphical model*. This model is one example of the more general class of graphical models, a point we will revisit briefly in Chapter 10.

A useful result in the context of Gaussian graphical models is that the partial correlation coefficients may be expressed in the form

$$\rho_{ij|V \setminus \{i,j\}} = \frac{-\omega_{ij}}{\sqrt{\omega_{ii}\omega_{jj}}} , \quad (7.31)$$

⁶ Note that it is actually the non-edges in G that correspond to conditional independence, and the edges, to conditional dependence.

where ω_{ij} is the (i, j) -th entry of $\Omega = \Sigma^{-1}$, the inverse of the covariance matrix Σ of the vector $(X_1, \dots, X_{N_v})^T$ of vertex attributes. See Lauritzen [248, Ch. 5.1] or Whittaker [403, Ch. 5.8], for example. The matrix Ω is known as the *concentration* or *precision* matrix, and its non-zero off-diagonal entries, occurring as they do in the numerator of (7.31), are linked in one-to-one correspondence with the edges in G . As a result, G also is sometimes referred to as a *concentration graph*.

The problem of inferring G from data is known in this context as the ‘covariance selection problem’ (Dempster [115]). A standard approach to this problem employs a recursive, likelihood-based procedure that effectively tests the hypotheses

$$H_0 : \rho_{ij|V \setminus \{i, j\}} = 0 \quad \text{versus} \quad H_1 : \rho_{ij|V \setminus \{i, j\}} \neq 0 , \quad (7.32)$$

using the empirical partial correlations $\hat{\rho}_{ij|V \setminus \{i, j\}}$, defined as in (7.31), but with the entries of $\hat{\Omega} = \hat{\Sigma}^{-1}$ in place of those of Ω , where $\hat{\Sigma}$ is as in (7.18). Starting with the complete graph on N_v vertices as an initial estimate $\hat{G}^{(0)}$, and a pre-specified significance level, say α , edges corresponding to those vertex pairs $\{i, j\}$ for which the null hypothesis in (7.32) is not rejected at level α are removed. The resulting graph, say $\hat{G}^{(1)}$, now summarizes a set of conditional (in)dependencies. Taking these now as known constraints, the procedure is repeated, yielding an estimate $\hat{G}^{(2)}$, and so forth, until no further edges are removed. See Whittaker [403, Ch. 6] or Lauritzen [248, Ch. 5], for details.

For large-scale network graphs, this approach presents a number of problems. First, it is computationally intensive and really only feasible for graphs of quite modest size. Second, there is no attention paid to multiple testing. And third, it is frequently the case in large-scale settings that $n \ll N_v$, and so the estimated covariance matrix $\hat{\Sigma}$ will not be of full rank, and hence will not have a proper inverse. A number of methods have been proposed for inference in this setting, which we review here, organized according to those based on testing and those based on regression.

7.3.3.1 Methods Based on Testing

Drton and Perlman [127] suggest a method of graph inference that addresses the problem of multiple testing in the case where $n > N_v$. Working with the Fisher transformation $z_{ij|V \setminus \{i, j\}}$, defined in analogy to (7.28), they propose a collection of tests jointly at the α significance level, that assign an edge to each pair $\{i, j\}$ if and only if

$$|z_{ij|V \setminus \{i, j\}}| > n_{N_v}^{-1/2} c_{N_v}(\alpha) , \quad (7.33)$$

where $n_{N_v} = n - N_v$ and

$$c_{N_v}(\alpha) = \Phi^{-1} \left[0.5(1 - \alpha)^{[2/N_v(N_v - 1)]} + 0.5 \right] , \quad (7.34)$$

for $\Phi(\cdot)$ the cumulative distribution function of the Gaussian distribution with mean 0 and variance 1. Under the assumption that G is the true conditional independence graph underlying the data, it can be shown that G will be correctly inferred by this procedure with probability at least $1 - \alpha$, for large n .

Note that this procedure is not applicable to the case $n \ll N_v$. When the sample size is less than the number of vertices, the estimate $\hat{\Sigma}$ in (7.18) will still be unbiased for Σ but will suffer from undue variance, which in turn will be detrimental to any inference on the partial correlation coefficients (7.31) that use (pseudo)inversion of $\hat{\Sigma}$ to estimate Ω . Schäfer and Strimmer [340, 341] have suggested a number of methods for network inference in this context, involving various techniques for producing reduced-variance estimates of Σ .

For example, in Schäfer and Strimmer [340] the use of bootstrap aggregation – or ‘bagging’ – is proposed for variance reduction. Specifically, simple random sampling of the N_v -dimensional sample vectors \mathbf{x} is used to create a collection of some large number B ‘new’ datasets, and for each new dataset $b = 1, \dots, B$, an estimate $\hat{\Sigma}^{(b)}$ is computed. A smoothed covariance estimate $\hat{\Sigma}^{bag}$ can then be obtained through the average of the $\hat{\Sigma}^{(b)}$, and an improved estimate of Ω results from computing the Moore-Penrose inverse of $\hat{\Sigma}^{bag}$. Substitution in (7.31) then yields partial correlation estimates, say $\hat{\rho}_{ij|V \setminus \{i,j\}}^{bag}$. See Breiman [62] or Hastie, Tibshirani, and Friedman [194, Ch. 8.7] for background and motivation on the principle of bagging.

For testing the hypotheses in (7.32), in order to construct an inferred conditional independence graph, an appropriate null distribution for the estimates $\hat{\rho}_{ij|V \setminus \{i,j\}}^{bag}$ must be specified. Since an analytic approach to this task is daunting, Schäfer and Strimmer instead choose to model this distribution, proposing a mixture model of the form

$$f(\rho) = \pi f_0(\rho; v) + (1 - \pi) f_1(\rho) , \quad (7.35)$$

where f_0 has the same functional form as the distribution of the estimates $\hat{\rho}_{ij}$ and $\hat{\rho}_{ij|k_1, \dots, k_m}$, given in Hotelling [206], and f_1 is the uniform distribution on $[-1, 1]$. The parameter v in f_0 is a degree-of-freedom parameter that is estimated empirically. The mixture weight π also can be estimated empirically. This mixture model itself is similar to models underlying the pFDR methodology of Storey [370], and also Efron’s [130] extension of FDR, and therefore multiple testing procedures can be integrated easily into the analysis. See Schäfer and Strimmer [340, 341] for details.

7.3.3.2 Methods Based on Penalized Linear Regression

It has also been proposed to use penalized regression methods to infer G , exploiting a variation on the well-known connection between linear correlation and linear regression. Suppose that the random vector $(X_1, \dots, X_{N_v})^T$ of vertex attributes has a multivariate Gaussian distribution, with covariance Σ , as assumed above, and also zero mean. Then a standard result yields that, for the attribute X_i of a fixed vertex i , given the values of the attributes at the remaining vertices, say $\mathbf{X}^{(-i)} = (X_1, \dots, X_{i-1}, X_{i+1}, \dots, X_{N_v})^T$, its conditional expectation has the form

$$\mathbb{E}[X_i | \mathbf{X}^{(-i)} = \mathbf{x}^{(-i)}] = (\boldsymbol{\beta}^{(-i)})^T \mathbf{x}^{(-i)}, \quad (7.36)$$

where $\boldsymbol{\beta}^{(-i)} \in \mathbb{R}^{N_v-1}$. See, for example, Johnson and Wichern [214, Ch. 4]. Furthermore, importantly, the entries of $\boldsymbol{\beta}^{(-i)}$ can be expressed in terms of the entries of the precision matrix $\boldsymbol{\Omega}$, as $\beta_j^{(-i)} = -\omega_{ij}/\omega_{ii}$. See Lauritzen [248, App. C]. Hence, a potential edge $\{i, j\}$ is in the edge set E defined by (7.30) if and only if $\beta_j^{(-i)}$ (and, therefore, also $\beta_i^{(-j)}$) is not equal to zero.

These observations suggest that inference of G be pursued via inference of the non-zero elements of $\boldsymbol{\beta}^{(-i)}$ in (7.36), using regression-based methods of estimation and variable selection. In fact, it can be shown that the vector $\boldsymbol{\beta}^{(-i)}$ is the solution to the optimization problem

$$\arg \min_{\tilde{\boldsymbol{\beta}}: \tilde{\beta}_i=0} \mathbb{E} \left[\left(X_i - (\tilde{\boldsymbol{\beta}}^{(-i)})^T \mathbf{X}^{(-i)} \right)^2 \right]. \quad (7.37)$$

It is natural, therefore, to replace this problem by a corresponding least-squares optimization. However, because there will be $N_v - 1$ variables in the regression for each X_i , and in addition it may be the case that $n \ll N_v$, a penalized regression strategy is prudent.

Meinshausen and Bühlmann [275], for example, have suggested using estimates of the form

$$\hat{\boldsymbol{\beta}}^{(-i)} = \arg \min_{\boldsymbol{\beta}: \beta_i=0} \sum_{k=1}^n \left(x_{ik} - (\boldsymbol{\beta}^{(-i)})^T \mathbf{x}_k^{(-i)} \right)^2 + \lambda \sum_{j \neq i} |\beta_j^{(-i)}|. \quad (7.38)$$

This strategy is based on the use of the Lasso method of Tibshirani [380], which not only estimates the coefficients in $\boldsymbol{\beta}^{(-i)}$ but also forces values $\hat{\beta}_j^{(-i)} = 0$ where the association between X_i and X_j is felt to be too weak, with respect to the choice of penalty parameter λ . In other words, the Lasso methodology performs simultaneous estimation and variable selection, a characteristic due to the particular form of the penalty. The estimates in (7.38) can be computed, using the so-called LARS procedure of Efron, Hastie, Johnstone, and Tibshirani [132], in $O(nN_v \min\{n, N_v\})$ steps. Therefore, to compute estimates for all vertices $i \in V$, and presuming that $n \ll N_v$, the computational cost will be slightly worse than $O(N_v^3)$.

Given the estimate $\hat{\boldsymbol{\beta}}^{(-i)}$ for all $i \in V$, an inferred network graph can be constructed by assigning edges according to whether the corresponding estimated coefficients are zero or non-zero. Note that it is not true that $\hat{\beta}_j^{(-i)} \neq 0$ implies $\hat{\beta}_i^{(-j)} \neq 0$, and vice versa. One can therefore specify that a potential edge $\{i, j\}$ be assigned if either $\hat{\beta}_j^{(-i)}$ or $\hat{\beta}_i^{(-j)}$ are non-zero, or require that both be non-zero. Meinshausen and Bühlmann show that, under conditions on Σ, λ, N_v , and n , the true graph G will be inferred with high probability using either of these conventions, even in cases where $n \ll N_v$. Note that the choice of λ is important. Meinshausen and Bühlmann show that selecting λ by cross-validation will yield provably bad results, because

the goal is one of variable selection and not prediction. These authors suggest an empirically driven choice of λ , for which a small simulation study suggests some promise.

The use of penalized regression methods for inference in Gaussian graphical models has generated a substantial amount of interest in the research community. Other contributions in this area include Dobra et al. [119], who describe a Bayesian approach to the problem, Wainwright, Ravikumar, and Lafferty [390], who propose a Lasso-like penalized logistic regression approach, and Yuan and Lin [410], who present a method of penalized maximum likelihood. Friedman, Hastie, and Tibshirani [161] have developed a computationally fast procedure – the so-called graphical lasso – for solving this penalized maximum likelihood problem.

7.3.4 Case Study: Inferring Genetic Regulatory Interactions

Recall our discussion of gene regulatory interaction and microarray measurements in Examples 7.1, 7.2, and 7.3. Here we use a complete set of microarray data and correlation-based methods like those described above to do large-scale inference of regulatory interactions between TF/target gene pairs in the bacteria *Escherichia coli* (*E. coli*). The dataset, as well as the basic structure of our analysis, are taken from Faith et al. [140], although the choice of methodologies used here differs.

The data consist of log-expression levels of RNA, relative to a background, measured for genes in *E. coli* when the cells were subjected to various conditions, such as changes in pH, growing media, heat, oxygen concentrations, and genetic composition. In full, a total of 445 vectors of measurements – called ‘expression profiles’ – were collected across 4,345 genes. Among these genes, 153 are known TFs. Also available is information on currently known TF/target pairs in *E. coli*, for these 153 TFs and 1,058 other of the 4,345 genes, extracted from the database RegulonDB [338]. While the regulatory information in this database is not complete, it is largely so, and the information therein is held to be quite accurate, although not perfect. Thus we have available – somewhat uniquely for this problem – a high-quality truth-set by which to assess the usefulness of the regulatory relationships predicted by the methods we consider.

Figure 7.5 shows an image representation of the (log-relative) RNA expression levels across experiments for the 153 genes listed as known TFs in RegulonDB. Due to the nature of the regulation process, the expression levels of TFs and their targets often can be expected to be highly correlated with each other, as we saw earlier. And given that many TFs have multiple targets, including other TFs, the expression levels among many of the TFs can be expected to be highly correlated as well, as we also saw earlier. Such correlations are evident in the image plot, where the order of the genes (rows) and experiments (columns) has been chosen using a cluster analysis, to enhance the visual assessment of correlation.

In predicting regulatory interactions, we focused solely on interactions between TF/target gene pairs. We implemented three different correlation-based methods for

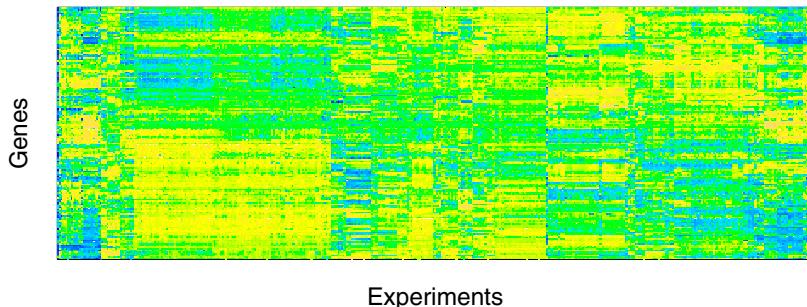


Fig. 7.5 Image representation of 445 microarray expression profiles collected for *E. coli*, under various conditions, for the 153 genes that are listed as known transcription factors in the RegulonDB database. Larger negative values are indicated with darker shades of blue, and larger positive values, in yellow to orange. Shades of green indicate values comparatively close to zero.

quantifying and detecting such interactions, wherein a regulatory interaction was declared for a TF/target pair if a p -value for the corresponding correlation-based statistic fell beneath a given threshold. The first method is based on the standard Pearson correlation, in (7.14), between the expression levels of a TF and a potential target gene. The second method is based on the partial correlation between the same pairs, controlling for possible shared effects of a single additional TF, across all 152 other TFs. More precisely, this method computes partial correlations like that in (7.22), with $m = 1$, with the expression levels of each TF individually playing the role of the conditioning random variable. It then declares a TF/target interaction based on the p -value in (7.29) for the testing problem in (7.26). Finally, the third method also uses partial correlation between TFs and potential targets, but simultaneously adjusts for the expression levels of all remaining $m = 152$ TFs (or $m = 151$, if the potential target also is a TF). We refer to this latter partial correlation as the ‘full partial correlation’ in what follows.

For each choice of correlation, the appropriate Fisher transformation was applied and p -values were computed using the corresponding asymptotic Gaussian distribution, based on a sample size $n = 445$. A threshold was then varied from 0 to 1. For p -values falling beneath the threshold, the corresponding TF/target pair were said to interact. An association network graph was created by assigning a vertex to each gene and an edge to each TF/target gene pair declared to interact.

We evaluated the performance of each of these three methods of network inference in two ways. First, each inferred network graph was compared to the graph induced by the information in the RegulonDB database, for just the 153 TFs and $153 + 1,058 = 1,211$ genes in that database. Figure 7.6 shows the resulting set of ROC curves. Overall, the partial and full partial correlation methods perform about

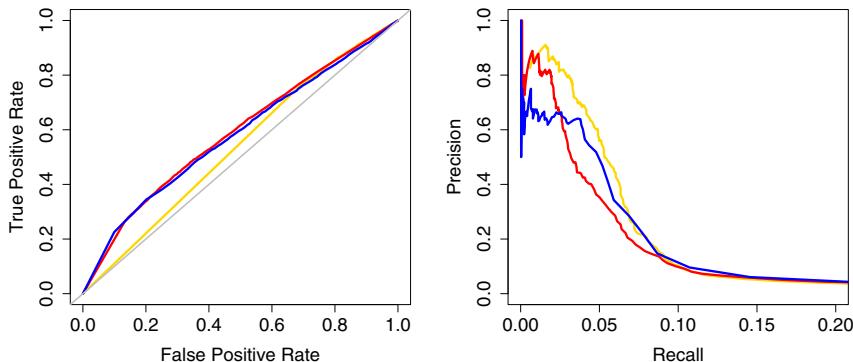


Fig. 7.6 ROC curves (left) and Precision/Recall curves (right) evaluating performance of the correlation (yellow), partial correlation (red), and full partial correlation (blue) methods of network inference described in the text.

equally well, and better than the correlation method. However, none of the methods perform in a particularly stellar fashion, which would seem to suggest that correlation between TF and target expression profiles – whether unadjusted or adjusted for the effects of other TFs – is not an especially strong global indicator for gene regulation in this data.

However, from the standpoint of biology, interest typically centers much more on the prediction of *new* interactions, rather than the recovery of old interactions. Accordingly, we also show in Figure 7.6 a plot of precision (i.e., the fraction of predicted links that are true) versus recall (i.e., the fraction of true links that are correctly predicted) for each of the three methods.⁷ All three methods are seen to share a small common region of quite high precision, with the correlation method arguably holding a slight advantage over the two partial correlation methods. On the other hand, in the same region, all three methods show abysmally low recall. Faith et al. [140] have argued that the low recall is due primarily to limitations in both the number and the diversity of the expression profiles produced by the available experiments.

Nevertheless, if we concentrate on the region of high precision, we find that we are able to accurately predict a number of TF/target interactions that were unknown (at least prior to the analysis of Faith et al. [140] with this same data). For example, Figure 7.7 shows the targets predicted by the correlation method, using a threshold

⁷ Note that the quantities on the axes of the two plots in Figure 7.6 are related, but not identical. Let TP and FP denote the number of true and false positive predictions, and TN and FN , the number of true and false negative predictions. Then the true positive rate is given by $TPR = TP/(TP + FN)$, and the false positive rate, by $FPR = FP/(FP + TN)$. The true positive rate (FPR) is the same as what is meant by ‘recall.’ On the other hand, the ‘precision’ – what is also sometimes called ‘positive predictive value’ – is given by $PPV = TP/(TP + FP)$.

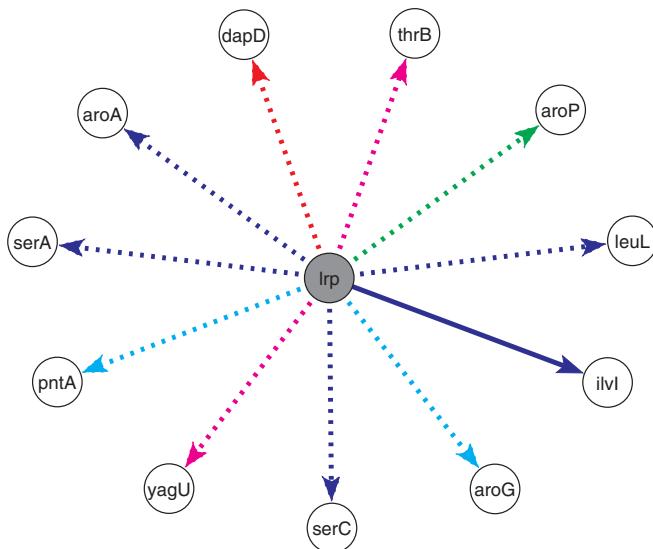


Fig. 7.7 Gene targets predicted by the correlation method for the transcription factor *lrp*, at 60% precision. Patterns and colors of arcs are described in the text.

corresponding to 60% precision⁸ in the plot of Figure 7.6, for the transcription factor *lrp*, which is a known regulator of various metabolic processes in *E. coli*. A total of 11 gene targets of *lrp* were predicted. Of these, five (in blue) are indicated to be true positives according to RegulonDB, and one (in green) is indicated to be a false positive. In Faith et al. [140], experimentation was done to directly verify many of the predictions of those authors, including all but one (shown with a solid edge) of the 11 here. All ten of the predictions tested (shown with dotted edges) were found to be true predictions, including the apparent false positive indicated by RegulonDB. Five (in magenta, cyan, and red) of the ten predicted gene targets tested were new, with four of them (in magenta and cyan) present in RegulonDB – but not as targets of *lrp*, and one of them (in red) not present. Of the four present in RegulonDB, two of them (in magenta) were genes previously known to be regulated by at least one other TF, while the other two (in cyan) had no previously known regulator.

7.4 Tomographic Network Topology Inference

In the field of imaging, *tomography* refers to the process of imaging by sections. A number of familiar examples of tomographic imaging are found in the field of medical imaging, where a combination of physical devices (e.g., X-ray or MRI scanners)

⁸ A *p*-value threshold of 10^{-125} was needed to obtain this precision, rendering an FDR-based correction for multiple testing relatively pointless.

and computers are used to create images of internal aspects of the human body (e.g., cross-sections of the head or torso) through reconstruction algorithms that exploit mathematical expressions relating the externally-acquired data to the unknown internal structure. *Tomographic network topology inference* is named in analogy to tomographic imaging and refers to the inference of ‘interior’ components of a network – both vertices and edges – from data obtained at some subset of ‘exterior’ vertices. Here ‘exterior’ and ‘interior’ are somewhat relative terms and generally are used simply to distinguish between vertices where it is and is not possible (or, at least, not convenient) to obtain measurements. For example, in computer networks, desktop and laptop computers are typical instances of ‘exterior’ vertices, while Internet routers to which we do not have access are effectively ‘interior’ vertices.

The problem of tomographic network topology inference is just one instance of what are more broadly referred to as problems of *network tomography*. The term was coined by Vardi [386] and originally was used to describe problems in the context of computer network monitoring, in which aspects of the ‘internal’ workings of the network, such as the intensity of traffic flowing between vertices, are inferred from ‘external’ measurements. The problem of traffic matrix estimation, described briefly in Example 2.3, is another such problem, which we will examine in detail later in Section 9.3. Surveys on network tomography may be found in Coates, Hero, Nowak, and Yu [96] and Castro et al. [78].

With information only obtainable through measurements at ‘exterior’ vertices, the tomographic network topology inference problem can be expected to be quite difficult in general. This difficulty is primarily due to the fact that, for a given set of measurements, there are likely many network topologies that conceivably could have generated them, and without any further constraints on aspects like the number of internal vertices and edges, and the manner in which they connect to one another, we have no sensible way of choosing among these possible solutions.

Our problem is thus an example of what is known as an ‘ill-posed inverse problem’ in mathematics – an inverse problem, in the sense of inverting the mapping from internal to external, and ill-posed, in the sense of that mapping being many-to-one. In order to obtain useful solutions to such problems, typically additional information is incorporated into the problem statement by way of model assumptions on the form of the internal structure to be inferred. For the tomographic inference of network topologies, a key structural simplification has been the restriction to inference of networks in the form of trees. In fact, nearly all work in this area to date has focused on this particular version of the problem, with exceptions seeming to mainly involve techniques for obtaining more general graphs by merging multiple trees (e.g., Huson and Bryant [210], Coates, Rabbat, and Nowak [97]).

In this section, we will therefore focus on the tomographic inference of tree topologies. Taken as a whole, the amount of work on this topic would appear to easily dwarf that on the topics in Sections 7.2 and 7.3 combined. There are two main areas in which such work may be found: in biology and the inference of phylogenies, and in computer network analysis and the identification of logical topologies (i.e., as opposed to physical topologies). The first area is decidedly older as a field, with roots going back as far as the late 1950’s, while the second area goes back no more

than a decade ago, although it has been quite active during that time. Phylogenetic inference is concerned with the construction of trees (i.e., phylogenies) from data, for the purpose of describing evolutionary relationships among biological species. For a comprehensive overview of this area, see the book by Felsenstein [141]; also useful, and shorter, are the two surveys by Holmes [204, 205]. In computer network topology identification, the goal is to infer the tree formed by a set of paths along which traffic flows from a given origin Internet address to a set of destination addresses. Castro et al. [78, Sec. 4] provide a survey of this area.

Despite the fact that there are very different systems, tools, and sciences underlying these two areas, there are actually many common aspects to the structure of the underlying tomographic inference problems, and similarly, to the statistical solutions proposed – although, interestingly, such commonalities do not seem to have been remarked upon widely to date. We will try to emphasize this commonality, wherever possible, in our statement of the problem and exposition of proposed solutions. However, our illustrations and the case study in Section 7.4.5 will emphasize computer network applications, which are somewhat simpler and more self-contained than phylogenetic inference problems, and which also have been comparatively free of the sometimes substantial contention that has traditionally accompanied phylogenetic inference in biology.

7.4.1 Tomographic Inference of Tree Topologies

An (undirected) *tree* $T = (V_T, E_T)$ is a connected graph with no cycles. A *rooted tree* is a tree in which a vertex $r \in V_T$ has been singled out. The subset of vertices $R \subset V_T$ of degree one are called the *leaves*; we will refer to the vertices in $V \setminus \{\{r\} \cup R\}$ as the *internal vertices*. The edges in E_T are often referred to as *branches*. A *binary tree* – also called a *bifurcating tree* – is a rooted tree in which, in moving from the root towards the leaves, each internal vertex has at most two children. We will restrict our attention here almost entirely to binary trees. Trees with more general branching structure can always be represented as binary trees and, indeed, methods for their inference can be built upon inferential methods for binary trees (e.g., Duffield, Horowitz, Lo Presti, and Towsley [128]).

Figure 7.8 shows a schematic representation of a binary tree $T = (V_T, E_T)$, annotated to illustrate our tomographic network inference problem. The five leaf vertices $R = \{1, 2, 3, 4, 5\}$, in yellow, are known to us, but the internal vertices $\{i_1, i_2, i_3\}$, in green, are not. Nor are the branches known to us. The root vertex r , in blue, may or may not be known to us, but its existence is assumed in imposing a rooted tree structure upon the unknown network graph we seek to infer. In addition, we associate with each branch a weight w , which we may or may not wish to infer.

We define our tomographic network inference problem as follows. Suppose that for a set of N_l vertices – possibly a subset of the vertices V of an arbitrary graph $G = (V, E)$ – we have n independent and identically distributed observations of some random variables $\{X_1, \dots, X_{N_l}\}$. Under the assumption that these vertices can be

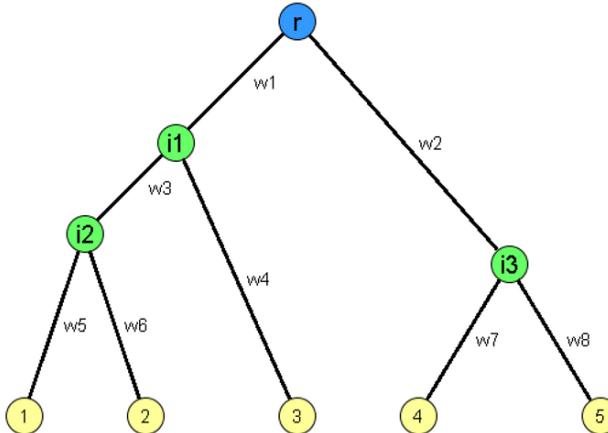


Fig. 7.8 Schematic representation of a binary tree in association with the tomographic network inference problem. Measurements are available at the leaves 1, 2, 3, 4, and 5, in yellow. The internal vertices, i_1, i_2 , and i_3 , in green, and possibly the root r , in blue, are unknown, as are the branches joining the various vertices.

meaningfully identified with the leaves R of a tree T , we aim to find that tree \widehat{T} in the set \mathcal{T}_{N_l} of all binary trees with N_l labeled leaves that best explains the data, in some well-defined sense. If we have knowledge of a root r , then the roots of the trees in \mathcal{T}_{N_l} will all be identified with r . In some contexts we may also be interested in inferring a set of weights for the branches in \widehat{T} .

Example 7.4 (Binary Measurements: Multicast Probes). As an illustration, consider the case where the random variables X_i are binary and each of the n measurements therefore consists of an N_l -tuple of zeros and ones. An example of such measurements is found in the context of computer networks, in the use of so-called multicast probes. Such a probe consists of a small packet of information, sent simultaneously from a single origin Internet address to multiple destination addresses. On the way towards its destinations, at the first intermediate routing device encountered for which destinations lie along multiple paths, the packet is effectively copied, and a copy is sent along each path. This action is continued as the routes toward the leaves of the corresponding multicast tree continue to branch outwards. In Figure 7.8, for example, copies would be made at each of the interior vertices i_1, i_2 , and i_3 .

It is possible, however, that along the way some of these packets will be ‘lost’ – for instance, they can be dropped as part of the manner in which Internet systems keep traffic levels and queuing delays in check. If a packet is lost at a given intermediate router, none of the destinations along paths continuing from that router will receive the packet. For example, in the tree shown in Figure 7.8, a packet sent from the vertex r that was forwarded by vertex i_1 but was lost by vertex i_2 would

not be received by vertices 1 and 2, but could⁹ be received by vertex 3. An N_l -tuple $\{X_1, \dots, X_{N_l}\}$ of binary measurements is used to indicate whether each of N_l destination addresses did (i.e., 1) or did not (i.e., 0) receive a copy of the packet. A set of n measurements corresponds to the results of sending n such packets.

Note that it is natural here to think of the leaf observations $\{X_1, \dots, X_{N_l}\}$ as arising through the result of a process originating at the root of the underlying tree. Specifically, for a multicast tree T , each vertex $j \in V_T$ can be assigned a binary variable X_j indicating whether or not it received a copy of the packet. The resulting process $\{X_j\}_{j \in V_T}$ will then have certain hereditary constraints to it. More precisely, for a subset $U \subset R$ of the leaves, denote by $a(U)$ the index of that vertex which is the closest common ancestor of the vertices in U . Similarly, for an internal vertex j , denote by $d(j)$ the indices of all immediate descendants (i.e., children) of j . Then $X_{a(U)} = 0$ implies that $X_j = 0$ for all $j \in U$, while if $X_j = 1$ for at least one $j \in d(k)$, then $X_k = 1$.

We will see that this type of structure can be exploited in designing methods of inference. \square

Binary measurements also can arise in phylogenetic tree inference. For example, DNA sequence data are often used to provide measurements on species, but sometimes the four DNA bases $\{A, G, C, T\}$ are grouped in pairs $\{A, G\}$ and $\{C, T\}$ (i.e., by purines and pyrimidines, respectively) and simply coded 0 and 1. An N_l -tuple of measurements then indicates whether each of N_l species being studied had a purine or pyrimidine at a given location in the genome. A set of n measurements corresponds to such N_l -tuples at n different locations. The notion of a tree-based process, generating sequence data at the leaves, is also common in this context, where it is motivated by the theory of evolution, as we will discuss briefly later in Section 7.4.3.

There are a number of classes of methods that have been proposed for tomographic inference of tree topologies. They differ in, for example, how the data are utilized (e.g., all or just part), the criteria for assessing the merit of a candidate tree $T \in \mathcal{T}_{N_l}$ in describing the data, and the manner in which the space \mathcal{T}_{N_l} is searched in attempting to find a tree(s) best meeting this criteria. We will focus on two popular classes of methods: (i) those based on hierarchical clustering and related ideas, and (ii) likelihood-based methods.

In the methods of both of these approaches, although the technical infrastructure can be quite different, the underlying idea is essentially the same. Specifically, they seek to exploit the supposition that, the closer two leaves are in the underlying tree, the more similar their observed characteristics (i.e., the measurements) will be. For example, for computer networks, in the case of the multicast measurements just described, in Example 7.4, it is expected that two destination addresses will be more

⁹ It would be received if there are no additional routers along the branch from i_1 to 3 that might lose it. If there are additional routers, then it is not certain that the packet would be received. But with the topology dictated by a given choice of leaves and root, we are effectively blind to the number of additional such routers present along any given branch. This issue lies at the heart of the distinction between physical and logical Internet topologies, and will also be seen to be relevant to constructing inferential methods.

likely to both receive or not receive a copy of the packet if they share more of the same path from the root. Similarly, two biological species are presumed to share more of their genome if they split from a common ancestor later in evolutionary time.

Note that regardless of the methodology used, if we truly aim to search over all of \mathcal{T}_{N_l} , we are faced with a daunting computational task. In particular, the space \mathcal{T}_{N_l} of (semi)labeled, rooted, binary trees is found to have

$$3 \times 5 \times 7 \times 9 \times 11 \times \dots \times (2N_l - 3) = (2N_l - 3)!! \quad (7.39)$$

elements (e.g., Felsenstein [141, Ch. 3]). Using Stirling's approximation, it can be shown that this quantity grows like $(N_l - 1)^{N_l - 1}$. Accordingly, many of the specific tree-selection problems posed in this area have been shown to be *NP*-hard, although sometimes clever use of efficient search strategies can make exhaustive search computationally feasible in problems of relatively small, but non-trivial, magnitude. An example is the 'maximum parsimony' problem in phylogenetic inference, which seeks to construct a tree for data involving the minimum number of necessary evolutionary changes, and for which the 'branch-and-bound' algorithm can be used to greatly speed up the search of \mathcal{T}_{N_l} . See Felsenstein [141, Ch. 5]). In general, however, exhaustive search is unrealistic and alternative approaches, based on greedy or randomized search algorithms, are utilized.

7.4.2 Methods Based on Hierarchical Clustering

Given a set of N_l objects, recall from our discussion in Section 4.3.3.1 that hierarchical clustering is concerned with the grouping of those objects into hierarchically nested sets of partitions, based on some notion of their (dis)similarity. And, in particular, recall that generally these nested partitions are represented using a tree. The usage of hierarchical clustering we saw in Section 4.3.3.1 was for the purpose of graph clustering. But it also is a natural tool to use for the tomographic inference of tree topologies, where we treat the N_l leaves as the 'objects' to be clustered and the tree corresponding to the resulting clustering as our inferred tree \hat{T} . It is perhaps worth emphasizing that, whereas in standard applications of hierarchical clustering the goal often ultimately is to select just one of the nested partitions to represent the data, here the tree corresponding to the entire set of partitions is our focus.

Recall that hierarchical clustering methods require some notion of (dis)similarity, which is usually constructed from the observed data, but sometimes measured directly. In the context of our tomographic inference problem, it is logical to use the n observations of the random variables $\{X_1, \dots, X_{N_l}\}$ at the leaves to derive an $N_l \times N_l$ matrix of (dis)similarities. In principle, standard notions of distance between vectors may be used, followed by an off-the-shelf hierarchical clustering algorithm of choice. See Gordon [185] or Jain, Murty, and Flynn [213], for example, for extensive reviews. In practice, however, it can be important to tailor the notion of

(dis)similarity to the tomographic inference problem at hand. We illustrate through the following variant of agglomerative hierarchical clustering for the multicast measurements of Example 7.4.

Example 7.5 (Agglomerative Hierarchical Clustering for Multicast Data). Ratnasamy and McCanne [323] observe that the observed rate of shared losses of packets should be fairly indicative of how close two leaf vertices (i.e., destination addresses) are on a multicast tree T . They propose a greedy, clustering-based algorithm for computer network topology identification that exploits this observation. Duffield, Horowitz, Lo Presti, and Towsley [128] formalize and extend their algorithm and establish certain of its properties, and it is their notation that we adopt here.

An important observation in this problem is that there are two different types of shared loss between a pair of leaf vertices $\{j, k\}$ – termed ‘true’ and ‘false’ shared loss by Ratnasamy and McCanne [323]. The true shared losses are due to loss of packets on the path common to the vertices i and j , while the false shared losses are due to loss on the parts of paths following after the closest common ancestor, $a(\{j, k\})$. For example, in the tree of Figure 7.8, true shared losses for the leaves 1 and 3 would be losses incurred on the path from r to the internal vertex i_1 ; false shared losses would refer to cases where packets were lost separately on the two paths from i_1 to the vertices 1 and 3, respectively. Since the net shared loss rate (i.e., the fraction of packets commonly lost to j and k) includes the contribution of both types of losses, it can be misleading to use this number as a similarity. Fortunately, it is possible to obtain information on the true loss rates from these net loss rates, through the use of a simple packet-loss model.

Specifically, consider the cascade process $\{X_j\}_{j \in V_T}$ described previously. We assume $X_r = 1$. For each internal vertex k , if $X_k = 0$, then $X_j = 0$ for all $j \in d(k)$. If not, then we assume

$$\mathbb{P}(X_j = 1 | X_k = 1) = 1 - \mathbb{P}(X_j = 0 | X_k = 1) = \alpha_j , \quad (7.40)$$

where $0 < \alpha_j < 1$ for all j . That is, we specify a Markov-like model down the tree, from the root to the leaves. The quantity $A(k) = \prod_{j \succ k} \alpha_j$ is the probability that a packet is successfully transmitted from r to k , where $j \succ k$ indicates ancestral vertices of k on the path from r . The true shared loss rate for two leaf vertices $j, k \in R$ is then just $1 - A(a(\{j, k\}))$.

Now define $R(k)$ to be the set of leaf vertices in R that are descendants of k , for interior vertices k , and let

$$\gamma(k) = \mathbb{P}(\cup_{j \in R(k)} \{X_j = 1\}) \quad (7.41)$$

be the probability that at least one of the leaves descended from k receive a packet. Similarly, for an arbitrary subset of vertices U , let

$$\gamma(U) = \mathbb{P}(\cup_{k \in U} \cup_{j \in R(k)} \{X_j = 1\}) \quad (7.42)$$

be the probability that at least one of the leaves descended from the vertices in U receive a packet. In the case $U = d(k)$, we have $\gamma(U) = \gamma(k)$. It is not difficult to

show that for any $U \subseteq d(k)$, we have the relation

$$[1 - \gamma(U)/A(k)] = \prod_{j \in U} [1 - \gamma(j)/A(k)] . \quad (7.43)$$

Therefore, given the values $\{\gamma(k)\}_{k \in V_T}$, we can find $\{A(k)\}_{k \in V_T}$ as the unique solutions to the appropriate equations (7.43). See Caceres, Duffield, Horowitz, and Towsley [71] and Duffield, Horowitz, Lo Presti, and Towsley [128].

Although the $\gamma(k)$ are not available to us, they can be estimated by the relative frequencies

$$\hat{\gamma}(k) = (1/n) \sum_{i=1}^n \left[\prod_{j \in R(k)} x_j^{(i)} \right] , \quad (7.44)$$

where $x_j^{(i)}$ is the observed value of X_j corresponding to the i -th sample. Estimates $\hat{A}(k)$ then follow accordingly. A greedy, agglomerative algorithm may then be defined as follows. For all pairs of vertices $j, k \in R$, the estimated shared loss rate $1 - \hat{A}(a(\{j, k\}))$ is calculated, where the ancestor $a(\{j, k\})$ is implicitly defined through the proposed grouping of j and k . That pair j, k for which this loss rate is maximized are merged and replaced by their corresponding ancestor vertex. The process is then repeated on the $N_l - 1$ remaining nodes (i.e., $a(\{j, k\})$ and $R \setminus \{j, k\}$) and iterated similarly thereafter, until a full tree is formed.

Duffield, Horowitz, Lo Presti, and Towsley [128] provide a formal statement of this algorithm and prove the consistency of the resulting estimator \hat{T} for recovering a binary multicast tree T , under the model assumptions. These authors also present and examine three extensions of this algorithm to the problem of inferring non-binary trees. \square

Hierarchical clustering methods have also been used quite frequently in phylogenetic inference, and in fact were some of the earliest proposed methods for this task. For DNA sequence data, it is standard to use some notion of dissimilarity of the genetic material of pairs of species as input for the clustering algorithm. A dissimilarity of this sort is commonly termed a ‘genetic distance,’ and methods involving such distances are called ‘distance-matrix’ methods. There are numerous measures of genetic distance. The simplest is just the fraction of mismatched elements in two (aligned) DNA sequences. More sophisticated measures attempt to correct for various expected characteristics of DNA evolution. These corrections often are done in a model-based fashion, using Markov-like cascade models similar in principle to that described above for the multicast data. Sometimes, the distances obtained from the data are treated as empirical versions of some true, unknown distances (e.g., along the branches of the underlying phylogenetic tree), and the empirical distances are used to estimate the true distances through, for example, least-squares methods. See Felsenstein [141, Ch. 11] for an overview of distance-matrix methods and a detailed discussion of some of the potentially critical issues associated with how genetic distance is defined. The property of consistency has been shown to hold for

certain distance-matrix methods of phylogenetic tree estimation, under appropriate conditions.

7.4.3 Likelihood-based Methods

We have seen that probability models for the leaf random variables $\{X_1, \dots, X_{N_l}\}$ are sometimes used in defining (dis)similarities in hierarchical clustering methods for tomographic inference of trees. But if we are willing to specify probability models, then we have the potential for likelihood-based methods of inference as well.

Suppose we have specified a conditional density or probability mass function $f(\mathbf{x}|T)$ for the N_l -length vector of random variables $\mathbf{X} = (X_1, \dots, X_{N_l})^T$, given a tree-topology T . If we assume independence among the n observations $\{\mathbf{x}^{(i)}\}_{i=1}^n$, then the likelihood has the form

$$\mathcal{L}(T) = \prod_{i=1}^n f(\mathbf{x}^{(i)}|T) , \quad (7.45)$$

and the maximum likelihood estimator is simply

$$\hat{T}_{ML} = \arg \max_{T \in \mathcal{T}_{N_l}} \mathcal{L}(T) . \quad (7.46)$$

However, it is typically the case that we specify a probability model involving additional parameters besides T , that is, $f(\mathbf{x}|T, \theta)$, for some m -length parameter vector θ . For example, θ might contain parameters relating to the evolution of a cascade process $\{X_j\}_{j \in V_T}$ down a tree T . Then the likelihood in (7.45) is an integrated likelihood, in that it takes the form

$$\mathcal{L}(T) = \prod_{i=1}^n \int f(\mathbf{x}^{(i)}|T, \theta) f(\theta|T) d\theta , \quad (7.47)$$

where $f(\theta|T)$ is an appropriately defined distribution on θ , given T . Depending on the nature of their integrands, the integrals in (7.47) may or may not lend themselves well to computational evaluation.

An alternative that can be more computationally tractable, or that can be used when we cannot or do not wish to specify a distribution $f(\theta|T)$, is to define \hat{T} through maximization of a profile likelihood. Specifically, let

$$\mathcal{L}(T, \theta) = \prod_{i=1}^n f(\mathbf{x}^{(i)}|T, \theta) , \quad (7.48)$$

and define

$$\hat{\theta}_T = \arg \max_{\theta} \mathcal{L}(T, \theta) . \quad (7.49)$$

Then let

$$\widehat{T}_{PL} = \arg \max_{T \in \mathcal{T}_{N_l}} \mathcal{L}(T, \hat{\theta}_T) . \quad (7.50)$$

The quantity $\mathcal{L}(T, \hat{\theta}_T)$ is called the profile likelihood, and \widehat{T}_{PL} , the maximum profile likelihood estimator. See Berger, Liseo, and Wolpert [35] for an overview of issues and trade-offs related to the use of integrated versus profile likelihoods.

We illustrate some of these concepts in the context of the multicast data described earlier.

Example 7.6 (Profile Maximum Likelihood Estimation of Multicast Trees). Recall the cascade model for multicast measurements, surrounding equation (7.40), in Example 7.5. The likelihood (7.48), with θ replaced by a vector α containing the parameters $\{\alpha_j\}$, takes the form

$$\mathcal{L}(T, \alpha) = \prod_{i=1}^n \prod_{j \in V_T} \eta_j^{(i)} , \quad (7.51)$$

where

$$\eta_j^{(i)} = \begin{cases} 1, & \text{if } j = r , \\ \alpha_j^{x_j^{(i)}} (1 - \alpha_j)^{1 - x_j^{(i)}} , & \text{if } x_{pa(j)}^{(i)} = 1 , \\ 1, & \text{if } x_{pa(j)}^{(i)} = 0 , \end{cases} \quad (7.52)$$

and $pa(j)$ denotes the parent of j in T . Duffield, Horowitz, Lo Presti, and Towsley [128] propose the profile maximum likelihood estimator

$$\widehat{T}_{PL} = \arg \max_{T \in \mathcal{T}_{N_l}} \mathcal{L}(T, \hat{\alpha}_T) , \quad (7.53)$$

where

$$\hat{\alpha}_T = \arg \max_{\alpha} \mathcal{L}(T, \alpha) , \quad (7.54)$$

and prove its consistency under appropriate conditions.

Implementation of this estimator, however, is non-trivial. For example, consider the estimation of α , a parameter vector of potentially quite large dimension. An efficient, recursive estimation algorithm is proposed by Caceres, Duffield, Horowitz, and Towsley [71]. These authors show that the elements α_k may be recovered uniquely from the values $A(k)$, through the relation

$$\alpha_k = A(k)/A(pa(k)) . \quad (7.55)$$

And recall that the $A(k)$ may be obtained from the values $\gamma(k)$ through equation (7.43). Accordingly, estimates, say $\tilde{\alpha}_k$, of the α_k may be obtained from the estimates $\hat{\gamma}(k)$ of the $\gamma(k)$, defined in (7.44). Although the resulting estimator $\tilde{\alpha}$ is not strictly the maximum likelihood estimate, Caceres et al. show that, when $\alpha_k \in (0, 1)$ for all k , we nevertheless have $\tilde{\alpha} = \hat{\alpha}$ with high probability.

As for the optimization of the profile likelihood $\mathcal{L}(T, \hat{\alpha}_T)$ in T , Duffield, Horowitz, Lo Presti, and Towsley [128] seem to have performed this optimization

only in simulation and for trees with very small values of N_l , such as $N_l = 3$ and 5, for which they were able to conduct an exhaustive search of \mathcal{T}_{N_l} . For topologies of less trivial sizes, presumably either greedy techniques (like those mentioned below) or MCMC-based techniques (like that we describe later in Section 7.4.5) may be adopted. \square

Likelihood-based methods are also used in phylogenetic inference. Similar to the case of our multicast measurements, it is common in this context to assume independence between measurements $\mathbf{x}^{(i)}$ (e.g., between locations in the measured DNA sequence) and, as was remarked earlier, to use Markov-like cascade models to capture the effects of evolution through time. A simple example of such a model, in the case of binary measurements (e.g., when the DNA base characters are coded simply as purines or pyrimidines), is a symmetric model of change between the two states of zero and one, in traversing from one end of a branch to the other, according to the probability

$$\mathbb{P}(X_j = 1 | X_k = 0) = \frac{1}{2} (1 - e^{-2w_j}) , \quad (7.56)$$

where $j \in d(k)$ and w_j is the length of the branch leading to j . More sophisticated models, for various types of molecular data, are discussed in Felsenstein [141, Chs. 13–15].

The likelihood corresponding to this model is expressed as

$$\mathcal{L}(T, \mathbf{w}) = \prod_{i=1}^n \sum_{\{\mathbf{x}_j^{(i)}\}_{j \in V_T \setminus R}} \prod_{j \in V_T} \eta_j^{(i)} , \quad (7.57)$$

where

$$\eta_j^{(i)} = \begin{cases} \frac{1}{2} (1 - e^{-2w_j}), & \text{if } x_j^{(i)} \neq x_{pa(j)}^{(i)}, \\ \frac{1}{2} (1 + e^{-2w_j}), & \text{if } x_j^{(i)} = x_{pa(j)}^{(i)}. \end{cases} \quad (7.58)$$

Note that the expression in (7.57) is similar to that in (7.51), except for the presence of an additional summation term between the two products. This summation is over all possible ways that the states zero or one can be assigned to the internal vertices of the tree T , and is absent in the case of multicast data because of the hereditary constraints on the process $\{X_j\}_{j \in V_T}$. In general, the presence of such product-sum combinations can be problematic from a computational perspective. However, in this particular case, the likelihood can be calculated efficiently using a dynamic programming algorithm – the so-called pruning algorithm – proposed by Felsenstein [142]. Working recursively from the leaves towards the root, components of the likelihood are computed on sub-trees and combined in a clever fashion to yield the likelihoods on larger sub-trees containing them, until the likelihood for the entire tree is obtained.

Because the branch-lengths \mathbf{w} are meaningful and important in a phylogeny, the maximum likelihood approach aims to find the pair $(\hat{T}, \hat{\mathbf{w}})$ that maximizes (7.57). This problem was only recently shown by Chor and Tuller [82] to be *NP*-hard, although it was widely suspected to be so for some time. In practice, a profile max-

imum likelihood method typically is used, as described, for example, in Felsenstein [143]. For a given tree T , optimal branch lengths $\hat{\mathbf{w}}_T$ can be calculated using an algorithm that generalizes the pruning algorithm mentioned above. The space \mathcal{T}_{N_l} of trees T is then explored in a greedy fashion. For example, Felsenstein [143] proposes an algorithm that grows a tree by sequentially adding species one at a time, where the branch is placed in a position optimal at the current stage of the calculations, and which pauses between each new addition to explore the possibility of ‘local re-arrangements.’ This algorithm will not search all of tree space, of course, but it is argued that it will explore at least $2N_l^2 - 9N_l + 8$ different topologies.

In general, maximum likelihood inference of tree topologies may also be pursued through the use of MCMC, such as we will do in the case study of Section 7.4.5. And MCMC is critical to the use of Bayesian methods to tomographic inference of trees. With a likelihood in hand, if we are willing and able to specify prior distributions for trees T and parameters θ , then inference may be based upon the posterior $\mathbb{P}(T|\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(n)})$. A sizable sub-literature exists on Bayesian methods for phylogenetic inference. See the review by Huelsenbeck, Ronquist, Nielsen, and Bollback [207], for example, and also Felsenstein [141, Ch. 18]. Bayesian methods also have been suggested in the computer network context. See Duffield, Horowitz, Lo Presti, and Towsley [128], for example, in the case of multicast measurements. For a prior on the space \mathcal{T}_{N_l} of trees, the uniform distribution is a common choice. See Felsenstein [141, Ch. 18] for a discussion of the choice of prior in the context of phylogenetic inference, including various non-uniform priors. Felsenstein also provides some discussion on the important issue of appropriate design of MCMC algorithms to ensure effective exploration of the space of trees \mathcal{T}_{N_l} .

7.4.4 Summarizing Collections of Trees

All of the material so far in this section has focused on methods for producing a single inferred tree \widehat{T} . That is, we have focused on roughly the equivalent of a point estimate in standard statistical inference. But a much richer inferential statement would involve producing a collection of trees, in the spirit of an interval estimate. In practice, such collections arise in various ways, such as listing a number of trees of nearly maximum likelihood, rather than simply a single maximum-likelihood tree, or from bootstrap re-sampling in an effort to assess the variability in an inferred tree, or from MCMC sampling of an appropriate posterior distribution on \mathcal{T}_{N_l} .

As an illustration, consider the process of bootstrapping in tomographic tree inference, which has been studied extensively in the context of phylogenetic trees (e.g., see Felsenstein [141, Ch. 20]). The essence of the basic method in this context is identical to its implementation in more standard problems, which is described in Efron and Tibshirani [131], for example. Given a sample $\{\mathbf{x}^{(i)}\}_{i=1}^n$ of size n , assumed to consist of independent and identically distributed draws from an underlying distribution, we draw a set of B bootstrap samples, say $\{\mathbf{x}^{i,b}\}_{i=1}^n$, for $b = 1, \dots, B$, each of which derives from random sampling with replacement from the original sam-

ple. For each bootstrap sample, we apply the same tree-inference method used to construct our original estimate \widehat{T} from the original data, resulting in a collection of estimates $\{\widehat{T}^{(b)}\}_{b=1}^B$.

Given such a collection of trees, how can we usefully summarize the information therein? Of course, what is ‘useful’ will depend to no small extent on the questions being asked. But two common ways to think about summarizing collections of trees are by concentrating on (i) how they are similar and (ii) how they are different.

For thinking about similarities among trees, consensus methods are popular. A *consensus tree* is a single tree that aims to summarize the information in a collection of trees in a ‘representative’ manner. There are many methods that have been proposed for defining consensus trees. For example, Margush and McMorris [268] define a class of consensus trees, called M_l -trees, indexed by a parameter $l \in [0.5, 1]$. An M_l -tree contains all groups of leaves $U \subseteq R$ (i.e., as determined by the branching) that occur in more than a fraction l of the trees in a collection. In the case of $l = 1.0$, we have what is termed a *strict consensus tree*, while for $l = 0.5$, it is called a *majority-rule consensus tree*. The computation of an M_l -tree over a collection of t trees with N_l leaves has traditionally been done using $O(tN_l^2)$ algorithms, but Amenta, Clarke, and St. John [9] have more recently proposed a randomized algorithm with $O(tN_l)$ expected run time. A comprehensive summary and classification of these and additional consensus tree methods can be found in the review by Bryant [68].

Note that additional information can be added to a consensus tree in the form of branch weights, where appropriate. For example, in phylogenetic inference, when the collection of trees results from bootstrapping, it is common to annotate each branch in a majority-rule consensus tree with the fraction of bootstrap trees $\widehat{T}^{(b)}$ that contain its corresponding group of leaves.¹⁰

For thinking about differences among trees, various notions of distance between pairs of trees have been employed. For example, the *symmetric difference*, introduced by Bourque [49] and by Robinson and Foulds [332], works by counting branches. More specifically, given two rooted trees, each branch of each tree is associated with the group of leaf vertices descended from it (similar as to the M_l -tree consensus methods described above), and the symmetric difference is defined to be the number of groups that differ (i.e., that are induced by one tree but not both). It is not hard to show that this quantity forms a proper distance (i.e., a metric) on \mathcal{T}_{N_l} . Another popular quantity for quantifying the difference between two trees is the *nearest-neighbor interchange* (NNI) distance, which essentially counts the number of swaps of adjacent branches that must be made to transform one tree into the other.¹¹ Unfortunately, the task of calculating the NNI distance is computationally daunting. Li and Zhang [257] have shown a similar problem is *NP*-complete. In

¹⁰ These annotations, intended as estimates of the evidence in support of each branch, have been noted to have a particular type of bias to them. Various methods, based on more sophisticated resampling schemes, have been suggested for correction of this bias. See Felsenstein [141, Ch. 20], for details and references.

¹¹ More formally, it is associated with counting the number of ‘rotation’ operations, of two ‘sibling’ branches about their common vertex, needed to move from one tree to the other in \mathcal{T}_{N_l} .

practice, greedy approximation algorithms are used (e.g., see Felsenstein [141, Ch. 4]). NNI distance too can be shown to be a metric on \mathcal{T}_{N_t} .

Perhaps not surprisingly, relationships can be demonstrated between certain consensus measures and difference measures. For example, Barthélémy and McMorris [21] show that, if we define a *median tree* in a collection to be a tree T whose total distance – based on the symmetric difference – to all other trees is a minimum, then this tree is equivalent to the majority-rule tree, when the number of trees t in the collection is odd.

In general, as Holmes [205] has remarked, there remains much to be done in the area of formal methods of comparisons of trees (not to mention more general graphs), and in the usage of such methods for higher-level tasks of statistical modeling and inference.

7.4.5 Case Study: Computer Network Topology Identification

We have already encountered, through the several examples running throughout this section, an illustration of the basic structure of a computer network topology identification problem and its solution, in the context of the binary multicast-based measurements of packet loss. Here we present an extended illustration of the same problem, but based on somewhat different measurements of a continuous type. The data and analysis we describe are taken from the work of Coates and Nowak and their colleagues (e.g., Coates et al. [95], Castro, Coates, and Nowak [79]).

In properly functioning computer networks, the loss of packets should be a relatively rare phenomenon (e.g., with a rate of less than 2%). While this fact is good news for network users, it also means that multicast-based methods can have trouble discerning the underlying topology in such circumstances due to lack of information. An alternative is to utilize the time delays between when packets are sent and when they are received, which yields a set of measurements that have been found to provide greater discernibility.

For example, Coates et al. [95] propose a measurement scheme they call ‘sandwich probing,’ which sends a sequence of three packets, two small and one large. The large packet is sent second, after the first small packet and before the second small packet. The two small packets are sent to one of a pair $\{i, j\}$ of leaf vertices, say i , while the large packet is sent to the other, that is, j . The basic idea is that the large packet induces a greater delay in the second small packet, compared to that of the first, and that the difference in delays of the two small packets will vary in a manner reflective of how much of the paths are shared from the origin to i and j . For example, in Figure 7.8, we expect a longer difference in delays for the two small packets when they are sent to vertex 1, and the large, to vertex 2, as compared to when the large is instead sent to vertex 3. This reasoning follows from the fact that in the former case the second small packet lingers behind the large packet from r until i_2 , while in the latter case, it does so only from r until i_1 .

Figure 7.10 (at the top) shows the logical topology, during a period of 2001, of that portion of the Internet leading from a desktop computer in the Electrical and Computer Engineering Department at Rice University to similar machines at ten other university locations. Specifically, two of the destination machines were located on a different sub-network at Rice, two at separate locations in Portugal, and six at four other universities in the United States. The topology shown here was itself measured, using the `traceroute` tool described in Section 3.5.2, and therefore is not a perfect representation of the true logical topology, but is an acceptable proxy for a ‘groundtruth’.¹²

Coates et al. [95] conducted an experiment in which, over a period of eight minutes, a sandwich probe was sent to a randomly chosen pair of these ten destinations, once every 50 ms. These pairs included self-pairs (i.e., where the two small packets and the large packet were all sent to the same destination), which allows for a type of calibration in interpreting the measurements. For each probe, the difference in delay measured in the two small packets was recorded. The experiment resulted in a total of 9,567 measured delay differences, after accounting for probes with packet losses, of which there were very few. Figure 7.9 shows an image representation of the mean delay differences, symmetrized to eliminate variations within each pair due to receipt of the small packets versus the large packet. The hierarchical relationship among the destinations in the logical topology in Figure 7.10 is clearly evident in the relative magnitudes of the mean delay differences. This association can be used to infer the topology from the delay differences.

More specifically, consider the following model. For each pair of destinations $i, j \in R$, let x_{ij} be the average delay difference, over the course of the experiment, where the first index, i , refers to the destination receiving the two small packets, and the second, j , the large packet. The central limit theorem suggests that the x_{ij} , as averages, will be approximately Gaussian in distribution, say with means μ_{ij} and variances σ_{ij}^2 . In addition, based on the construction of the experiment, an assumption of independence between the x_{ij} is not unreasonable. Coates et al. [95] use these basic Gaussian model components to derive two methods of topology inference for this data, the results of which are shown in Figure 7.10.

The first inferred topology derives from the agglomerative likelihood tree (ALT) method of Castro, Coates, and Nowak [79], which is basically a hierarchical clustering algorithm with a likelihood-based measure of similarity. More specifically, note that since the expected delay difference μ_{ij} is ideally the sum of expected delay differences on all branches shared along the paths from the root to the destinations i and j , it is appealing as a measure of similarity between i and j . We obviously do not know the μ_{ij} , but can estimate them from the x_{ij} . Let $\ell_{ij}(\mu) = \log f(x_{ij}|\mu)$ denote the Gaussian log-likelihood for the single variable x_{ij} , as a function of its mean μ , where we have suppressed the dependence on σ_{ij}^2 , which we assume to be known

¹² Of course, one may ask, if `traceroute` may be used to measure and construct such topologies, why is there even a need for the tomographic methods described here? In reality, `traceroute` has its own problems, not the least of which is the fact that an increasingly greater fraction of routing devices in the Internet over the years have had their ability to interact with `traceroute` probes ‘turned off’ by network administrators, in part due to security concerns.

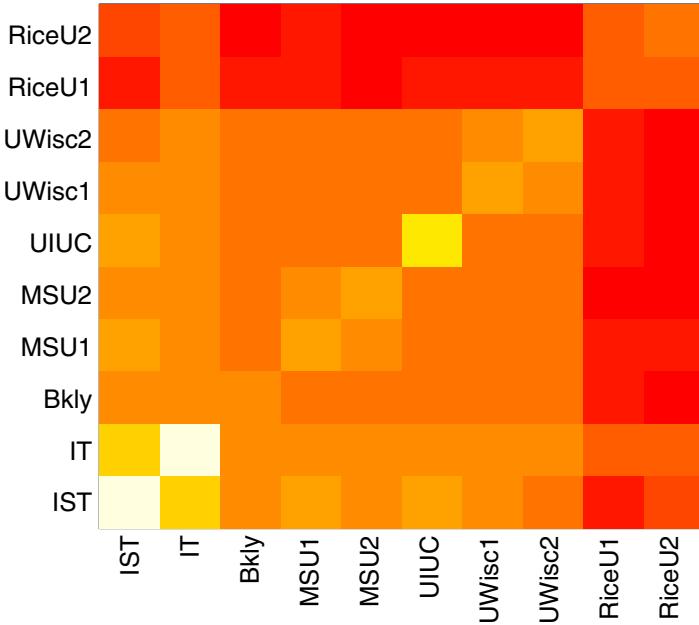


Fig. 7.9 Image representation of pairwise delay differences in the data of Coates et al. [95], with increasing darker red corresponding to lower values, and increasingly brighter yellow, to higher values.

(or, practically speaking, well estimated from the data). Let S be a set of vertices, initialized to be the set of leaves (i.e., $S = R$), and define our similarity measure among leaves to be the estimated mean delay difference

$$\hat{\mu}_{ij} = \hat{\mu}_{ji} = \arg \max_{\mu} [\ell_{ij}(\mu) + \ell_{ji}(\mu)] , \quad (7.59)$$

for all distinct pairs $i, j \in S$. The ALT algorithm merges those two leaves $i, j \in R$ with the largest value $\hat{\mu}_{ij}$, and replaces them in the set S by an ancestor vertex $a(\{i, j\})$. The algorithm then iterates, calculating

$$\hat{\mu}_{kl} = \hat{\mu}_{lk} = \arg \max_{\mu} \sum_{m \in R(k)} [\ell_{ml}(\mu) + \ell_{lm}(\mu)] , \quad (7.60)$$

for all distinct pairs $k, l \in S$, until the set S consists of a single vertex. Recall that $R(k)$ denotes the set of leaf vertices in R descended from an interior vertex k .

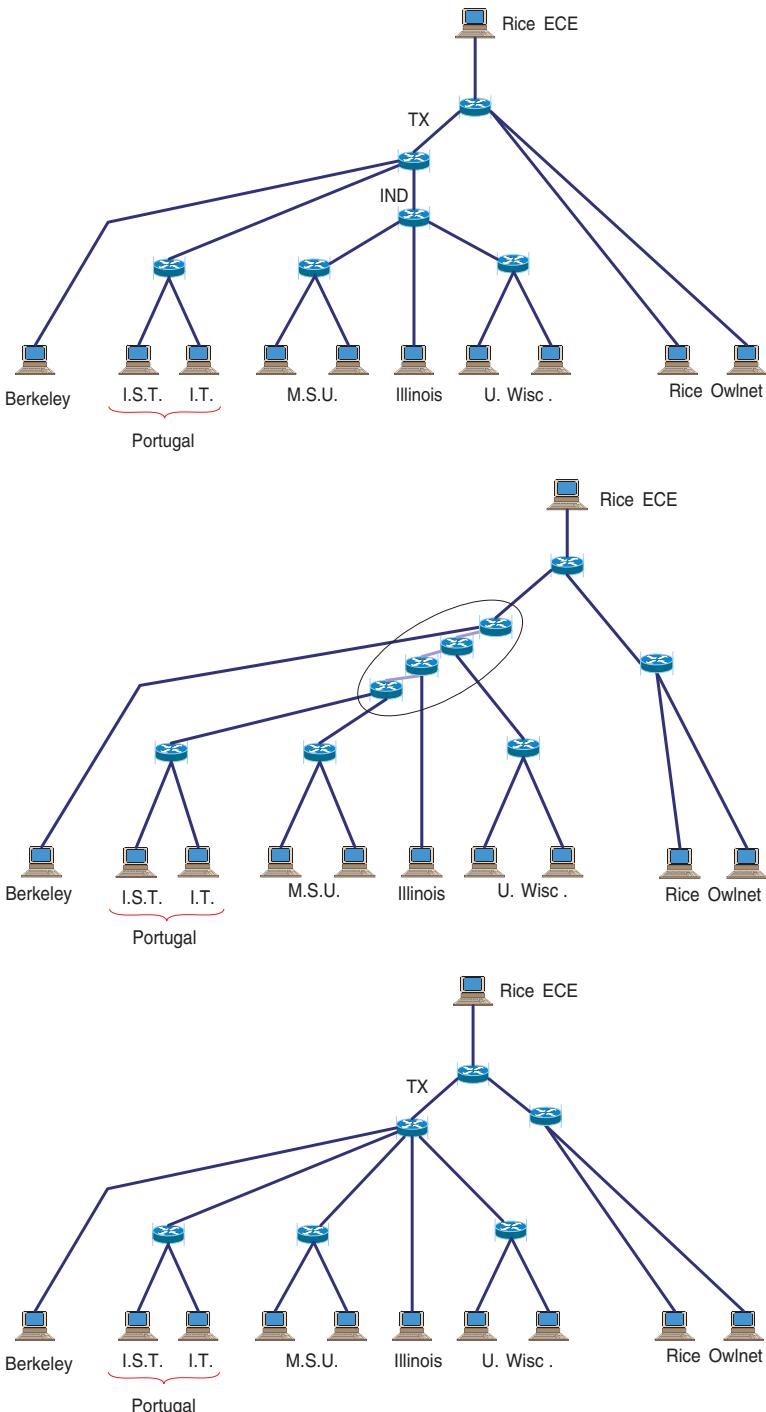


Fig. 7.10 Computer network topologies from the experiment of Castro, Coates, and Nowak [79]. Top: Groundtruth topology, based on traceroute measurements. Middle: Inferred topology, based on ‘sandwich’ probe measurements, using a hierarchical clustering algorithm. Bottom: Inferred topology, based on the same measurements, using a complexity penalized maximum likelihood method. Figures courtesy of Rui Castro.

We note that the ALT algorithm is similar to the unweighted pair group method with arithmetic mean (UPGMA), also known as the average linkage method, which was developed originally in the context of phylogenetic inference by Sokal and Michener [362].

The second inferred topology in Figure 7.10 derives from a complexity penalized maximum (profile) likelihood method, proposed by Coates et al. [95]. More precisely, the topology is defined through the expression

$$\hat{T} = \arg \max_T [\log \mathcal{L}(T, \hat{v}_T) - \lambda J(T)] , \quad (7.61)$$

where $\mathcal{L}(T, v)$ is the likelihood of the full set of observations, as a function of the tree T and a parameter vector v , $J(T)$ is a penalty that counts the number of branches in T , and $\lambda \geq 0$ is a user-specified smoothing parameter. The vector $v = v_T$ contains, for a given tree T , the set of expected delay differences on the individual branches of T , the various sums of which determine the expected delay differences μ_{ij} on the shared portions of paths to all pairs of leaf vertices $i, j \in R$. The quantity \hat{v}_T is the maximum likelihood estimate of v , conditional on T . The function $J(T)$ acts to encourage the use of parsimonious representations of the data, while λ dictates the trade-off between model goodness-of-fit and model complexity. Together, through the term $-\lambda J(T)$, they essentially act in the role of a (unnormalized) log-prior on T , and \hat{T} , in the role of a *maximum a posteriori* estimator. The optimization in (7.61) is over all trees (i.e., binary and non-binary) with N_l leaf vertices.

Computation of the complexity-penalized maximum likelihood estimate in (7.61) is facilitated by the use of MCMC sampling. Coates et al. [95] propose the use of a standard Metropolis-Hastings algorithm, where the proposed moves from tree to tree are controlled through insertion or deletion of individual interior vertices between two existing vertices. See Coates et al. [95] and Castro, Coates, and Nowak [79] for details. Although this approach is rather simple, compared to many of the MCMC methods in this area, the algorithm appears to work quite well, a fact that the authors attribute in good part to their use of the ALT-inferred tree as a starting value.

Looking at the two inferred topologies, and the groundtruth topology, we can see a number of interesting differences. First, while the ALT topology is, naturally, binary, the maximum likelihood topology is not. The biggest implication of this difference seems to be in the inclusion in the ALT topology of a sequence of four closely spaced consecutive internal vertices where the maximum likelihood topology has only one. In fact, Castro, Coates, and Nowak [79] state that the improvement due to inclusion of these vertices is slight – an order of magnitude smaller than for other branches, and conclude that they are largely an artifact of the binary nature of the underlying hierarchical agglomeration methodology. Second, beyond this artifact, the ALT and maximum likelihood topologies are quite similar. Both, for example, miss the Indiana (IND) router, and both add an additional vertex between the Rice ECE root and the two Rice leaf vertices. Coates et al. [95] speculate that the Indiana router is missed because the connection between the Texas (TX) and Indiana routers is too high-speed to produce a measurable delay difference. Interestingly, upon further examination, the extra vertex between the Rice machines was found to be real

– a so-called switching device that, not being itself a router, would not be detectable by `traceroute` but does indeed produce a *bona fide* branching in the underlying logical topology.

7.5 Additional Related Topics and Reading

We have focused our discussion in this chapter upon three arguably canonical problems in network inference. But various other network inference problems have been considered in the literature as well, arising from specific applications. Two examples are the problem of inferring networks from so-called co-occurrence data, as considered in Rabbat, Figueiredo, and Nowak [318, 319], motivated by measurements in telecommunications and in genetics, and the problem of inferring protein complexes, as considered by Scholtens and Gentleman [343]. Another interesting variation, related to some extent with the former of these two problems, is that of inferring networks with links defined according to the dynamic interactions of elements in a system (e.g., such as of information passing between neurons firing in sequence). See Smith et al. [355] for an example of such ‘information flow networks.’ Finally, while our presentation of the material on tomographic network topology inference concentrated primarily on the two contexts of computer networks and phylogenetic inference, related to the latter is the problem in biology of inferring coalescent trees, which have to do with the evolution of genes within a population (i.e., below the level of species). See Felsenstein [141, Ch. 26] for an overview and references.

Our treatment of association network inference in Section 7.3 was centered on the use of measures of association defined in terms of linear correlation. But other measures may be preferable, depending upon the nature of the data. For example, Spearman rank correlation and partial correlation might be used as robust alternatives to Pearson correlation (e.g., see Shipley [350, Sec. 3.9] and the discussion in de la Fuente, Bing, Hoeschele, and Mendes [110]). Or a measure capable of summarizing nonlinear association, such as mutual information, might be used, such as Faith et al. [140] have done in the context of gene regulatory network inference.

The Gaussian graphical models we encountered in Section 7.3.3 are, as was mentioned, just one representative of the broader class of graphical models. This class also includes, for example, models based on directed graphs and models for discrete data, both aspects of which have found substantial usage in problems of network inference. For problems of inferring biological networks, the graphical models formalism has been used in various ways with notable success. See, for example, the paper by Friedman, Linial, Nachman, and Pe’er [162], in the context of gene regulatory networks, and that by Sachs et al. [337], in the context of protein-signaling networks. However, with the adoption of such models in their full generality, there comes a number of non-trivial challenges for network inference in terms of model specification, fitting, and selection, especially in large-scale problems. While these issues are now reasonably well studied, they are not necessarily ‘solved,’ and a care-

ful development of the topic is beyond the scope of this book. Nevertheless, we shall revisit it briefly in Chapter 10.

Lastly, we point out, per our discussion in the introduction in Section 7.1, that while the adoption of a statistical inference framework allows for the *potential* for questions of validation of network inferences to be addressed in a precise and well-grounded manner, we still lack much of the technical infrastructure needed to do so to the extent we currently can with traditional, non-network problems. For example, most work on basic questions of consistency, consensus, robustness, and similar has only been done in the context of trees, in the area of phylogenetic inference, and even there the topic has been nowhere near exhausted. See Holmes [205], for example, for discussion in this vein, with respect to phylogenetic inference, and Ni, Xie, Tatikonda, and Yang [302], for progress in the area of computer network topology inference. More fundamentally, even basic questions regarding notions of accuracy merit substantial additional study. In most large-scale network inference problems an inferred graph \hat{G} will not fully match an underlying network graph G , and often waiting for increasingly larger sample sizes is unrealistic. Hence, consistency results can be less than compelling. An open area of inquiry, therefore, is how to best characterize the merit of partially accurate estimates. Such characterizations might depend in turn on intended high-level uses of \hat{G} , focusing for instance on the accuracy with which small subgraphs (e.g., network motifs) or paths are recovered.

Exercises

7.1. For a network of your choice, implement and compare some of the various methods for link prediction discussed in Section 7.2, using a cross-validation design like that used in the study presented in Section 7.2.3. Then examine the effect that modifying the mechanism of missingness in your data has on predictive performance. You should change the mechanism of missingness from missing-at-random to missing in some more substantive way. For example, you might allow missingness to be driven by the degrees of the vertices incident to an edge or, if available, vertex attributes.

7.2. For the partial correlation coefficients defined in (7.22), there exist recursive relationships among coefficients at adjacent orders $m - 1$ and m that allow for their efficient recursive calculation. Here we derive this relationship for the case of just three random variables, say X , Y , and Z , and $m = 1$. Specifically, you are asked to derive an expression relating $\rho_{XY|Z}$, the partial correlation of X and Y , given Z , to the Pearson product-moment correlations ρ_{XY} , ρ_{XZ} , and ρ_{YZ} .

For convenience, we express the covariance matrix Σ of $(X, Y, Z)^T$, in standardized form (i.e., as a correlation matrix), as

$$\Sigma = \left[\begin{array}{cc|c} 1 & \rho_{XY} & \rho_{XZ} \\ \rho_{YX} & 1 & \rho_{YZ} \\ \hline \rho_{ZX} & \rho_{ZY} & 1 \end{array} \right] ,$$

where the horizontal and vertical lines indicate a block partitioning in the manner of equation (7.24). Use (7.23) to derive a closed-form expression for the partial covariance matrix $\Sigma_{XY|Z}$. Then combine the elements of this matrix according to (7.22) to show that

$$\rho_{XY|Z} = \frac{\rho_{XY} - \rho_{XZ}\rho_{YZ}}{\sqrt{(1-\rho_{XZ}^2)(1-\rho_{YZ}^2)}} .$$

7.3. The regression-based approach to inference of a conditional independence graph G in a Gaussian graphical model is motivated by the optimization problem in (7.37). Here we derive the solution to that optimization problem in the case of three variables.

Let $(X, Y, Z)^T$ be a Gaussian random vector with zero mean and covariance matrix Σ , where Σ is defined as in Exercise 7.2. Consider the task of optimally predicting Z as a linear combination of X and Y , using mean-squared error as a cost function. That is, consider the optimization problem

$$\min_{\beta_{ZX}, \beta_{ZY}} \mathbb{E} \left[(Z - \beta_{ZX}X - \beta_{ZY}Y)^2 \right] . \quad (7.62)$$

a. Show that under our model assumptions the expectation in (7.62) takes the form

$$1 + \beta_{ZX}^2 + \beta_{ZY}^2 - 2\beta_{ZX}\rho_{ZX} - 2\beta_{ZY}\rho_{ZY} + 2\beta_{ZX}\beta_{ZY}\rho_{XY} .$$

b. Differentiating the expression in part (a), with respect to β_{ZX} and β_{ZY} and setting the resulting expressions equal to zero, show that the vector $(\beta_{ZX}, \beta_{ZY})^T$ must satisfy the linear system of equations

$$\begin{pmatrix} \rho_{ZX} \\ \rho_{ZY} \end{pmatrix} = \begin{bmatrix} 1 & \rho_{XY} \\ \rho_{YX} & 1 \end{bmatrix} \begin{pmatrix} \beta_{ZX} \\ \beta_{ZY} \end{pmatrix} .$$

c. Solve the above linear system to show that

$$\begin{pmatrix} \beta_{ZX} \\ \beta_{ZY} \end{pmatrix} = \begin{pmatrix} \frac{\rho_{ZX} - \rho_{ZY}\rho_{XY}}{1 - \rho_{XY}^2} \\ \frac{\rho_{ZY} - \rho_{ZX}\rho_{XY}}{1 - \rho_{XY}^2} \end{pmatrix} .$$

Compare this solution to the partial correlations $\rho_{ZX|Y}$ and $\rho_{ZY|X}$.

7.4. Consider the problem of tomographic inference of tree topologies.

- a. The multicast measurement model described in Examples 7.4 and 7.5 is straightforward to simulate. Using the tree $T = (V_T, E_T)$ shown in Figure 7.8, and probabilities $\{\alpha_j\}_{j \in V_T}$ of your choosing, write a computer program to simulate the cascade process $\{X_j\}_{j \in V_T}$ defined by (7.40), and hence measurements $\{X_j\}_{j \in R}$ at the leaf vertices $j \in R \subset V_T$.
- b. Implement the hierarchical clustering algorithm described¹³ in Example 7.5 and apply it to the simulated data generated by your work in part (a). Characterize the performance of this algorithm in recovering the underlying tree topology, for various choices of probabilities $\{\alpha_j\}_{j \in V_T}$.

¹³ For a formal statement of this algorithm, see Duffield, Horowitz, Lo Presti, and Towsley [128, Fig. 2].

Chapter 8

Modeling and Prediction for Processes on Network Graphs

Whereas the previous two chapters addressed the modeling and inference of network graphs, in this chapter we consider the network graph to be known. We focus on models and associated problems of statistical inference and prediction for processes defined on network graphs.

8.1 Introduction

Throughout this book so far, we have seen numerous examples of network graphs that provide representations – useful for various purposes – of the interaction among elements in a system under study. Often, however, it is some quantity (or attribute) associated with each of the elements that ultimately is of most interest. In such settings it frequently is not unreasonable to expect that this quantity be influenced in an important manner by the interactions among the elements. For example, the behaviors and beliefs of people can be strongly influenced by their social interactions; proteins that are more similar to each other, with respect to their DNA sequence information, often are responsible for the same or related functional roles in a cell; computers more easily accessible to a computer infected with a virus may in turn themselves become more quickly infected; and the relative concentration of species in an environment (e.g., animal species in a forest or chemical species in a vat) can vary over time as a result of the nature of the relationships among species.

Quantities associated with such phenomena can usefully be thought of as stochastic processes defined on network graphs. More formally, they can be represented in terms of collections of random variables, say X , indexed at least in part on a network graph $G = (V, E)$, either of the form $\{X_i\}$, for $i \in V$, or $\{X_i(t)\}$, for $i \in V$ and $t \in \mathbb{T}$, where \mathbb{T} is a (discrete or continuous) range of times. For example, the functionality of proteins can be viewed as categorical variables associated with each $i \in V$. So too can various behaviors and beliefs of individuals be represented using such variables, possibly indexed in time. Similarly, the spread of a computer virus can be captured using a set of binary variables (i.e., ‘infected’ or ‘not infected’) that evolve over

time. Finally, the relative concentration of chemicals in a reaction over time can be viewed as a time-indexed vector in \mathbb{R}^{N_v} .

We will refer to processes $\{X_i\}$ as *static processes* and $\{X_i(t)\}$ as *dynamic processes*. Given appropriate measurements, statistical problems that arise in the study of such processes include their modeling and the inference of model parameters, and also, in particular, prediction. To date, most such work arguably has been done in the context of static processes, although this situation is beginning to change. Accordingly, our presentation will concentrate largely on the case of static processes, with the development of methods and models in Sections 8.2 through 8.4, followed by a case study in Section 8.5. A brief discussion of dynamic processes appears in Section 8.6.

8.2 Nearest Neighbor Prediction

We begin by describing the problem of predicting a static process on a graph. Prediction is one of the most common tasks associated with statistical modeling and inference of such processes. We use this discussion to motivate the idea of exploiting network graph structure to predict a static process, and we then illustrate the promise of this idea with one of the most commonly used methods for this task – nearest-neighbor prediction.

Consider a collection of vertex attributes, which, as in previous chapters, we will express succinctly in vector form as $\mathbf{X} = (X_i)_{i \in V}$. Such attributes may be inherently independent of time, and hence form a truly static process, or perhaps more commonly, may constitute a ‘snapshot’ of a dynamic process in a given ‘slice’ of time (e.g., see the discussion in Section 3.6). In Chapters 6 and 7, such attributes were used in modeling and predicting the presence or absence of edges in a network graph G . That is, we modeled the behavior of the variables $\mathbf{Y} = [Y_{ij}]_{(i,j) \in V^{(2)}}$, conditional on \mathbf{X} . Alternatively, however, in some contexts it may be the behavior of \mathbf{X} , conditional on \mathbf{Y} , that is of interest instead. We illustrate through the following example.

Example 8.1 (Predicting Type of Practice Using Collaboration Among Lawyers). Recall the network of collaboration among lawyers of Section 6.5.4, shown in Figure 6.7, where an edge is present for those pairs of lawyers for which both indicated in a survey that they worked together in a substantive manner. Each of these 36 lawyers were also noted as concentrating on one of two types of legal practice – 20 work in litigation and 16 work in corporate law (shown as red and cyan vertices, respectively, in Figure 6.7). It seems plausible to suspect that lawyers collaborate more frequently with other lawyers in the same legal practice. If so, knowledge of collaboration may be useful in predicting practice.

More formally, suppose we have observed the adjacency matrix $\mathbf{Y} = \mathbf{y}$ of collaboration among these lawyers, as well as the practice of all but one of them, where practice is coded as a binary variable $X_i = 0$ if lawyer i concentrates on litigation,

		Lawyer i	
		Litigation	Corporate
Lawyer j	Litigation	29	43
	Corporate	43	43

Table 8.1 Table of counts of edges $\{i, j\}$, representing collaboration, between lawyers with one of two types of practice (i.e., litigation or corporate) for the lawyer collaboration network in Figure 6.7. (The value ‘43’ in the two off-diagonal cells counts the same edges, due to the undirectedness of the underlying network graph.)

and $X_i = 1$, if the concentration is instead on corporate. How useful would it be to know the practice of one of the neighbors of this one lawyer of unknown practice? Table 8.1 shows a breakdown of the 115 edges in the network graph, according to the practice of the pairs of lawyers corresponding to the vertices defining each edge. The two litigation lawyers involved in no collaborations have been dropped from this analysis. Looking at the rows of this table, we see that, of the collaborations involving the 18 litigation lawyers, roughly 40% are with other litigation lawyers and 60% are with corporate lawyers. Similarly, among collaborations involving corporate lawyers, the same two numbers are 50% each. These numbers suggest that, in predicting the practice of a given lawyer, knowing the practice of a single collaborator provides little predictive value.

However, examination of the diagonal of Table 8.1 indicates that of the 115 edges in the network graph, $29 + 43 = 72$ (or 63%) involve pairs of lawyers with common practice, which suggests that in aggregate the knowledge of collaborator practice is more informative than not. What if we were to simultaneously use the information on the known practice of all of the collaborators of a given lawyer? For example, for a given vertex i , in predicting X_i we might compute the fraction of all neighboring vertices with an attribute value of 1, and assign the value 1 to X_i if that fraction exceeds a certain threshold. In the case of predicting practice among collaborating lawyers, this method – not surprisingly – yields a decidedly better predictor than one based on knowledge of only a single collaborator. Figure 8.1 shows the distribution of these fractions, where each vertex is given a turn as a central vertex with ‘unknown’ practice, and vertices have been separated in two groups according to their true practice. A threshold of 0.5 would yield a predictive rule allowing us to infer 13 of the 16 corporate lawyers correctly (i.e., 81%), and 16 of the 18 litigation lawyers correctly (i.e., 89%), for an overall error rate of just under 15% (i.e., 5 misclassified out of 34). \square

The method just described is an example of a *nearest-neighbor* method. In fact, it is simply the application of nearest-neighbor smoothing on a graph, being based on quantities of the form

$$\frac{\sum_{j \in \mathcal{N}_i} x_j}{|\mathcal{N}_i|}, \quad (8.1)$$

which is just the average of the observed vertex process in the neighborhood of i . See Hastie, Tibshirani, and Friedman [194, Ch. 2.3.2], for example, for background

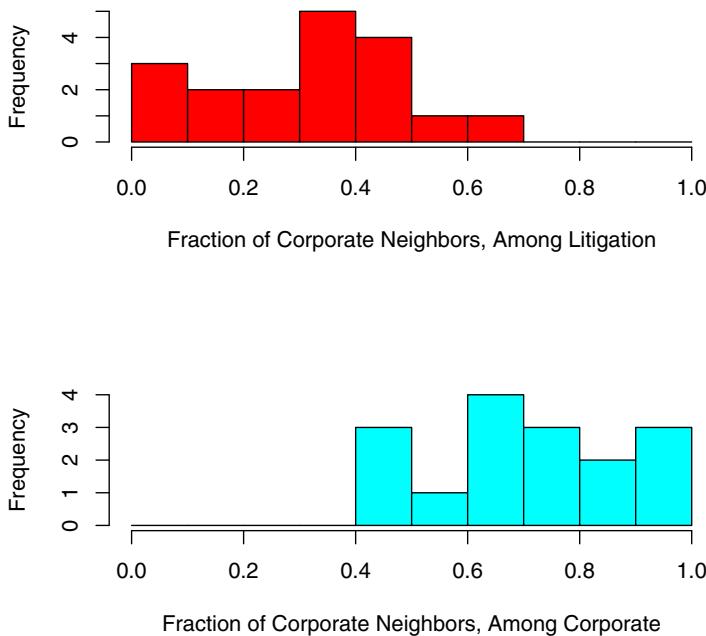


Fig. 8.1 Histograms of the fraction of corporate collaborators among lawyers in the network of Figure 6.7, separated according to the practice of each lawyer (top: litigation; bottom: corporate).

on nearest-neighbor methods. Note that the average in (8.1) is well defined not just for binary processes \mathbf{X} , but also continuous processes as well. In the binary context, such methods are also known as ‘guilt-by-association’ methods in some fields.

An important point to note in regards to nearest-neighbor methods (and, for that matter, all of the methods described in this chapter) is that their use presupposes the relevance of the network structure to the vertex attribute of interest. A poorly chosen combination of vertex attribute X and graph G cannot, of course, be expected to yield particularly useful predictions.

While nearest-neighbor methods may seem rather informal and simple, they often are found to be quite competitive with more formal and complex methods. We will see this point illustrated in the context of prediction of binary processes later in the case study of Section 8.5. Nevertheless, there are good reasons to criticize nearest-neighbor methods as well.

For example, there is no obvious mechanism by which to formally account for missing data. That is, rather than imagining that we have only one single value X_i to predict, as was tacitly assumed in Example 8.1, more generally we may suppose that we observe $\mathbf{X}^{obs} = \mathbf{x}^{obs}$ and wish to predict \mathbf{X}^{miss} , where $\mathbf{X} = (\mathbf{X}^{obs}, \mathbf{X}^{miss})$. In this

setting, for predicting a given element X_i of \mathbf{X}^{miss} , how best to evaluate (8.1) may not be clear in the case where X_j is not observed for one or more $j \in \mathcal{N}_i$. One solution to this dilemma is to redefine (8.1) in terms of only those vertices $j \in V \setminus \{i\}$ that are both adjacent to i and for which X_j is observed. Alternatively, we might impute values for the unobserved X_j , substituting, for instance, the overall average $\sum_{k \in V^{obs}} x_k / |V^{obs}|$. Although neither of these approaches is entirely satisfactory, imputation is likely to work better in practice, particularly when the relative number of missing observations is not small.

The principles underlying the nearest-neighbor method can be formalized and extended through the construction of appropriate statistical models. Such modeling can allow, for example, for probabilistically rigorous predictive statements as well as estimation and testing of model parameters. In addition, a modeling perspective facilitates a methodical approach to the inclusion of network structure beyond the immediate neighborhood of vertices. It can also facilitate the handling of missing data. In the following sections we will discuss two different approaches to modeling attributes \mathbf{X} in terms of the structure of a network graph G , the first being probabilistic and the second more akin to regression.

8.3 Markov Random Fields

We introduce the class of Markov random field models for vertex attributes on network graphs and discuss statistical issues relating to inference and prediction with these models. We also briefly describe a handful of related models.

8.3.1 Markov Random Field Models

Let $G = (V, E)$ be a graph and $\mathbf{X} = (X_1, \dots, X_{N_v})^T$ be a collection of discrete random variables defined on V . We say that \mathbf{X} is a *Markov random field* (MRF) on G if

$$\mathbb{P}(\mathbf{X} = \mathbf{x}) > 0, \quad \text{for all possible outcomes } \mathbf{x}, \quad (8.2)$$

and

$$\mathbb{P}(X_i = x_i | \mathbf{X}_{(-i)} = \mathbf{x}_{(-i)}) = \mathbb{P}(X_i = x_i | \mathbf{X}_{\mathcal{N}_i} = \mathbf{x}_{\mathcal{N}_i}), \quad (8.3)$$

where $\mathbf{X}_{(-i)}$ is the vector $(X_1, \dots, X_{i-1}, X_{i+1}, \dots, X_{N_v})^T$ and $\mathbf{X}_{\mathcal{N}_i}$ is the vector of all X_j for $j \in \mathcal{N}_i$. The positivity assumed in (8.2) for the joint distribution of \mathbf{X} is a useful technical condition. The expression in (8.3) is a Markov condition, asserting that X_i is conditionally independent of all other X_k , given the values of its neighbors, where the neighborhood structure is determined by G .

The concept of an MRF can be seen as a generalization of a Markov chain (i.e., compare (8.3) to (2.8) in Chapter 2), and has its roots in statistical mechanics, going back to the work of Ising [211] on ferromagnetic fields. MRFs are used extensively

in spatial statistics (e.g., see Cressie [102, Ch. 6.4]) and in image analysis (e.g., see Li [258]).

A key feature facilitating the practical usage of Markov random fields is their equivalence, under appropriate conditions, with *Gibbs random fields* i.e., random vectors \mathbf{X} with distributions of the form

$$\mathbb{P}(\mathbf{X} = \mathbf{x}) = \left(\frac{1}{\kappa} \right) \exp \{U(\mathbf{x})\} . \quad (8.4)$$

Here $U(\cdot)$ is called the *energy function* and

$$\kappa = \sum_{\mathbf{x}} \exp \{U(\mathbf{x})\} , \quad (8.5)$$

the *partition function*. The equivalence of MRFs and Gibbs random fields is established by the Hammersley-Clifford theorem, a proof of which may be found in Besag [36].

Importantly, the energy function can be decomposed as a sum over cliques in G , in the form

$$U(\mathbf{x}) = \sum_{c \in \mathcal{C}} U_c(\mathbf{x}) , \quad (8.6)$$

where \mathcal{C} denotes the set of all cliques of all sizes in G , and a clique of size 1 consists of just a single vertex $v \in V$. The functions $U_c(\cdot)$ are called *clique potentials*. Specification of the clique potentials therefore provides a natural way in which to specify an MRF,¹ in that the conditional probability in (8.3) can be shown to take the form

$$\begin{aligned} \mathbb{P}(X_i = x_i | \mathbf{X}_{(-i)} = \mathbf{x}_{(-i)}) &= \frac{\mathbb{P}(\mathbf{x})}{\sum_{\mathbf{x}' : \mathbf{x}'_{(-i)} = \mathbf{x}_{(-i)}} \mathbb{P}(\mathbf{x}')} \\ &= \frac{\exp \left\{ \sum_{c \in \mathcal{C}_i} U_c(\mathbf{x}) \right\}}{\sum_{\mathbf{x}' : \mathbf{x}'_{(-i)} = \mathbf{x}_{(-i)}} \exp \left\{ \sum_{c \in \mathcal{C}_i} U_c(\mathbf{x}') \right\}} , \end{aligned} \quad (8.7)$$

where \mathcal{C}_i is the set of all cliques involving vertex i , which indicates that the Markov condition in (8.3) indeed holds.

In this abstract form, MRF models can involve extremely complicated expressions, a fact which, while arguably an indication of their richness on the one hand, can adversely impact both interpretability and computations. In practice, these models often are simplified by assumptions of homogeneity, in the sense that the form of the clique potentials U_c is assumed not to depend on the particular positions of the cliques $c \in \mathcal{C}$. Furthermore, usually cliques of only a limited size are defined to have non-zero partition functions U_c , which reduces the complexity of the decom-

¹ Note that MRFs are not uniquely defined by a given set of clique potentials, and additional constraints are needed. For example, it is common to impose on each U_c the condition that it equal zero when a particular element of \mathbf{x} is zero. See Cressie [102, Ch. 6.4.1], for example.

position in (8.6). This later step has direct implications on the nature of the assumed dependency in \mathbf{X} . We illustrate with an example.

Example 8.2 (Auto-Logistic MRFs). Besag [36] suggested introducing the additional conditions that (i) only cliques $c \in \mathcal{C}$ of size one or two have non-zero potential functions U_c , and (ii) the conditional probabilities in (8.3) have an exponential family form (i.e., a form like that in (6.23)). The first condition is sometimes referred to as ‘pairwise-only dependence.’ Under these conditions, the energy function takes the form

$$U(\mathbf{x}) = \sum_{i \in V} x_i H_i(x_i) + \sum_{\{i,j\} \in E} \beta_{ij} x_i x_j , \quad (8.8)$$

for some functions $H_i(\cdot)$ and coefficients $\{\beta_{ij}\}$. Besag called the class of Markov random field models with energy functions like that in (8.8) *auto-models*.

Now suppose that the X_i are binary random variables (i.e., taking on just the values zero and one). Under appropriate normalization conditions, the functions H_i can be made to only contribute to the expansion of $U(\mathbf{x})$ in (8.6) in a non-trivial fashion when $x_i = 1$, in which case (8.8) is equivalent in form to

$$U(\mathbf{x}) = \sum_{i \in V} \alpha_i x_i + \sum_{\{i,j\} \in E} \beta_{ij} x_i x_j , \quad (8.9)$$

for certain parameters $\{\alpha_i\}$. The resulting MRF model is called an *auto-logistic* model, because the conditional probabilities in (8.3) have the form

$$\mathbb{P}(X_i = 1 | \mathbf{X}_{\mathcal{N}_i} = \mathbf{x}_{\mathcal{N}_i}) = \frac{\exp(\alpha_i + \sum_{j \in \mathcal{N}_i} \beta_{ij} x_j)}{1 + \exp(\alpha_i + \sum_{j \in \mathcal{N}_i} \beta_{ij} x_j)} , \quad (8.10)$$

indicating logistic regression of x_i on its neighboring x_j 's. The Ising model in statistical mechanics is a particular case of this model, with G defined to be a regular lattice.

Assumptions of homogeneity can further simplify this model. For example, specifying that $\alpha_i \equiv \alpha$ and $\beta_{ij} \equiv \beta$, the probability in (8.10) reduces to

$$\mathbb{P}(X_i = 1 | \mathbf{X}_{\mathcal{N}_i} = \mathbf{x}_{\mathcal{N}_i}) = \frac{\exp(\alpha + \beta \sum_{j \in \mathcal{N}_i} x_j)}{1 + \exp(\alpha + \beta \sum_{j \in \mathcal{N}_i} x_j)} . \quad (8.11)$$

This model can be read as dictating that the logarithm of the conditional odds that $X_i = 1$ scales linearly in the number of neighbors j of i with the value $X_j = 1$,

$$\log \frac{\mathbb{P}(X_i = 1 | \mathbf{X}_{\mathcal{N}_i} = \mathbf{x}_{\mathcal{N}_i})}{\mathbb{P}(X_i = 0 | \mathbf{X}_{\mathcal{N}_i} = \mathbf{x}_{\mathcal{N}_i})} = \alpha + \beta_1 \sum_{j \in \mathcal{N}_i} x_j . \quad (8.12)$$

Similarly, the log-odds can be made to scale linearly as well in the number of neighbors j of i with the value $X_j = 0$,

$$\log \frac{\mathbb{P}(X_i = 1 | \mathbf{X}_{\mathcal{N}_i} = \mathbf{x}_{\mathcal{N}_i})}{\mathbb{P}(X_i = 0 | \mathbf{X}_{\mathcal{N}_i} = \mathbf{x}_{\mathcal{N}_i})} = \alpha + \beta_1 \sum_{j \in \mathcal{N}_i} x_j + \beta_2 \sum_{j \in \mathcal{N}_i} (1 - x_j) , \quad (8.13)$$

through the parameter choice $\alpha_i = \alpha + |\mathcal{N}_i|\beta_2$ and $\beta_{ij} = \beta_1 - \beta_2$. Other variations follow similarly. \square

The auto-logistic model has been extended to the case where the X_i take on values $\{0, 1, \dots, m\}$, for arbitrary positive integer m , yielding a class of models called *multi-level logistic* or *multi-color* models. See Strauss [371]. Other auto-models of interest include the auto-binomial, the auto-Poisson, and the auto-Gaussian.

For the auto-Gaussian and other MRF models involving continuous random variables, the probability mass functions and summations in our definition of MRFs must be replaced by probability density functions and integrals, but the essence of the MRF framework otherwise carries through unchanged. The conditional distributions in (8.3) under the auto-Gaussian are univariate Gaussian, with conditional expectations of the form

$$\mathbb{E}(X_i | \mathbf{X}_{\mathcal{N}_i} = \mathbf{x}_{\mathcal{N}_i}) = \alpha_i + \sum_{j \in \mathcal{N}_i} \beta_{ij} (x_j - \alpha_j) , \quad (8.14)$$

and conditional variances $\mathbb{V}(X_i | \mathbf{X}_{\mathcal{N}_i} = \mathbf{x}_{\mathcal{N}_i}) = \sigma^2$. Hence the value of each X_i behaves, conditionally, like a weighted combination of the values of its neighbors. Under the conditions that $\beta_{ii} = 0$ and $\beta_{ij} = \beta_{ji}$, the joint distribution of \mathbf{X} is multivariate Gaussian, with mean vector $\mu = (\alpha_i)$ and covariance matrix $\Sigma = \sigma^2(\mathbf{I} - \mathbf{B})^{-1}$, where $\mathbf{B} = [\beta_{ij}]$. Homogeneity assumptions may of course be imposed, which can lead to simplified expressions. For example, a common model in the context of spatial statistics assumes that $\alpha_i = \alpha$ and $\beta_{ij} = \beta$, in which case the mean vector μ is determined by a single scalar and the covariance matrix Σ takes the form $\sigma^2(\mathbf{I} - \beta \mathbf{A})^{-1}$, where \mathbf{A} is the adjacency matrix of the underlying graph G . See Rue and Held [335] for details on such Gaussian Markov random fields and related models.

Note that an MRF model may be used alone, as developed above, or to specify just a component(s) of a larger, more complex model. For example, in the modeling of texture in image analysis it is common to use MRFs as prior distributions for image structure, and then, conditional on the image structure, to model measurements as a corruption of that image by additive noise. See the seminal article by Geman and Geman [167], for instance, or Li [258].

8.3.2 Inference and Prediction for Markov Random Fields

As noted earlier, a task of particular interest in the context of static processes $\mathbf{X} = (X_i)_{i \in V}$ on network graphs is the prediction of some or all of \mathbf{X} . We have seen a number of MRF models commonly used in practice in this context that can be expressed in the form

$$\mathbb{P}_\theta(\mathbf{X} = \mathbf{x}) = \left(\frac{1}{\kappa(\theta)} \right) \exp \{U(\mathbf{x}; \theta)\} , \quad (8.15)$$

where θ is a relatively low-dimensional parameter vector. Predictions can be generated based on $\mathbb{P}_\theta(\cdot)$. However, knowledge of θ is necessary, and in general θ is unknown. If measurements of the process \mathbf{X} are available, we can try to infer θ from the data.

We first describe methods of inference for MRFs, and then proceed to discuss the task of prediction.

8.3.2.1 Inference for MRFs

In principle, the task of inferring θ in (8.15) is most naturally approached through the method of maximum likelihood. But in practice this method often proves to be intractable. Specifically, although the log-likelihood can be expressed in the simple form

$$\log \mathbb{P}_\theta(\mathbf{X} = \mathbf{x}) = U(\mathbf{x}; \theta) - \log \kappa(\theta) , \quad (8.16)$$

in order to evaluate it we must be able to compute the partition function $\kappa(\theta)$, which in turn requires summation over all possible values of \mathbf{x} , as indicated in (8.5). Unfortunately, it will be computationally prohibitive to perform this summation in all but the smallest of problems.

The method of maximum pseudo-likelihood, originally proposed by Besag [37] for the analysis of spatial data, is a popular alternative to maximum likelihood in this context. Specifically, instead of optimizing (8.16), we instead seek to maximize

$$\sum_{i \in V} \log \mathbb{P}_\theta(X_i = x_i | \mathbf{X}_{\mathcal{N}_i} = \mathbf{x}_{\mathcal{N}_i}) , \quad (8.17)$$

which is the logarithm of the product of the conditional probabilities of each observed x_i , given the values of its neighbors. By (8.7) we see that these probabilities do not involve the partition function $\kappa(\theta)$, but rather are functions only of the clique potentials $U_c(\mathbf{x}; \theta)$, which presumably have been specified explicitly. As a result, maximum pseudo-likelihood estimation is often a much more tractable computational problem. On the other hand, it can produce estimates that differ substantially from the maximum likelihood estimates when the dependencies inherent in the full joint distribution are too substantial to be ignored.

Example 8.3 (Inference in the Auto-Logistic Model). Consider the homogeneous auto-logistic model in equation (8.11) of Example 8.2. The maximum likelihood estimate (MLE) of (α, β) is defined as

$$\left(\hat{\alpha}, \hat{\beta} \right)_{MLE} = \arg \max_{\alpha, \beta} [\alpha M_1(\mathbf{x}) + \beta M_{11}(\mathbf{x}) - \kappa(\alpha, \beta)] , \quad (8.18)$$

where $M_1(\mathbf{x})$ is the number of vertices with the attribute value 1 and $M_{11}(\mathbf{x})$ is the number of pairs of vertices where both have the attribute value 1. Calculation of

$\kappa(\alpha, \beta)$ is prohibitive, as it requires evaluation of M_1 and M_{11} across all 2^{N_v} binary vectors \mathbf{x} of length N_v , with respect to the network graph G .

In contrast, the maximum pseudo-likelihood estimate (MPLE) is

$$\left(\hat{\alpha}, \hat{\beta} \right)_{MPLE} = \arg \max_{\alpha, \beta} \left\{ \alpha M_1(\mathbf{x}) + \beta M_{11}(\mathbf{x}) - \log \left[1 + \exp \left(\alpha + \beta \sum_{j \in \mathcal{N}_i} x_j \right) \right] \right\}. \quad (8.19)$$

While the solution to this optimization does not have a closed-form expression, the second component, replacing $\kappa(\alpha, \beta)$ in (8.18) can now be computed easily, and the overall estimate can be computed using standard software for logistic regression, with the N_v pairs $(x_i, \sum_{j \in \mathcal{N}_i} x_j)$ serving as response and predictor, respectively.

For auto-logistic models with other parameterizations, such as that underlying (8.13), analogous formulas hold for the MLE and MPLE. Note that in general, in the absence of multiple observations \mathbf{x} of the assumed MRF, some sort of homogeneity assumption will be necessary to, at the very least, make the model identifiable. \square

Other methods of inference include the coding method of Besag [36] and the mean-field method from statistical mechanics. In the coding method, a number of disjoint subsets of vertices are formed, say S_1, \dots, S_K , where each subset S_k consists of vertices for which no pair are neighbors. The log-likelihood of $\{X_i : i \in S_k\}$ is then given exactly by the expression in (8.17), but with V replaced by S_k in the summation. As a result, optimization of this expression produces an MLE, say $\hat{\theta}^{(k)}$, albeit on a reduced subset of the data. An average of estimates $\hat{\theta}^{(k)}$ over k is often used as the final estimate. In the mean-field method, the observed value $\mathbf{x}_{\mathcal{N}_i}$ in (8.3) is replaced by its expected value $\mathbb{E}(\mathbf{X}_{\mathcal{N}_i})$ in all equations, which effectively leads to an approximation of the partition function $\kappa(\theta)$ as a product of local partition functions. This strategy sets up an iterative method for calculation of the likelihood $\mathbb{P}_\theta(\mathbf{X} = \mathbf{x})$, which may then be maximized numerically to yield an approximation of the MLE. See Li [258] and references therein.

One MRF model for which the partition function $\kappa(\theta)$ is not a deterrent to estimation is the auto-Gaussian model, where the joint distribution of \mathbf{X} is known in closed form. In this case, maximum likelihood optimization can be computationally tractable. For example, consider the homogeneous model described earlier, where the mean is a constant α across the network and the covariance is written $\sigma^2(\mathbf{I} - \beta \mathbf{A})^{-1}$. The unknown parameter vector $\theta = (\alpha, \sigma^2, \beta)$ can be estimated using a numerical approach, alternately maximizing the likelihood as a function of (α, σ^2) , given β , and of β , given (α, σ^2) . The first set of estimates can be found in closed form, while the second estimates can be obtained using a Newton-Raphson algorithm. Care needs to be taken to ensure that the estimate of β leaves $\mathbf{I} - \beta \mathbf{A}$ positive definite. See Cressie [102, Ch. 7.2.2].

8.3.2.2 Prediction for MRFs

Now consider the problem of predicting some or all of the process \mathbf{X} , given a value for the parameter θ . Predictions naturally derive from the distribution $\mathbb{P}_\theta(\cdot)$ in (8.15) but, as noted in the previous section, explicit evaluation of this distribution generally is prohibitive. However, it is relatively straightforward to simulate from this distribution using the Gibbs sampler, a type of Markov chain Monte Carlo algorithm.

The Gibbs sampler is an iterative algorithm that allows us to exploit the fact that the univariate conditional distributions $\mathbb{P}_\theta(X_i | \mathbf{X}_{(-i)} = \mathbf{x}_{(-i)})$ are generally available in relatively simple and closed form. Specifically, initializing the algorithm with a random value, say $\mathbf{X}^{(0)} = \mathbf{x}^{(0)}$, a new value $\mathbf{X}^{(m)}$ is generated from $\mathbf{X}^{(m-1)} = \mathbf{x}^{(m-1)}$ by drawing

$$\begin{aligned} X_1^{(m)} &\quad \text{from} \quad \mathbb{P}_\theta\left(X_1 | \mathbf{X}_{(-1)} = \mathbf{x}_{(-1)}^{(m-1)}\right) \\ X_2^{(m)} &\quad \text{from} \quad \mathbb{P}_\theta\left(X_2 | \mathbf{X}_{(-2)} = \mathbf{x}_{(-2)}^{(m-1)}\right) \\ &\quad \vdots & \quad \vdots \\ X_{N_v}^{(m)} &\quad \text{from} \quad \mathbb{P}_\theta\left(X_{N_v} | \mathbf{X}_{(-N_v)} = \mathbf{x}_{(-N_v)}^{(m-1)}\right) , \end{aligned} \quad (8.20)$$

where $\mathbf{x}_{(-i)}^{(m-1)} = (x_1^{(m-1)}, \dots, x_{i-1}^{(m-1)}, x_{i+1}^{(m-1)}, \dots, x_{N_v}^{(m-1)})$. The sequence $\mathbf{X}^{(1)}, \mathbf{X}^{(2)}, \mathbf{X}^{(3)}, \dots$ clearly forms a Markov chain. Under appropriate conditions, this chain has a stationary distribution and that distribution is equal to $\mathbb{P}_\theta(\cdot)$. See Liu [264] or Robert and Casella [325], for example, for details on Gibbs sampling and a discussion of issues relating to convergence.

Given the ability to sample from $\mathbb{P}_\theta(\cdot)$, we can predict \mathbf{X} in whole or in part using the appropriate empirical distributions, based on a sufficiently large sample. For example, we can predict individual elements X_i of \mathbf{X} using the corresponding empirical marginal distribution. In the case of binary \mathbf{X} , we might predict $X_i = 1$ if the empirical marginal frequency of this event is greater than some threshold. Similarly, we also can study higher-level aggregate properties of the process. For example, in the binary case, using simulation we can easily gain insight into questions regarding, say, the frequency at which vertices i of value $X_i = 1$ are found in a given subset of vertices in the network graph G .

If we observe only some of the elements of \mathbf{X} , say $\mathbf{X}^{obs} = \mathbf{x}^{obs}$, and we wish to predict the remaining elements \mathbf{X}^{miss} , the relevant distribution is

$$\mathbb{P}_\theta(\mathbf{X}^{miss} | \mathbf{X}^{obs} = \mathbf{x}^{obs}) . \quad (8.21)$$

A modified version of the Gibbs sampler described above may be used to sample from this distribution, where elements of \mathbf{X}^{miss} are drawn at each iteration using the conditional distributions

$$\mathbb{P}\left(X_i | \mathbf{X}^{obs} = \mathbf{x}^{obs}, \mathbf{X}^{miss} = \mathbf{x}_{(-i)}^{(m-1),miss}\right) . \quad (8.22)$$

Here $\mathbf{X}_{(-i)}^{\text{miss}}$ denotes all of \mathbf{X}^{miss} except X_i , for $i \in V^{\text{miss}}$, and $\mathbf{x}_{(-i)}^{(m-1),\text{miss}}$ is the vector of values sampled for $\mathbf{X}_{(-i)}^{\text{miss}}$ at the $(m - 1)$ -th iteration.

Note that in this context, if the parameter θ is unknown, we have only the values $\mathbf{X}^{\text{obs}} = \mathbf{x}^{\text{obs}}$ with which to estimate it, whereas our discussion of inference in Section 8.3.2.1 implicitly assumed that all of \mathbf{X} was available. So our inference procedures must be modified. For pseudo-likelihood, the common practice appears to be to simply ignore the missing data and to perform the relevant computations using neighborhoods redefined in terms of just those neighbors with observed values. This approach is equivalent to applying pseudo-likelihood to the subgraph G^* obtained through induced subgraph sampling (i.e., as described in Section 5.3.1) with sampled vertex set $V^* = V^{\text{obs}}$. Issues regarding the mechanism of missingness, analogous to those encountered with the link prediction problem in Section 7.2, are clearly relevant here, but do not seem to have been explored in any depth to date.

Finally, we note that when the MRF \mathbf{X} is not itself directly observed, but instead there is a noisy version, inference becomes more complicated. In this situation it usually is of interest to predict all of \mathbf{X} , but again, to do so the parameters θ typically must be estimated as well. Geman and Geman [167] propose a now-standard Bayesian framework for inference in this type in the context of image analysis, including a Monte Carlo scheme for calculating the relevant posterior of \mathbf{X} given the data. See Cressie [102, Ch. 7.4.3], for example, for a summary, as well as discussion of additional methods for the same problem.

8.3.3 Related Probabilistic Models

There are other probabilistic models for vertex attribute processes that relate to MRF models to varying extents. We describe two of them briefly.

Robins, Pattison, and Elliott [331] introduce a class of models in the social network literature designed to generalize the exponential random graph models (ERGMs) of Chapter 6.5 for the purpose of modeling social influence processes. Structurally their models bear strong resemblance to MRF models, but they differ from what we have described in a handful of important details. First, the models are expressed explicitly in terms of \mathbf{Y} , the adjacency matrix of the network graph G as a random matrix, and particular attention is paid to questions of dependency among individual Y_{ij} and Y_{kl} and their attributes X_i, X_j, X_k , and X_l . Second, their derivation is for directed graphs. Lastly, they describe a version of their framework for multivariate attributes. Parameter estimation is done using pseudo-likelihood methods.

Of a somewhat different nature are the auto-probit models of Weir and Pettitt [397], introduced for modeling spatial processes on a regular lattice. These models, proposed as an alternative to the auto-logistic model, are based on an extension of the concept of a probit model.² Here the binary process \mathbf{X} is no longer modeled

² For a binary variable Y and covariate vector \mathbf{X} , a probit model specifies that the Y_i are conditionally independent, given the \mathbf{X}_i , and that $\mathbb{P}(Y_i = 1 | \mathbf{X}_i = \mathbf{x}_i) = \Phi(\mathbf{x}_i^T \beta)$, for some vector β of

explicitly, as in the case of the auto-logistic model, but rather is defined implicitly through a latent process, say \mathbf{Z} , of continuous random variables. Specifically, the process \mathbf{Z} is modeled as a homogeneous auto-Gaussian MRF with $\sigma^2 = 1$ and, given $\mathbf{Z} = \mathbf{z}$, for each $i \in V$, $X_i = 1$ if $z_i \geq 0$, and $X_i = 0$ if $z_i < 0$. As a result, we have that

$$\begin{aligned}\mathbb{P}(X_i = 1) &= \Phi\left(\frac{\alpha}{[(\mathbf{I} - \beta\mathbf{A})^{-1}]_{ii}^{1/2}}\right) \\ &= \mathbb{E}\left[\Phi\left(\alpha + \beta \sum_{j \in \mathcal{N}_i} z_j\right) \mid \mathbf{Z}_{\mathcal{N}_i} = \mathbf{z}_{\mathcal{N}_i}\right],\end{aligned}\quad (8.23)$$

where \mathbf{A} is the adjacency matrix of the underlying graph G , and $\Phi(\cdot)$ is the CDF of a Gaussian random variable with mean 0 and variance 1. These models inherit the advantage of the auto-Gaussian model of not having to deal with an intractable partition function $\kappa(\theta)$. Weir and Pettitt [397] demonstrate that it is relatively straightforward to infer unknown parameters using Bayesian methods and standard MCMC algorithms, and that the biases sometimes encountered using pseudo-likelihood seem to be avoided. See also Weir and Pettitt [398], for an extended application of these models.

8.4 Kernel-based Regression

The probabilistic models we have just seen postulate a precise form for the dependency structure among vertex attributes X_i , with respect to the topology of the underlying graph G . But in some contexts, such as when prediction of unobserved vertex attributes is the only goal, it may be felt sufficient simply to ‘learn’ from the data a function relating the vertices to their attributes. The nearest-neighbor methods discussed in Section 8.2 in principle yield such a function, albeit implicitly. For a more explicit construction, a regression-based approach i.e., essentially regression on the graph G , would seem appealing. However, standard methods of regression, such as classical least squares regression, being set up as they are for relating response and predictor variables in Euclidean space, are not immediately applicable to graph-indexed data.

Kernel methods have been found to be useful for extending the classical regression paradigm to various settings with non-traditional data. At the most basic level, these methods consist of (i) a generalized notion of predictor variables (i.e., encoded in a so-called ‘kernel’), and (ii) regression of a response on these generalized predictors using an extension of ridge regression (i.e., see Chapter 2.2.3.2). In this section we first discuss the basic kernel approach to regression modeling, in the context of

parameters, where $\Phi(\cdot)$ is the CDF of a Gaussian random variable with mean zero and variance one. See McCullagh and Nelder [272], for example.

graphs specifically, and we then present a number of methods for constructing the necessary kernels on graphs.

8.4.1 Kernel Regression on Graphs

Let $G = (V, E)$ be a network graph and $\mathbf{X} = (X_1, \dots, X_{N_v})$ a vertex attribute process. Suppose that we observe $X_i = x_i$, for $i \in V^{obs} \subseteq V$, and let $n = |V^{obs}|$ be the number of vertices at which we have observations. We will take as our goal to learn from the data an appropriate function, say \hat{h} , mapping from V to \mathbb{R} , that describes well the manner in which attributes vary across the vertices. Such a function may be of interest simply as a summary of the relationship among observed pairs (i, x_i) , for $i \in V^{obs}$, or additionally as a tool to be used in predicting for unobserved pairs (i, X_i) , for $i \in V^{miss} = V \setminus V^{obs}$.

A key challenge in any regression problem is to decide within which class \mathcal{H} of functions h to restrict the search for \hat{h} . Certainly it is desirable to use a class that is rich enough to capture the expected relationships underlying the data, while at the same time, not so rich that it cannot be searched in a computationally efficient manner. For our problem of regression on a network graph, there is the additional issue that our data do not lie in a Euclidean space, as is traditionally the case. Kernel regression methods arguably lend themselves well to our needs, being (i) quite general in their basic formulation – applicable even in rather non-traditional contexts like ours, and (ii) flexible enough to be tuned to specific problems. Furthermore, they typically can be implemented using established methods of optimization.

At the heart of kernel methods is the notion of a kernel function. A function K , mapping from $V \times V$ to \mathbb{R} , is called a (*positive semi-definite*) *kernel* if, for each $m = 1, \dots, N_v$ and subset of vertices $\{i_1, \dots, i_m\} \subseteq V$, the $m \times m$ matrix $\mathbf{K}^{(m)} = [K(i_j, i_{j'})]$ is symmetric and positive semi-definite.³ Broadly speaking, kernels can be thought of as functions that produce similarity matrices. The predictor variables used in the kernel regression are in turn derived from these similarity matrices. In the present context of kernel regression on network graphs, the kernel describes the similarity among vertices in the underlying graph G . Since G itself often is defined to represent such similarities, it is common to construct kernels that summarize the topology of G . We defer until later, in Section 8.4.2, discussion of the construction and computation of kernels on graphs, concentrating first on laying out the essential elements of kernel regression.

To begin, we note that since V is a finite set, real-valued functions $h(\cdot)$ on V can equivalently be represented as vectors $\mathbf{h} \in \mathbb{R}^{N_v}$, with the i -th element of \mathbf{h} associated with the vertex $i \in V$. Our presentation therefore will be entirely in terms of such vectors. Given the observations \mathbf{x}^{obs} , the graph G , and a kernel K , kernel regression seeks to find an optimal choice of \mathbf{h} within the class

$$\mathcal{H}_K = \left\{ \mathbf{h} \in \mathbb{R}^{N_v} : \mathbf{h} = \Phi \beta \text{ and } \beta^T \Delta^{-1} \beta < \infty \right\}, \quad (8.24)$$

³ A matrix $\mathbf{M} \in \mathbb{R}^m$ is positive semi-definite if $\mathbf{x}^T \mathbf{M} \mathbf{x} \geq 0$, for all $\mathbf{x} \in \mathbb{R}^m$.

where Φ and $\Delta = \text{diag}[(\delta_j)]$ are $N_v \times N_v$ orthogonal and diagonal matrices, respectively, arising through the eigen-decomposition $\mathbf{K} = \Phi \Delta \Phi^T$. Here $\mathbf{K} = \mathbf{K}^{(N_v)}$ is the $N_v \times N_v$ matrix formed by evaluating K at all pairs $(i, j) \in V \times V$. The space \mathcal{H}_K is called a *reproducing kernel Hilbert space* (RKHS). Its members are all vectors that can be represented as linear combinations of the eigenvectors in Φ , with the condition that the weight vectors β satisfy the constraint that $\beta^T \Delta^{-1} \beta$ be finite.⁴ Formally, this latter term corresponds to a norm on the space \mathcal{H}_K , and so the constraint is in fact simply that all elements of \mathcal{H}_K have finite norm.⁵

In order to choose an appropriate element \mathbf{h} in \mathcal{H}_K , say $\hat{\mathbf{h}}$, a penalized regression strategy is employed in kernel regression in an effort to enforce that $\hat{\mathbf{h}}$ both be close to the observed data and have small norm. Specifically, an estimate $\hat{\mathbf{h}} = \Phi \hat{\beta}$ is produced by an optimization of the form

$$\min_{\beta} \left[\sum_{i \in V^{obs}} C(x_i; (\Phi \beta)_i) + \lambda \beta^T \Delta^{-1} \beta \right], \quad (8.25)$$

where $C(\cdot, \cdot)$ is a convex function that measures the loss incurred through predicting its first argument by its second, $(\Phi \beta)_i$ denotes that element of $\Phi \beta$ corresponding to $i \in V^{obs}$, and λ is a tuning parameter.⁶

The optimization in (8.25) is a type of complexity-penalized estimation strategy, generalizing that of the ridge regression defined in expression (2.37) of Chapter 2. The role of the predictor variable is played by the columns of the matrix Φ (i.e., the eigenvectors of the kernel matrix \mathbf{K}), and that of the response variable, by the elements in \mathbf{x}^{obs} . The loss captured by $C(\cdot, \cdot)$ encourages goodness of fit in the model, while the term $\beta^T \Delta^{-1} \beta$ penalizes excessive complexity, in the sense that eigenvectors with small eigenvalues are penalized more harshly than those with large eigenvalues. Note that the penalty term is essentially the logarithm of a multivariate Gaussian distribution, with mean zero and diagonal covariance Δ . So the penalty is in the spirit of an unnormalized Gaussian prior. The parameter λ dictates the relative importance of the loss versus the complexity penalty.

In order to find $\hat{\mathbf{h}}$, it would seem that we need to calculate the entire matrix Φ , and thus evaluate the kernel K at all pairs of vertices across the entire graph G , which in practice may be prohibitively expensive. But the celebrated Representer theorem of Kimeldorf and Wahba [227], which states that the solution $\hat{\mathbf{h}}$ will be of the form $\mathbf{h} = \mathbf{K}^{(N_v, n)} \alpha$, where $\mathbf{K}^{(N_v, n)}$ is the $N_v \times n$ matrix resulting from evaluation

⁴ Here, and throughout this section, we assume that Δ^{-1} exists or, if not, that the pseudo-inverse is used.

⁵ A reproducing kernel Hilbert space may be defined much more generally, as a Hilbert space \mathcal{H} of functions $h : \mathcal{X} \rightarrow \mathbb{R}$ on an arbitrary input space \mathcal{X} , satisfying the properties that (a) \mathcal{H} is effectively generated by all linear combinations of the elements $K(x, \cdot)$, viewed as functions of their second argument, for fixed $x \in \mathcal{X}$, and (b) $\langle h, K(x, \cdot) \rangle = h(x)$, for all $h \in \mathcal{H}$. This latter property is the so-called *reproducing property*.

⁶ Generally, in practice, functions of the form $\beta_0 + \Phi \beta$ are used, rather than $\Phi \beta$ alone, where $\beta_0 \in \mathbb{R}$ – sometimes called an ‘offset’ term – plays a role analogous to that of an intercept term in a linear regression. We omit this detail here and throughout, for ease of exposition.

of K on $V \times V^{obs}$, coupled with the so-called reproducing property of K (i.e., see Footnote 5), dictates that we may equivalently solve the optimization

$$\min_{\alpha} \left[\sum_{i \in V^{obs}} C(x_i; (\mathbf{K}^{(n)} \alpha)_i) + \lambda \alpha^T \mathbf{K}^{(n)} \alpha \right]. \quad (8.26)$$

Here $\mathbf{K}^{(n)}$ is the $n \times n$ matrix resulting from evaluation of K on $V^{obs} \times V^{obs}$, which often will be much smaller than \mathbf{K} . As a result, the size of the optimization problem itself will be similarly reduced.

For certain choices of loss, the optimization in (8.26) has a closed-form solution. In general, however, numerical methods must be used. We illustrate with two examples.

Example 8.4 (Kernel Ridge Regression). Suppose that the attribute random variables X_i are continuous, and let the loss $C(x; a) = (x - a)^2$ be the squared difference of its arguments. Then the overall loss is the residual sum of squares from prediction of the vector \mathbf{x}^{obs} by the vector $\mathbf{K}^{(n)} \alpha$, and the optimization defining $\hat{\alpha}$ takes the form

$$\min_{\alpha} \left[(\mathbf{x}^{obs} - \mathbf{K}^{(n)} \alpha)^T (\mathbf{x}^{obs} - \mathbf{K}^{(n)} \alpha) + \lambda \alpha^T \mathbf{K}^{(n)} \alpha \right]. \quad (8.27)$$

This particular method of kernel regression is called *kernel ridge regression*. To understand the connection with ridge regression, define $\Phi^{(n)}$ and $\Delta^{(n)}$ through the eigen-decomposition $\mathbf{K}^{(n)} = \Phi^{(n)} \Delta^{(n)} \Phi^{(n)T}$ and let $\beta^{(n)} = \Delta^{(n)} \Phi^{(n)T} \alpha$. Then (8.27) can be re-expressed as

$$\min_{\beta^{(n)}} \left[(\mathbf{x}^{obs} - \Phi^{(n)} \beta^{(n)})^T (\mathbf{x}^{obs} - \Phi^{(n)} \beta^{(n)}) + \lambda \beta^{(n)T} \Delta^{(n)-1} \beta^{(n)} \right], \quad (8.28)$$

which resembles (8.25), but with all relevant quantities now defined only with respect to the sample on V^{obs} . If we then rescale, introducing the matrix $\mathbf{M} = \Phi^{(n)} \Delta^{(n)1/2}$ and the parameter vector $\theta = \Delta^{(n)-1/2} \beta^{(n)}$, the optimization in (8.28) becomes simply

$$\min_{\theta} \left[(\mathbf{x}^{obs} - \mathbf{M} \theta)^T (\mathbf{x}^{obs} - \mathbf{M} \theta) + \lambda \theta^T \theta \right], \quad (8.29)$$

which is the form of a standard ridge regression of \mathbf{x}^{obs} on \mathbf{M} in the parameter θ .

From (2.38) we know that the solution to (8.29) is

$$\hat{\theta} = (\mathbf{M}^T \mathbf{M} + \lambda \mathbf{I})^{-1} \mathbf{M}^T \mathbf{x}^{obs}. \quad (8.30)$$

The solution $\hat{\alpha}$ to (8.27) then follows through the expression $\hat{\alpha} = \Phi^{(n)} \Delta^{(n)-1/2} \hat{\theta}$. In turn, we may then calculate the kernel regression function $\hat{h}(\cdot)$ in the form of the vector $\hat{\mathbf{h}} = \mathbf{K}^{(N_v, n)} \hat{\alpha}$. \square

Example 8.5 (Kernel Logistic Regression). Suppose that the attribute random variables X_i indicate class membership. In the kernel regression literature, for the two-class case, class membership typically is encoded through the values $X = -1$ or $X = +1$. If we now specify the loss function as

$$C(x; a) = \ln(1 + e^{-xa}) , \quad (8.31)$$

this choice corresponds to the negative log-likelihood of a Bernoulli random variable, and hence a logistic loss.

Kernel logistic regression selects $\hat{\alpha}$ through the optimization

$$\min_{\alpha} \left[\sum_{i \in V^{obs}} \left(1 + e^{-x_i (\mathbf{K}^{(n)} \alpha)_i} \right) + \lambda \alpha^T \mathbf{K}^{(n)} \alpha \right] . \quad (8.32)$$

The solution to this optimization does not have a closed-form expression. However, an iteratively re-weighted least-squares algorithm, derived from application of the Newton-Raphson method, may be applied, similar to that used in the case of standard logistic regression. Details may be found in Zhu and Hastie [416, Sec. 3], for example.

Given the value $\hat{\alpha}$, we again obtain the kernel regression function $\hat{h}(\cdot)$ in the form of the vector $\hat{\mathbf{h}} = \mathbf{K}^{(N_v, n)} \hat{\alpha}$. Prediction of a new X_i , or one of the missing $X_i, i \in V^{miss}$, can be based upon the probability

$$\widehat{\mathbb{P}}(X_i = 1 | \mathbf{X}^{obs} = \mathbf{x}^{obs}) = \frac{e^{\hat{h}_i}}{1 + e^{\hat{h}_i}} . \quad (8.33)$$

Compare this expression to that in (7.7), for example. \square

There are many other versions of kernel regression in the literature. Kernel logistic regression, for example, may be compared to the related *support vector machine* (SVM) methods popular in the machine learning literature. For instance, the hard-margin SVM replaces the logistic loss function in (8.31) by the ‘hinge loss’ $C(x; a) = [1 - xa]_+$, where $[.]_+$ is the function that returns its argument, if positive, or else zero, if not positive. The corresponding optimization problem (8.25) cannot be solved in closed-form, and neither can the Newton-Raphson method be brought to bear, due to the discontinuity in the derivative of the loss function at zero. But the problem can be reformulated as a quadratic programming problem and solved using standard software for this purpose. Predictions in this case take the form $\text{sign}[\hat{h}_i]$. See Schölkopf and Smola [342, Ch. 1.5] for additional details, and Zhu and Hastie [416] for a more involved discussion comparing kernel logistic and SVM methods.

In all such kernel regression methods, there is of course the choice of the tuning parameter λ to be made. Typically this problem is approached as one in which we seek to find a value for λ that minimizes the expected cost of the model $\hat{\mathbf{h}}_\lambda = \mathbf{K}^{(N_v, n)} \hat{\alpha}_\lambda$, where $\hat{\alpha} = \hat{\alpha}_\lambda$ solves the optimization in (8.26) for fixed λ . Cross-validation is commonly employed for this purpose, fitting the model on a series of k training sets and minimizing a cost based on evaluation of the fitted models on the

corresponding test sets (i.e., as described in Footnote 2 of Chapter 7). In some cases, depending on the form of the model, certain analytical approximations may be used to simplify calculations. See Wahba [389], for example, or also Gu [187].

8.4.2 Designing Kernels on Graphs

As described so far, there is actually nothing inherent to our description of kernel regression that is specific to regression on graphs *per se*. Rather, with appropriate modification of notation, our discussion holds equally well for kernel regression of arbitrary random variables $\{Z_i\}$ on any finite collection of elements $s_i \in \mathcal{S}$. The connection with modeling vertex attribute processes on network graphs is complete only when the kernel function K is designed specifically with respect to a network graph G , with $\mathcal{S} = V$.

In designing a kernel on a graph G , we seek a matrix $\mathbf{K} = \mathbf{K}^{(N_v)}$ that is at the very least symmetric and positive semi-definite. In addition, however, recall that the kernel function essentially quantifies the similarity among its arguments. Thus, a kernel on a network graph G should be designed to capture suspected similarity relationships among vertices in V , as they pertain to the attribute process \mathbf{X} . Based on the presumption that proximity of vertices in G is already indicative of some notion of similarity, kernels proposed in this area typically derive their form largely through the topology of G . Importantly, therefore, it should be recognized from the start that if we construct a network graph G in which vertex proximity is *not* indicative of their similarity on characteristics *relevant* to the vertex attribute process, then just as when there is little association between a response variable and a predictor in classical regression, we can expect to gain little from our kernel regression.

Although vertex proximity is naturally encoded in the adjacency matrix \mathbf{A} , it is more common to see the graph Laplacian used in this context.⁷ Recall from Section 2.1.3 that the graph Laplacian is defined as $\mathbf{L} = \mathbf{D} - \mathbf{A}$, where $\mathbf{D} = \text{diag}[(d_v)]$. The *Laplacian kernel* is defined simply as the (pseudo)inverse of the Laplacian (i.e., $\mathbf{K} = \mathbf{L}^-$). It is not difficult to show that \mathbf{L}^- is symmetric and positive semi-definite, and hence a proper kernel matrix. Various authors have examined its use. See Belkin, Matveeva, and Niyogi [29], for example, who propose the use of this kernel for kernel ridge regression and establish certain theoretical properties of the resulting method.

Note that under this choice of kernel, the penalty term in (8.25) has the form

$$\begin{aligned}\beta^T \Delta^{-1} \beta &= \beta^T \Phi^T \Phi \Delta^{-1} \Phi^T \Phi \beta \\ &= \mathbf{h}^T \mathbf{K}^- \mathbf{h} \\ &= \mathbf{h}^T \mathbf{L} \mathbf{h} .\end{aligned}\tag{8.34}$$

Therefore, since

⁷ We could equally well use the normalized graph Laplacian in the discussion that follows.

$$\mathbf{h}^T \mathbf{L} \mathbf{h} = \sum_{\{i,j\} \in E} (h_i - h_j)^2 , \quad (8.35)$$

we see that use of this kernel will encourage the kernel regression to seek vectors $\mathbf{h} = \Phi \beta$ that are locally ‘smooth’ with respect to the topology of G . That is, vertices adjacent in G will be assigned values in \mathbf{h} that are more likely to be close, so as to reduce the differences in (8.35).

The basic Laplacian kernel $\mathbf{K} = \mathbf{L}^-$ encodes similarity among vertices through the adjacency matrix \mathbf{A} . We can also conceive of encoding similarity through higher-order topological characteristics of G , such as its path structure. Recall, for example, that the entries of the r -th power of \mathbf{A} represent the number of walks of length r between pairs of vertices. Note too that any even power of a symmetric matrix \mathbf{M} is positive semi-definite. So powers of \mathbf{A} – or, as used in practice, of \mathbf{L} – can also define kernel matrices \mathbf{K} .

A popular choice, incorporating all such powers of \mathbf{L} , is the *diffusion kernel*

$$\mathbf{K} = \sum_{m=0}^{\infty} \frac{(-\zeta)^m}{m!} \mathbf{L}^m , \quad (8.36)$$

which is typically expressed as $\mathbf{K} = \exp(-\zeta \mathbf{L})$, where $\exp(\cdot)$ here denotes matrix exponentiation, rather than component-wise exponentiation. The value $\zeta > 0$ is a decay factor, used to control the level of similarity assigned to vertices separated by paths of successively greater lengths. This kernel takes its name from the fact that we may write $\frac{d}{d\zeta} \mathbf{K} = -\mathbf{L} \mathbf{K}$, where \mathbf{K} is treated as a function of ζ . This expression parallels the heat equation in physics, which governs the diffusion of heat through a continuous medium. See Kondor and Lafferty [234].

Computation of kernels deriving from powers of \mathbf{L} is facilitated by the fact that, for any symmetric matrix \mathbf{M} , the matrix \mathbf{M}^m shares the same eigenvectors as \mathbf{M} , and its eigenvalues are just those of \mathbf{M} raised to the m -th power. So, following the notation in (8.34), and writing $\mathbf{L} = \Phi \Gamma \Phi^T$, where $\Gamma = \text{diag}[(\gamma_i)] = \text{diag}[(\delta_i^{-1})] = \Delta^{-1}$ contains the eigenvalues of \mathbf{L} , sorted in non-decreasing order, we have $\mathbf{L}^m = \Phi \Gamma^m \Phi^T$. Hence, $\mathbf{K} = \mathbf{L}^{-m} = \Phi \Delta^m \Phi^T$. In the case of the diffusion kernel (8.36), a similar expression holds, but with the eigenvalues δ_i^m replaced by $\exp(-\zeta \gamma_i)$.

In fact, the Laplacian and diffusion kernels are simply two examples of a more general class of kernels introduced by Smola and Kondor [356], based on a certain notion of ‘regularization.’ In this formulation, the kernel matrix has the form

$$\mathbf{K} = \sum_{i=1}^{N_v} r^{-1}(\gamma_i) \phi_i \phi_i^T , \quad (8.37)$$

where ϕ_i is the i -th column of Φ and $r(\cdot)$ is a real-valued, non-negative, and increasing function. In other words, \mathbf{K} is defined as the inverse of a regularized Laplacian $r(\mathbf{L}) = \Phi r(\Gamma) \Phi^T$, where $r(\Gamma)$ indicates application of the function $r(\cdot)$ to each of the diagonal entries γ_i of Γ . Smola and Kondor study choices of r expressible as power series, which include the identity function $r(\gamma) = \gamma$, the exponential function $r(\gamma) = \exp(\zeta \gamma)$, and the linear inverse function $r(\gamma) = (1 - \zeta \gamma)^{-1}$.

Note that for all of these various choices of Laplacian-based kernels, the eigenvectors of \mathbf{K} remain unchanged, and only the eigenvalues are modified. Therefore, the effect on the kernel regression in (8.25) of modifying the kernel within this class is not to change the predictor variables (i.e., the columns of Φ), which remain the same, but rather to alter the form of the penalty term $\beta^T \Delta^{-1} \beta$, by changing the values of the weights from Δ^{-1} to $r(\Delta^{-1})$. The following example illustrates the role of the eigenvalues and eigenvectors further.

Example 8.6 (Kernels for the Lawyer Collaboration Network). Recall the network of collaboration among lawyers, as shown in Figure 6.7. This network has 36 lawyers, of which 34 form a connected component. We will restrict our attention to just this connected component, and examine the composition of a handful of Laplacian-based kernels on this sub-graph.

In Figure 8.2 is shown a plot of the eigenvalues $\gamma_1, \dots, \gamma_{34}$ of the Laplacian \mathbf{L} , sorted from smallest to largest. Recall that the smallest eigenvalue γ_1 is zero, by virtue of the definition of \mathbf{L} . Also shown in Figure 8.2 are values for $r^{-1}(\gamma_i)$, $i = 2, \dots, 34$, for $r(\cdot)$ the identity, exponential, and inverse linear functions, where the coefficient ζ in the latter two functions is set to $\zeta = 3$ and $\zeta = \gamma_{34}^{-1}$, respectively. For the first two choices of $r(\cdot)$, the effects of most of the larger eigenvalues are damped, and therefore only the eigenvectors corresponding to the first few smallest eigenvalues play a substantial role in the definition of \mathbf{K} . For the last choice of $r(\cdot)$ (i.e., the linear inverse function), the decay in $r^{-1}(\cdot)$ is comparatively much more gradual, and pretty much all of the eigenvectors play a non-trivial role.

A visual representation of the eigenvectors ϕ_i in Φ , for $i = 2, \dots, 10$, is shown in Figure 8.3. The network is presented using a simple circular layout. Each vertex is plotted with an area proportional to the magnitude of the component of ϕ_i at

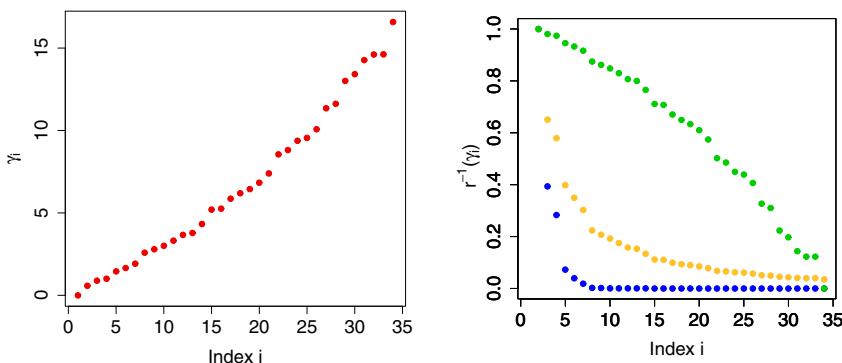


Fig. 8.2 Left: Eigenvalues γ_i of the Laplacian \mathbf{L} for the network of collaboration among lawyers. Right: Regularized eigenvalues $r^{-1}(\gamma_i)$, using the identity (yellow), exponential (blue), and linear inverse (green) functions $r(\cdot)$. (Note: Regularized eigenvalues have been normalized to facilitate display.)

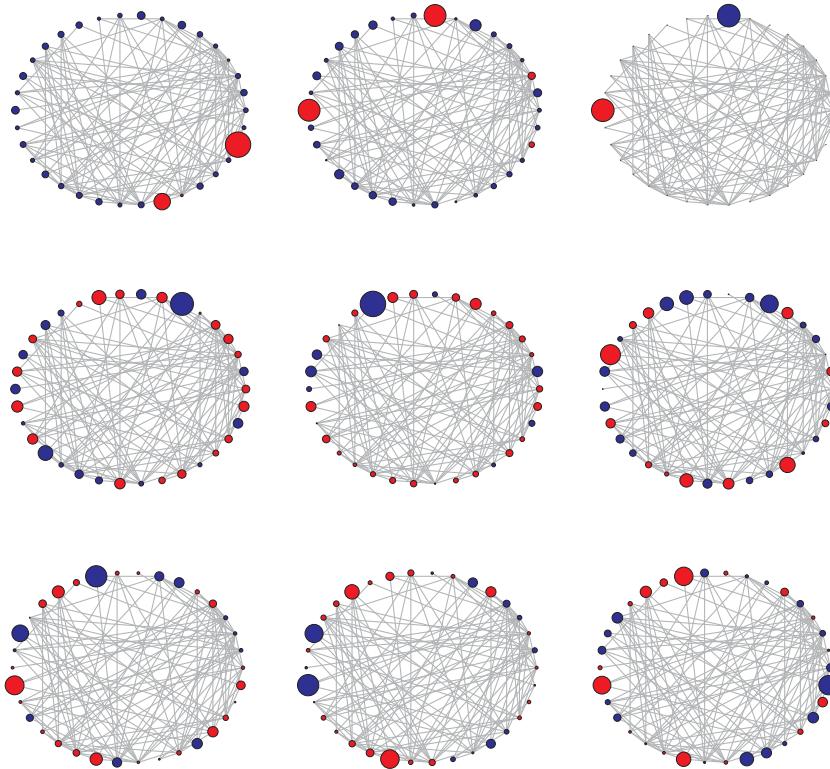


Fig. 8.3 Visual representation of the eigenvectors ϕ_i corresponding to the first nine (non-trivial) smallest eigenvalues γ_i of L in the network of collaboration among lawyers. Top row: $i = 2, 3, 4$; Middle row: $i = 5, 6, 7$; Bottom row: $i = 8, 9, 10$.

that particular vertex, with positive values indicated by red, and negative, by blue. Edges are shown in gray. We can see that the early eigenvectors, corresponding to the smallest eigenvalues, have entries that are somewhat more uniform in size (other than a few outstanding exceptions) and color than in the later eigenvectors, corresponding to larger eigenvalues. In other words, the eigenvectors can be seen to become less ‘smooth’ with increasing eigenvalue. \square

Note that the kernel regression framework dictates that there be a single kernel K specified. In practice, however, we may have a number of kernels of potential interest to us, say K_1, \dots, K_L , in which case we must somehow either select one or combine them.

For comparing kernels, Cristianini, Shawe-Taylor, and colleagues [103, 104] have introduced a notion of correlation between a kernel and the observed value $\mathbf{X} = \mathbf{x}$ of a vertex attribute. For two kernel matrices, say \mathbf{K}_1 and \mathbf{K}_2 , the *alignment* between the kernels is defined as

$$a(\mathbf{K}_1, \mathbf{K}_2) = \frac{\langle \mathbf{K}_1, \mathbf{K}_2 \rangle}{\sqrt{\langle \mathbf{K}_1, \mathbf{K}_1 \rangle \langle \mathbf{K}_2, \mathbf{K}_2 \rangle}} , \quad (8.38)$$

where $\langle \cdot, \cdot \rangle$ here denotes the inner product of its matrix arguments when the matrices are each ‘stretched out’ as a single vector.⁸ Since the outer product $\mathbf{x}\mathbf{x}^T$ of the observed vertex attributes is symmetric and positive semi-definite, it constitutes a (trivial) kernel matrix. The alignment between a kernel \mathbf{K} and a ‘target’ \mathbf{x} is thus defined as $a(\mathbf{K}, \mathbf{x}\mathbf{x}^T)$. This value may be computed and compared across various choices of kernel.

For example, with lawyer practice serving as our vertex attribute process, as in Example 8.1, the kernels in Example 8.6 above, based on the choice of identity, exponential, and inverse linear regularization functions, have kernel-target alignments of 0.0848, 0.0272, and 0.1299, respectively. These numbers suggest that the inverse linear kernel is a better match than the other two. Cristianini, Kandola, Elisseeff, and Shawe-Taylor [104] provide analytical arguments and numerical examples supporting the notion that larger kernel-target alignment should yield more accurate prediction.

If we instead desire to combine multiple kernels, we can think of the problem as one of determining an appropriate set of weights $\{\omega_1, \dots, \omega_p\}$, among all sets $\{\omega_l\}_{l=1}^p$ such that $\omega_l \geq 0$ and $\sum_{l=1}^p \omega_l = 1$, and forming the quantity $K = \sum_{l=1}^p \omega_l K_l$. That is, we can define K to be a convex combination of our p kernels, which can be shown to be itself a kernel. In the case that $\omega_l = 1$, for some l , and $\omega_{l'} = 0$, for all other $l' \neq l$, we simply have $K = K_l$. Methods proposed for selecting these weights in a data-dependent manner include incorporating them as an additional set of parameters in the optimization in (8.25), as in the work of Lin and Zhang [262, 413] and the work of Lanckriet et al. [246], or optimizing the kernel-target alignment (e.g., Cristianini, Shawe-Taylor, and colleagues [103, 104]). Other work of this nature includes Argyriou, Herbster, and Pontil [14], Zhu, Kandola, Ghahramani, and Lafferty [417], and Zhu, Kandola, Lafferty, and Ghahramani [418]. For a Bayesian approach addressing this problem, see Liang et al. [259].

8.5 Case Study: Predicting Protein Function

Proteins are fundamental to the complex molecular and biochemical processes taking place within organisms. An understanding of their role – or ‘function’ – is therefore critical in biology and bio-related areas, for purposes ranging from general knowledge to the development of targeted medicine and diagnostics. High-throughput sequencing technology has identified a tremendous number of genes that code for proteins with no known functional annotation. In fact, it has been conjectured that on average as many as 70% of the genes in a genome code for proteins with poorly known or unknown functions (see Murali, Wu, and Kasif [286]). Es-

⁸ More formally, $\langle \mathbf{M}_1, \mathbf{M}_2 \rangle = \text{tr}(\mathbf{M}_1^T \mathbf{M}_2)$ is the *Frobenius inner product* of the matrices $\mathbf{M}_1, \mathbf{M}_2$, where $\text{tr}(\cdot)$ denotes the trace of its argument.

tablishing the functionality of these proteins is therefore an important problem in functional genomics and, given the vast number of proteins to be annotated, it is natural to approach this problem from the perspective of statistical prediction.

Originally, work in this area built predictions based on known or measured characteristics of each individual protein. But with the advent of large-scale relational data for proteins, derived from high-throughput technologies, approaches incorporating network-based information have become standard. For example, there is an established tendency for proteins that are nearer to each other in a network graph G of protein interactions, such as that described in Section 4.2.1.1, to possess more similar functional roles. This observation suggests the use of statistical methods of prediction that, roughly speaking, are more likely to predict a certain function for a given protein when more proteins in a neighborhood of that protein have that same or similar functions.

Initial attempts to implement this strategy were based on simple nearest-neighbor voting rules among the immediate neighbors of a protein in G , like those described in Section 8.2. More sophisticated approaches have since been proposed based on methodologies like those discussed above in Sections 8.3 and 8.4. In this case study we illustrate the three classes of methods we have encountered in this chapter (i.e., nearest-neighbor, Markov random fields, and kernels), through a cross-validation analysis of their performance in a protein function prediction task. For a survey of other related methods used in this area, see the review by Sharan, Ulitsky, and Shamir [347].

We use data on Baker's yeast – formally known as *Saccharomyces cerevisiae*, a popular model organism for development and testing of both biological and methodological techniques. This organism is unicellular and much is known about its biology already. A common vocabulary used in describing the function of proteins in yeast (as well as other organisms) is the Gene Ontology (GO),⁹ an attempt to standardize and organize terminology used in this area. GO terms are listed under three general categories – *biological process*, *cellular component*, and *molecular function* – and arranged in a hierarchical fashion within each category, moving from general terms to more specific terms. Here we will consider the prediction of the functional annotation *intracellular signaling cascade* (ICSC), a sub-category of *biological process*, defined as “a series of reactions within the cell that occur as a result of a single trigger reaction or compound.” Proteins participating in ICSCs are important in that they are involved in signal transduction, which influences how cells react to their environment, both at the level of responses to simple stimuli like heat and more complex stimuli like disease.

The network graph G we will use comes from the network of yeast protein interactions described in Section 4.2.1.1. Recall that the original network graph consists of 5,151 proteins and 31,301 edges. For the purposes of illustration, we restrict our attention to only those proteins annotated in the January 2007 GO database with the term *cell communication*, and the interactions among these. *Cell communication* is defined as “any process that mediates interactions between a cell and its surround-

⁹ <http://www.geneontology.org/>.

ings” and is a term more general than ICSC (i.e., an ancestor of ICSC in the GO hierarchy) but more specific than *biological process* (i.e., a descendant of *biological process* in the GO hierarchy). The resulting network graph is shown Figure 8.4, and consists of 134 vertices (proteins), after deleting isolated vertices, and 241 edges (protein interactions). There are four components to the graph, one of order 127, one of order 3, and two of order 2. Vertices have been colored to indicate whether they are (yellow) or are not (blue) annotated with ICSC in the GO database. We can see that there is a good deal of homogeneity to the vertex labels, with neighbors frequently sharing the same color. This observation suggests that prediction of ICSC on this network graph should be reasonably feasible.

Let \mathbf{X} denote the vertex process corresponding to the annotation ICSC. That is, $X_i = 1$ if the protein corresponding to vertex i has the ICSC annotation, and $X_i = 0$, otherwise. Three methods were compared for predicting the annotation ICSC. These methods were implemented on the largest connected component, whose 127 vertices consisted of 70 labeled with the annotation ICSC and 51 without. Predictions were generated using ten-fold cross-validation, in which 90% of the labels were used to

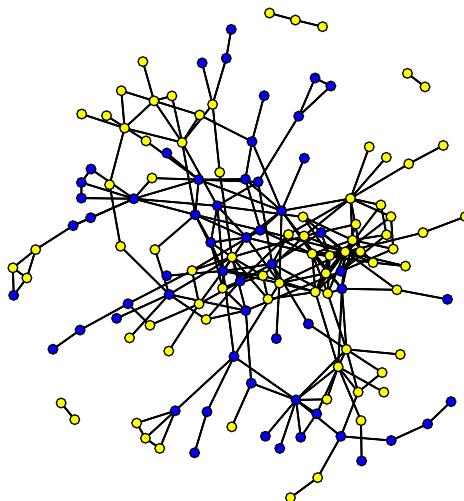


Fig. 8.4 Network of interactions among proteins known to be responsible for cell communication in yeast. Yellow vertices denote proteins that are known to be involved in intracellular signaling cascades, a specific form of communication in the cell. The remaining proteins are indicated in blue.

train the prediction methods and the remaining 10% were used to test the resulting predictors. Vertices were assigned randomly to the ten folds.

The first method was simply the nearest-neighbor method. For each vertex in a given test set, if the fraction of neighbors in the training set annotated with ICSC was greater than a given threshold t , that vertex was also assigned the ICSC annotation. If there were no annotated neighbors, a fraction of 0.5 was assigned.

The second method was based on a Markov random field model, with two predictors, counting the number of neighbors with and without the ICSC annotation, as in (8.13). The parameters $(\alpha, \beta_1, \beta_2)$ were estimated using pseudo-likelihood, based on the subgraph defined by the vertices in the training set and the edges among them. Given these estimates, 1,000 samples of annotations for vertices in the test set were generated using the Gibbs sampling methodology described in Section 8.3.2.2. A burn-in of 100 iterations was used, with a lag of ten iterations between samples. Predictions were then produced by comparing to a threshold t the probabilities $\mathbb{P}(X_i = 1 | \mathbf{X}^{obs} = \mathbf{x}^{obs})$ that a vertex had the label 1, where the probabilities were approximated by the corresponding relative frequencies over the 1,000 samples.

The last method was based on kernel-logistic regression, using the implementation¹⁰ of Zhu and Hastie [416]. The kernel \mathbf{K} was chosen to be the inverse Laplacian (i.e., \mathbf{L}^-), and the tuning parameter λ was, after some experimentation, set to $\lambda = 0.01$. Predictions were obtained by comparing to a threshold t the probabilities calculated according to (8.33).

Figure 8.5 shows receiver operating characteristic (ROC) curves for each of the three methods, as the threshold t is varied from zero to one, based on an average over the ten cross-validation folds. All three methods can be seen to perform comparably. The area under the curve (AUC) for the nearest neighbor, Markov random field, and kernel methods, respectively, is approximately 0.80, 0.82, and 0.83, while the misclassification rate at a threshold value of $t = 0.5$ is approximately 0.23, 0.23, and 0.24. These numbers confirm our earlier conjecture, based on the network graph in Figure 8.4, that the interaction among proteins is reasonably predictive of ICSC annotation.

The fact that these methods performed so similarly is not entirely surprising. For example, the nearest neighbor and the MRF methods, under the model in (8.13), use essentially the same two statistics, $\sum_{j \in \mathcal{N}_i} x_j$ and $\sum_{j \in \mathcal{N}_i} (1 - x_j)$. Furthermore, the nearest neighbor algorithm, in a certain form, has been shown to be equivalent to a particular form of graph partitioning (Blum and Chawla [40]), and the graph Laplacian \mathbf{L} – which underlies our choice of kernel in our kernel logistic regression – plays a central role in many methods for partitioning graphs. In addition, it has been frequently noted that relatively simple methods, like nearest-neighbors and naive Bayes, often perform quite comparably to more sophisticated methods in classification tasks. See Friedman [160], for example, for background and discussion of this phenomenon.

However, it should also be noted that the MRF and kernel frameworks have the distinct advantage over the nearest-neighbor method of being able to easily incorpo-

¹⁰ Software available at <http://www.stat.lsa.umich.edu/jizhu/code/bivm/>.

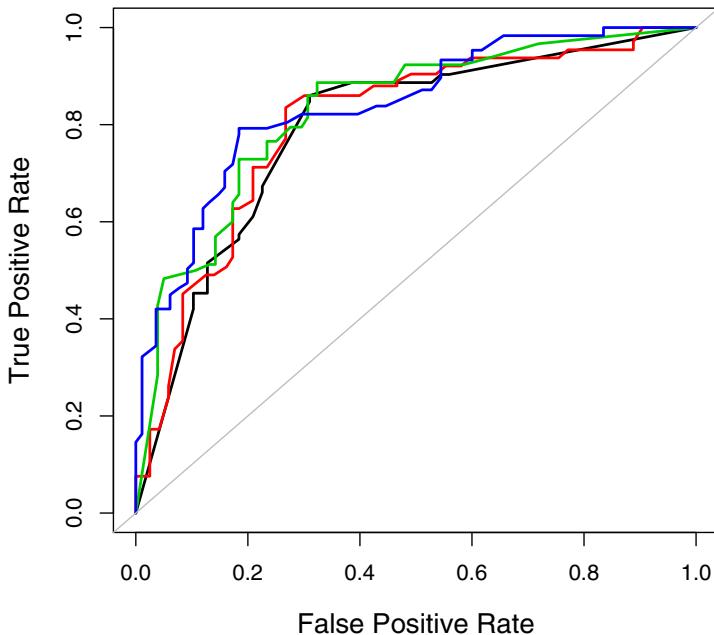


Fig. 8.5 Receiver operating characteristic (ROC) curves summarizing the accuracy of the nearest neighbor (black), Markov random field (red), and kernel logistic regression (green) methods, for predicting proteins involved in intracellular signaling cascades (ICSC) based on protein interactions. Also shown (blue) is the ROC curve for prediction with a kernel logistic regression using a kernel combining information on protein interactions and sequence motifs.

rate additional information – of both a relational and non-relational nature – beyond that contained in the network graph G . Recall, for example, that with kernel-based methods, such information can be incorporated through the combination of multiple kernels. In predicting protein function it is known that the presence of certain DNA sequence motifs can be useful. Among the 127 proteins in our analysis, 114 of them are known to be associated with one or more of 154 different motifs. We can encode this information in a 127×154 binary matrix, say \mathbf{M} , and then construct a positive semi-definite kernel \mathbf{MM}^T through simple inner-products. Re-running our kernel logistic regression, with kernel $\mathbf{K} = \omega \mathbf{L}^- + (1 - \omega) \mathbf{MM}^T$, for $\omega = 0.5$ and $\lambda = 0.1$, yields some small degree of improvement. Figure 8.5 shows the corresponding ROC curve, which has an AUC of approximately 0.85. The misclassification rate for a threshold of $t = 0.5$ is approximately 0.20.

For more sophisticated approaches to integrating multiple sources of information, in the context of protein function prediction, see the work of Deng, Chen, and

Sun [116] for an extension of the MRF framework, and the work of Lanckriet, Christianini, Jordan, and Nobel [247] for an extension of the kernel regression framework. Nariai, Kolaczyk, and Kasif [287] also offer an extension in a similar spirit, based on a localized variant of the MRF framework and naive Bayes principles.

8.6 Modeling and Prediction for Dynamic Processes

As we have already remarked elsewhere in this book, many of the systems studied from a network-based perspective are intrinsically dynamic in nature. Not surprisingly, therefore, many processes defined on networks are more accurately thought of as dynamic, rather than static, processes. Examples include a cascade of failures (e.g., as an electrical power-grid strains under a heat-wave), the diffusion of knowledge (e.g., as a rumor spreads in a population), the search for information (e.g., as an Internet-based search engine formulates a response to a query), the spread of disease (e.g., as a virus propagates through a population of humans or computers), the synchronization of behavior (e.g., as neurons fire in the brain), and the interaction of ‘species’ in an environment (e.g., as genes in a cell auto-regulate among themselves).

Conceptually, we may think of processes like these as time-indexed vertex attribute processes $\mathbf{X}(t) = (X_i(t))_{i \in V}$, for $t \in \mathbb{T}$, where \mathbb{T} is a range of times. Both deterministic and stochastic perspectives are commonly adopted for modeling such processes. Deterministic models are based on difference and differential equations, whereas stochastic models are based on time-indexed stochastic processes – usually Markov processes.¹¹ Contributions in this area come from across the sciences, including biology, engineering, epidemiology, neuro-science, physics, and sociology.

In this section, we will not attempt to survey the field of dynamical processes on networks as a whole, a field that has been tremendously active in recent years and already is quite large. For such a survey, we refer the reader to the book by Barrat, Barthélemy, and Vespignani [20]. Instead, we will focus here simply on presenting an extended illustration in the context of epidemic processes. This material will then be followed by a brief discussion of related work on other types of processes. Besides the issue of the sheer magnitude of contributions in this area, there are other reasons for our brevity on the topic. First, the technical level demanded by this material is somewhat more substantial than that required throughout the rest of this book, and therefore beyond our scope. Second, while there has now been a fair amount of work – most of it in just the past ten years – on the mathematics associated with models of dynamic processes on network graphs, there has been comparatively much less work on the statistics. Indeed, although there are certainly notable statistical contributions that have been made in this area, on specific classes of models for specific types of processes, there has yet to emerge a concentrated

¹¹ Informally speaking, a *Markov process* is a stochastic process in which the future of the process at any given point in time depends only upon its present state and not its past. The discrete-time Markov chain model, as described in Section 2.2.1.1, is a canonical example of a Markov process.

body of results of this nature. Third, it is still relatively uncommon in most fields interested in dynamic processes on network graphs that time-indexed data of sufficient quality may be collected in sufficient quantity to make meaningful statistical inference and prediction possible, although this situation is beginning to change.

8.6.1 Epidemic Processes: An Illustration

The term *epidemic* refers to a phenomenon that is prevalent in excess to what might be expected. It is most commonly used in the context of diseases and their dissemination throughout a population – such as with malaria, bubonic plague, and AIDS – but it is also at times used more broadly in other contexts, such as in describing the spread of perceived problems in a society. The quantitative study of epidemics goes back roughly 300 years, although the modern mathematical treatment of epidemic models arguably goes back no more than a century, to the seminal work of Kermack and McKendrick [225]. See Daley and Gani [106], for example, for an introduction to this area, or Anderson and May [10], for a more in-depth treatment. In more recent years, epidemic modeling has been an area of intense interest among researchers working on network-based dynamic process models.

Epidemic modeling is concerned with three primary issues: (i) understanding the mechanisms by which epidemics spread, (ii) predicting the future course of epidemics, and (iii) achieving an ability to control the spread of epidemics. Statistics has an important role to play in this context. Below we provide a brief overview of results for a traditional epidemic model, followed by a similar characterization of some of the analogous results that have emerged in the literature on network-based extensions.

8.6.1.1 Traditional Epidemic Modeling

The most commonly used class of continuous-time epidemic models is the class of *susceptible-infected-removed* (SIR) models. In this section, we will focus on the stochastic formulation of what is arguably the simplest member of this class – the so-called *general epidemic model*. There is no network explicit in this model. Rather, a (closed) population of, say, $N + 1$ elements is imagined such that, at any point in time t , there are some random number $N_S(t)$ of elements susceptible to infection (called ‘susceptibles’), $N_I(t)$ elements infected (called ‘infectives’), and $N_R(t)$ elements recovered and immune (or, alternatively, removed). Starting with one infective and N susceptibles, that is, with $N_I(0) = 1$ and $N_S(0) = N$, and letting s and i generically denote some numbers of susceptibles and infectives, respectively, it is assumed that the triple $(N_S(t), N_I(t), N_R(t))$ evolves according to the instantaneous transition probabilities

$$\mathbb{P}(N_S(t + \delta t) = s - 1, N_I(t + \delta t) = i + 1 | N_S(t) = s, N_I(t) = i) \approx \beta s i \delta t \quad (8.39)$$

$$\begin{aligned}\mathbb{P}(N_S(t + \delta t) = s, N_I(t + \delta t) = i - 1 | N_S(t) = s, N_I(t) = i) &\approx \gamma i \delta t \\ \mathbb{P}(N_S(t + \delta t) = s, N_I(t + \delta t) = i | N_S(t) = s, N_I(t) = i) &\approx 1 - (\beta s + \gamma) i \delta t\end{aligned},$$

where δt refers to the usual infinitesimal and the role of $N_R(t)$ is omitted due to the constraint $N_S(t) + N_I(t) + N_R(t) = N + 1$.

The above model states that, at any given time t , a new infective will emerge from among the susceptibles (due to contact with and infection by one of the infectives) with instantaneous probability proportional to the product of the number of susceptibles s and the number of infectives i . Similarly, infectives recover with instantaneous probability proportional to i . These probabilities are scaled by the parameters β and γ , usually referred to as the infection and recovery rates, respectively. The product form for the probability with which infectives emerge corresponds to an assumption of ‘homogeneous mixing’ (or ‘mass-action,’ in chemistry) among members of the population, which asserts that the population is (i) homogeneous and (ii) well mixed, in the sense that any pair of members are equally likely to interact with each other.

The bivariate stochastic process $(N_S(t), N_I(t))$ forms a continuous-time Markov chain.¹² An equivalent formulation of the model states that, given s susceptibles and i infectives at time t , the process remains in the state (s, i) for an amount of time distributed as an exponential random variable, with rate $(\beta s + \gamma)i$. A transition then occurs, which will be either to the state $(s - 1, i + 1)$, with probability $\beta s / [(\beta s + \gamma)i]$, or to the state $(s, i - 1)$, with probability $\gamma i / [(\beta s + \gamma)i]$. This formulation of the model enables easy Monte Carlo simulation of individual realizations of the process. Figure 8.6 shows a schematic characterization of the typical behavior of our stochastic SIR process under simulation.¹³ Starting with a population composed almost entirely of susceptibles and just a few infectives, we see an initial exponential increase in the number of infectives and a corresponding decrease in the number of susceptibles. This initial period is followed by a peak in the number of infectives, after which this number decays exponentially, as the supply of susceptibles is depleted and the infectives recover. Individual realizations of our stochastic SIR process obtained through Monte Carlo demonstrate natural variations about such trends (e.g., see Figure 8.7).

Corresponding to the expressions in (8.39), in analogy to the matrix of transition probabilities for discrete-time Markov chains, is a matrix of functions (the transition kernel) of the form

$$P_{s,i}(t) = \mathbb{P}(N_S(t) = s, N_I(t) = i | N_S(0) = N, N_I(0) = 1) . \quad (8.40)$$

In principle, everything we might wish to predict about an epidemic under our SIR model is contained in these probabilities. It is known that they must satisfy the

¹² Background on continuous-time Markov chains may be found in any standard text on stochastic processes. See Ross [333], for example, for an accessible introduction to the topic.

¹³ More precisely, these curves are generated from the corresponding deterministic version of our SIR model, which corresponds to the expected behavior of our stochastic SIR model in the appropriately defined limit of an infinite population.

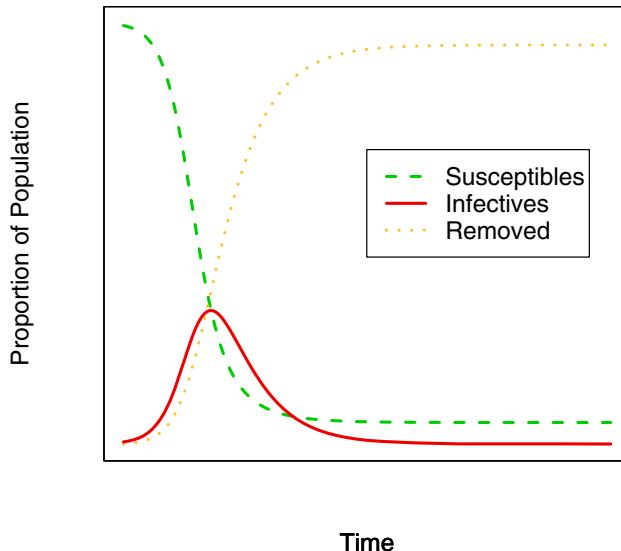


Fig. 8.6 Schematic characterization of an SIR process, showing how the relative proportions of those susceptible (green), infective (red), and removed (yellow) vary over time.

following system of differential equations¹⁴

$$\begin{aligned} \frac{dP_{N,1}(t)}{dt} &= -(\beta N + \gamma)P_{N,1}(t) \\ \frac{dP_{s,i}(t)}{dt} &= \beta(s+1)(i-1)P_{s+1,i-1}(t) - i(\beta s + \gamma)P_{s,i}(t) + \gamma(i+1)P_{s,i+1}(t) , \end{aligned} \quad (8.41)$$

for $0 \leq s+i \leq N+1$, $0 \leq s \leq N$, and $0 \leq i \leq N+1$, subject to the initial conditions that $P_{N,1}(0) = 1$ and $P_{s,i}(0) = 0$ otherwise. However, while an exact analytical solution for this system is possible (e.g, see Daley and Gani [106, Ch. 3.3.1]), its form is quite complicated.

Fortunately, it is possible, without an explicit formula for the functions $P_{s,i}(t)$, to derive certain results regarding the basic qualitative behavior of the epidemic process under this SIR model. Of single greatest interest is the *basic reproduction number* of the epidemic, typically denoted as R_0 , which usually is defined as the number of infections expected in the early stages of an epidemic by a single infec-

¹⁴ These equations are known as the Kolmogorov forward equations or, in chemistry and physics, as the ‘master equations.’

tive in a population of susceptibles.¹⁵ The importance of this quantity in the study of epidemics derives from its role in so-called threshold theorems, which state under which conditions the presence of an infective in a population will lead to an epidemic. For the general epidemic SIR model, the basic reproduction number is $R_0 = N\beta/\gamma$. A celebrated result of Whittle [404] effectively states that, for large N , if $R_0 \leq 1$, there is no chance of an epidemic, while if $R_0 > 1$, an epidemic will occur with probability $1 - (1/R_0)$. Threshold-theorems like this provide a useful point of focus in designing epidemic control procedures (e.g., based on vaccination, education, removal, etc.), in that such procedures essentially must reduce R_0 to less than unity in order to be successful.

Now in practice, the quantities β and γ , and hence R_0 , are unknown. Given adequate data, however, estimates can be produced, with accompanying standard errors. Methods for this purpose are reviewed in Andersson and Britton [13, Chs. 9–12] and Becker and Britton [27]. For example, if $(N_S(t), N_I(t))$ is observed for all $0 \leq t \leq \tau$, then maximum likelihood estimates can be derived in closed form. Specifically, it can be shown that

$$\hat{\beta} = \frac{N - N_S(\tau)}{(1/N) \int_0^\tau N_S(t) N_I(t) dt} \quad (8.42)$$

and

$$\hat{\gamma} = \frac{N_R(\tau)}{\int_0^\tau N_I(t) dt}. \quad (8.43)$$

From these estimates, the maximum likelihood estimate of R_0 then follows as $\hat{R}_0 = N\hat{\beta}/\hat{\gamma}$.

Unfortunately, it is rare that such complete measurements are available. More commonly, only the final state of an epidemic outbreak is observed. That is, we have only ‘final size’ data, in the form of $N_R(\tau)$. Separate estimation of β and γ is no longer possible, since they relate to time and we have no time information. Nevertheless, a version of the method-of-moments can be applied here, producing an estimate of R_0 of the form

$$\hat{R}_0 \approx \frac{-\log[1 - N_R(\tau)/N]}{N_R(\tau)/N}, \quad (8.44)$$

with an accompanying formula for approximate standard errors.

A number of estimation methods have been proposed based on the use of Monte Carlo Markov chain, which exploit the fact that SIR processes often are relatively straightforward to simulate. Such methods are able to deal with some forms of missing data, for example by exploring the space of possible values for variables unobserved over time. Prior distributions can be incorporated in this setting as well. Note that MCMC approaches are in principle available for models much more complicated than the general epidemic SIR model we have considered here, long after an-

¹⁵ A more formal definition relates R_0 to the largest eigenvalue of a matrix of the expected number of offspring of various types, in a branching process approximation of the early stages of the epidemic. In the case of the general epidemic model, this definition reduces to that introduced here. See Heesterbeek [196].

alytical approaches like maximum likelihood and method-of-moments break down. On the other hand, more complicated model structures necessitate increasingly more careful design of the MCMC algorithm to be ensured of proper convergence and mixing of the underlying Markov chain. This is particularly likely to be the case for larger datasets. See Becker and Britton [27] for discussion and references in this area.

8.6.1.2 Network-based Epidemic Modeling

Despite the substantial amount of work in the literature involving the general epidemic SIR model and others like it, the underlying assumption of homogeneous mixing is admittedly simple and, for many diseases, too poor of an approximation to reality. As a result, interest has turned increasingly towards so-called ‘structured population’ models, in which assumed contact patterns take into account some structure(s) within the population of interest. Such structure might derive from spatial proximity (e.g., diseases of plants, in which infection occurs through the help of carriers over short distances), social contact (e.g., sexual contact in the transmission of AIDS), or demographics (e.g., households, age brackets, etc.). Models introduced in this area include independent household models, two-level mixing models, random network models, and social clustering models. A list of some representative citations can be found in Demiris and O’Neill [114], for example, including the seminal paper by Ball, Mollison, and Scalia-Tomba [17].

The end effect of all of these models is, in one way or another, to impose restrictions on the contact structure within the population. Often it is convenient to represent this structure as a graph $G = (V, E)$, where the vertices $i \in V$ represent elements of the population and edges $\{i, j\} \in E$ indicate contact between elements i and j . Note that ‘contact’ here does not indicate actual infection.¹⁶ Rather, it only implies the possibility for infection. The lack of an edge between vertices indicates that no infection is possible between the two. Note too that, viewed from this perspective, the use of homogeneous mixing in a model is equivalent to the assumption that G is a complete graph (i.e., each vertex shares an edge with every other vertex).

The study of epidemic models on network graphs is a topic of much interest, particularly in the areas of applied probability and statistical mechanics, as the study of traditional topics like percolation and phase transitions have strong parallels here. See Barrat, Barthélemy, and Vespignani [20, Ch. 9], for example, for an overview of work in this area. These models are defined at the level of a dynamic network process $\mathbf{X}(t)$. We illustrate by describing a network-based extension of the general epidemic SIR model.

¹⁶ In contrast, digraphs are sometimes used in the literature for analytical purposes to represent actual infections, with an arc between two vertices indicating that the first infected the second. See, for example, Andersson and Britton [13, Ch. 7.1]; also Britton and O’Neill [64] and Demiris and O’Neill [114], who use this device for computational purposes in proposing MCMC-based methods of inference.

Let G be a network graph describing the contact structure among N_v elements in a population. We assume that initially, at time $t = 0$, one vertex is infected and the rest are susceptible. Infected vertices remain infected for an amount of time distributed exponentially, with rate γ , after which they are considered recovered. During the infectious period a vertex has infectious contacts independently¹⁷ with each neighbor, according to an exponential distribution with rate β , where an infectious contact automatically results in infection if the other individual is susceptible. Define $X_i(t) = 0, 1$, or 2 , according to whether vertex i is susceptible, infected, or removed at time t , respectively.

The stochastic process $\mathbf{X}(t)$ forms a continuous-time Markov chain, with state vectors \mathbf{x} consisting of entries with values $0, 1$, or 2 . Successive changes of states, say from \mathbf{x} to \mathbf{x}' , will involve a change in one and only one element at a time. Suppose that \mathbf{x} and \mathbf{x}' differ in the i -th element. Then we may write

$$\mathbb{P}(\mathbf{X}(t + \delta t) = \mathbf{x}' | \mathbf{X}(t) = \mathbf{x}) \approx \begin{cases} \beta M_i(\mathbf{x}) \delta t, & \text{if } x_i = 0 \text{ and } x'_i = 1 , \\ \gamma \delta t, & \text{if } x_i = 1 \text{ and } x'_i = 2 , \\ 1 - [\beta M_i(\mathbf{x}) + \gamma] \delta t, & \text{if } x_i = 2 \text{ and } x'_i = 2 , \end{cases} \quad (8.45)$$

where we define $M_i(\mathbf{x})$ to be the number of neighbors $j \in \mathcal{N}_i$ for which $x_j = 1$ (i.e., the number of neighbors of i infected at time t). Note that in the case of the traditional general epidemic SIR model, where G is the complete graph K_{N_v} , $M_i(\mathbf{x})$ is just the total number of infectives in the population.

Now given $\mathbf{X}(t)$, the processes $N_S(t), N_I(t)$, and $N_R(t)$ may be defined, counting the numbers of susceptible, infective, and removed vertices at time t , respectively, in analogy to the traditional case. In light of the expressions in (8.45), we can expect the characteristics of these processes to be affected to at least some extent by the characteristics of the network graph G . Simulation confirms this expectation. In Figure 8.7, for example, are shown realizations of $N_I(t)$ in the case where G is a random graph of Erdős-Rényi, Barabási-Albert, or Watts-Strogatz type. Although we see that in all three cases the curves $\mathbb{E}(N_I(t))$ have the same general form as in the traditional case (i.e., as depicted in Figure 8.6), it is clear in comparing these curves to each other that they differ in important ways, such as in their rates of growth and decay and in the effective duration of the epidemic.

In principle, differential equations in the spirit of those in (8.41) can be written for network-based SIR processes, but it is rare that they lead to analytically tractable expressions to describe the evolution of these processes. Nevertheless, as in the case of the traditional general epidemic SIR model, it is still possible to derive certain results describing basic qualitative characteristics. For example, modeling G as a generalized random graph, from a collection \mathcal{G} with degree distribution $\{f_d\}$, the basic reproduction number can be shown to equal

¹⁷ Technically, independence pertains only to sufficiently small time intervals. Over the life of the epidemic, the events that an element in the population infects two neighbors are not independent, but rather positively correlated.

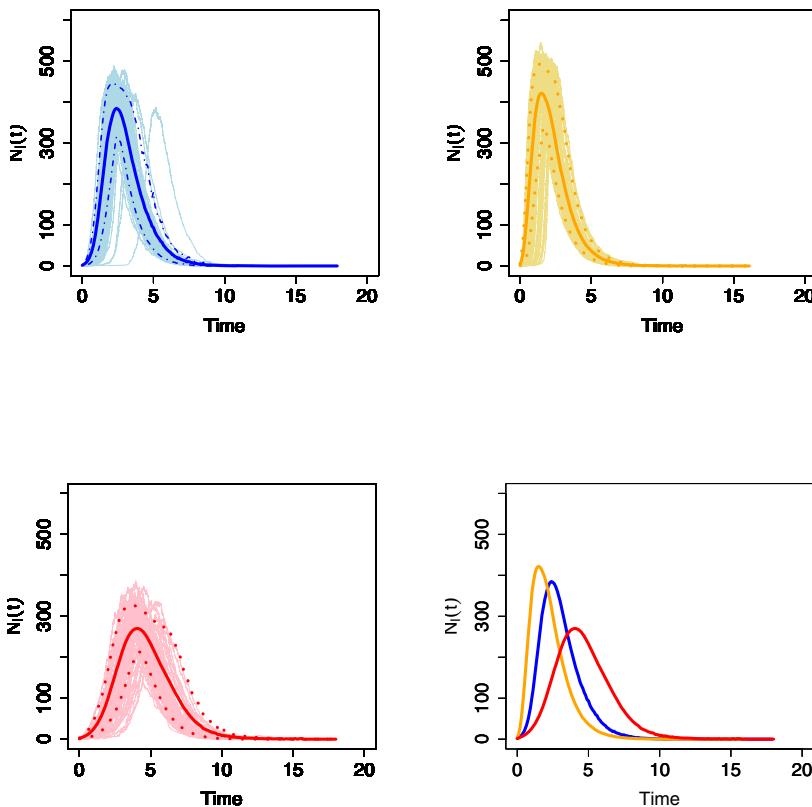


Fig. 8.7 Realizations of the number of infectives $N_I(t)$ for the network-based SIR process simulated on an Erdős-Rényi random graph (blue), a Barabási-Albert random graph (yellow), and a Watts-Strogatz ‘small-world’ random graph (red). Graphs were of order $N_v = 1,000$, with parameters set so that the average degree \bar{d} was approximately 10 in each case. The rewiring probability for the Watts-Strogatz random graph was $p = 0.10$. Rates of infection and recovery were $\beta = 0.5$ and $\gamma = 1.0$, respectively. Darker curves indicate the average (solid) and the 10th and 90th percentile (dotted), over a total of 1,000 epidemics. Results of a randomly selected 100 epidemics are shown in light curves for each graph. The three mean functions are compared in the lower right-hand plot.

$$R_0 = \frac{\beta}{\beta + \gamma} \left[\frac{\mathbb{E}(d^2)}{\mathbb{E}(d)} - 1 \right] , \quad (8.46)$$

where $\mathbb{E}(d)$ and $\mathbb{E}(d^2)$ are the first two moments of f_d . See Andersson [12] and Andersson and Britton [13, Ch. 7], and also Newman [292]. The second term in this expression is the expected number of neighbors in G of a single infective during the

early stages of the epidemic, and the first term is the probability that the infective transmits the infection to an arbitrary neighbor before recovering.

Based on (8.46), we can make certain broad characterizations as to the effect of network topology on the likelihood of an epidemic. In the case where $\mathcal{G} = \mathcal{G}_{N_v, p}$ is a collection of Erdős-Rényi random graphs, with parameter p , we see that $R_0 \approx (\beta N_v p) / (\beta + \gamma)$. Alternatively, however, in the case where the degree distribution $\{f_d\}$ is heterogeneous, we can expect $\mathbb{E}(d^2) \gg \mathbb{E}(d)$, which has the effect of dramatically increasing R_0 . Thus, epidemics can occur much more easily in comparison to the homogeneous case. Intuitively, we can see that such behavior is to be expected, given that the infection of even a small number of high-degree vertices will quickly place a large fraction of the rest of the population at risk. As a side note, we point out that the expression $R_0 = \beta N_v / \gamma$ for the traditional general epidemic SIR model and that given in (8.46) can be made to coincide with an appropriate limiting argument. See Andersson and Britton[13, Ch. 7.3].

In the area of statistical inference with network-based epidemic process models, it appears that to date only a relatively small amount of work has been done. Ideally we would like to observe both the stochastic process $(N_S(t), N_I(t), N_R(t))$, over all times $0 \leq t \leq \tau$, and the underlying network graph G . In practice, however, one or both of these are likely to be at least partially unobserved. Keeling [222] has conducted an initial exploration of the case in which $N_I(t)$ is observed, but G is unobserved, using nonlinear regression to fit simple parametric functions to $N_I(t)$. But the task of specifying an appropriate functional form, which captures the implicit effects of the network, appears to be nontrivial. Britton and O'Neill [64] consider the situation in which $(N_I(t), N_R(t))$ is observed, and model G as an Erdős-Rényi random graph in $\mathcal{G}_{N_v, p}$. They propose a Bayesian approach to inference of the parameters β, γ , and p , treating both G and the pattern of transmission of the infection among elements as missing data. While the likelihood under this model is intractable, being composed of a sum over all possible realizations of the missing data, these authors are able to offer an MCMC-based algorithm to sample effectively from the corresponding posterior. They also address the case where only $N_R(t)$ is observed, treating $N_I(t)$ as missing as well.

Alternatively, Britton, Nordvik, and Liljeros [65] illustrate how, in the context of a sexually transmitted infection, expressions like (8.46) can be used to produce plug-in estimates of R_0 , in the case where a sample, say G^* , of G is observed and data relating to the probability of transmission are available. Such an approach assumes, of course, that G^* is sufficiently representative of G , which, as we saw in Chapter 5, need not necessarily be the case. In general, the sampling of epidemic contact networks is a task fraught with challenges. See Keeling and Eames [223] for discussion. In addition, from a modeling perspective, and reminiscent of the topics in Sections 3.2 and 3.3, we note that an effective choice of the network graph G most relevant for a given disease may not always be clear or, even if reasonably clear, may not always be measurable.

8.6.2 Other Dynamic Processes

Beyond epidemics, substantial work may be found as well on the network-based modeling of certain other dynamic processes. We briefly describe work in a few such areas. Again, however, as in the case of epidemic modeling, we note that the majority of the work to date appears to be focused on mathematical aspects, rather than statistical aspects.

The tasks of navigation and search in many contexts can be usefully related to random walks and diffusions on graphs. That is, we can picture a particle(s) moving along edges, from vertex to vertex, in a random manner, as an approximation to, say, how a human might follow hyperlinks in the World Wide Web in search of a web page with a particular type of content. Whether or not each vertex has or has not been occupied by a particle, as a function of time t , can be represented as a binary stochastic process $\mathbf{X}(t)$. Results involving, for instance, the chance that a vertex is ‘occupied’ or the time for a search to return to a given vertex, have been obtained, in the limit of large time t . The effects of degree distributions and small-world structure on such quantities have been topics of particular interest. See Barrat, Barthélémy, and Vespignani [20, Ch. 8], for example, for an overview.

Similarly, another related topic is the mathematical modeling of collective behavior in social systems, such as the spread of rumors or information throughout a population. In some situations, epidemic models for disease spread, like those described above, can be adopted wholesale, where susceptibles are now called ‘ignorants’ and infectives, ‘spreaders.’ In other cases, however, important changes are necessary, such as in the presence of ‘stiflers’ (i.e., spreaders that, upon encountering another spreader, stop spreading). The fact that this halt in spreading depends upon the interactions in the process, rather than a spontaneous random event, impacts the qualitative behavior of the system. See, for example, Barrat, Barthélémy, and Vespignani [20, Ch. 10].

Finally, in the area of neuro-science, a great deal of work has been done on the mathematical modeling of neurological systems, where a network-based perspective is natural. In this setting, the choice of elements in the system ranges, in the brain for instance, from individual neurons to entire functional regions (e.g., the frontal cortex), and interactions between elements typically are associated with the coupling of their behavior. The processes $\mathbf{X}(t)$ often are modeled as taking on continuous states, rather than discrete states, a difference that necessitates a somewhat more technical treatment of the underlying mathematics. The types of questions of interest in this context also differ, focusing, for example, on issues of synchronization among elements in the network; that is, where the functions $X_i(t)$ follow a common pattern of evolution over time t , for different $i \in V$. Again, the impact of various classes of network topology (e.g., small-world, Erdős-Rényi, scale-free, etc.) on the behavior of the system has been a central theme. See Barrat, Barthélémy, and Vespignani [20, Ch. 7].

With respect to statistical work in contexts like those above, in principle there are a variety of paradigms that can be – and have been, to various extents – applied for parameter inference and prediction. Examples include traditional time se-

ries analysis (e.g., Brockwell and Davis [66] and similar texts), Kalman filtering and related methods of Bayesian forecasting (e.g., West and Harrison [400]), and approaches based on diffusion approximations and nonlinear regression (e.g., see Ramsay, Hooker, Campbell, and Cao [321], which includes a substantial survey of related work). However, there has yet to be a concerted effort at exploring a principled extension of such tools to network-based statistical modeling nor, indeed, an effort analogous to that on the mathematical side to assess the impact that incorporation of a network aspect may or may not have.

Although this state of affairs is perhaps beginning to change. Some initial efforts in this direction may be found, for example, in the work of Snijders and colleagues [361, 70], in the context of social networks, and of Golightly and Wilkinson [180], for modeling biochemical networks. Snijders and colleagues develop models from the perspective of continuous-time Markov chains, where the rates at which the process changes states are modeled using functions motivated by socio-economic principles. Parameter estimates are fit using a method-of-moments approach and MCMC. In applications, however, the method is implemented only on small networks with just a handful of time points. The basic model of Golightly and Wilkinson is also a continuous-time Markov chain, derived from consideration of stochastic kinetic models, which is then approximated (i.e., using a so-called ‘diffusion approximation’) to create a time-indexed Gaussian-noise model and fit using MCMC methods. This method requires data obtained from real-time monitoring of gene expression levels and was developed with an eye towards expected future advances in the necessary measurement technology. It too, however, has difficulties scaling beyond relatively small networks.

8.7 Additional Related Topics and Reading

Our focus in this chapter has been largely on the modeling and prediction of static processes on network graphs, and we have concentrated mainly on presenting certain of the somewhat more established methods. Various other methods have been proposed as well. One such class of methods are those based on relational probabilistic models, as developed by Koller and colleagues (e.g., see the volume edited by Getoor and Taskar [168]). These models are an extension of the class of graphical models – which we discuss briefly in Chapter 10 – to relational data. We note too that there are various more informal approaches that have been proposed as well for specific versions of the prediction problem, such as those based on clustering. See the review by Sharan, Ulitsky, and Shamir [347], for instance, for a survey of such methods in the context of protein function prediction.

A literature related to the material of this chapter is that pertaining to graph-based analysis of high-dimensional data that lie on or near a low-dimensional surface (i.e., a manifold), which was already mentioned briefly at the end of Section 4.6. The fundamental interest in this context has been on dimension reduction, where the goal is to take n high-dimensional observations and replace them by n low-dimensional

values that represent the original values accurately in some well-defined sense. Exploiting the expectation that the low-dimensional representation be driven by an underlying low-dimensional surface, a graph is constructed for the data, for instance by linking each observation to its k nearest neighbors, and an approximate geometry of the surface is inferred from the graph using, for example, the first $m \ll n$ eigenvectors of the graph Laplacian. Seminal work of this nature includes Roweis and Saul [334], Tenenbaum, Silva, and Langford [378], Belkin and Niyogi [28], and Donoho and Grimes [120]. Such methods may be viewed as nonlinear extensions of the method of principle component analysis used in multivariate statistical analysis (e.g., see Hastie, Tibshirani, and Friedman [194, Ch. 14.5]).

With respect to the modeling and prediction of dynamic processes on network graphs, we note that the assumption that the graph G be fixed, while the process $\mathbf{X}(t)$ varies in time, is generally only an approximation to the reality of most systems. In practice, over appropriate time scales, G too is likely to be time varying. However, in principle, if the evolution of G is slow enough, compared to the dynamics of $\mathbf{X}(t)$, this approximation should be reasonable, as noted, for example, by Keeling and Eames [223] in the context of epidemic modeling. Otherwise, it may be necessary to model the evolution of both network graph and process jointly in time. Some initial work on this challenging task has been conducted by Snijders and colleagues [361, 70] and by Getoor, Friedman, Koller, and Taskar [169].

Exercises

8.1. A slightly different formulation of an auto-Gaussian model specifies that the conditional distributions $\mathbb{P}(X_i = x_i | \mathbf{X}_{(-i)} = \mathbf{x}_{(-i)})$ in (8.3) be univariate Gaussian, with expectation

$$\mathbb{E}(X_i | \mathbf{X}_{\mathcal{N}_i} = \mathbf{x}_{\mathcal{N}_i}) = \alpha_i + \sum_{j \in \mathcal{N}_i} \beta_{ij} (x_j - \alpha_j) ,$$

as in (8.14), and variance σ_i^2 , where $\beta_{ij}\sigma_j^2 = \beta_{ji}\sigma_i^2$. Under these conditions, it can be shown (e.g., Cressie [102, Ch. 6.6]) that the joint distribution of \mathbf{X} is multivariate Gaussian, with mean vector $\mu = (\alpha_i)$ and covariance matrix $\Sigma = (I - \mathbf{B})^{-1} \mathbf{M}$, where $\mathbf{M} = \text{diag}[(\sigma_i^2)]$.

- a. Let $G = (V, E)$ be a connected network graph, and define $\beta_{ij} = d_i^{-1}$, for $j \in \mathcal{N}_i$, and $\beta_{ij} = 0$, otherwise, where d_i is the degree of vertex i . In addition, define $\sigma_i^2 = \sigma^2/d_i$. Argue that in this case \mathbf{X} is distributed as multivariate Gaussian, with mean vector $\mu = (\alpha_i)$ and covariance matrix $\Sigma = \sigma^2 \mathbf{L}^-$, where \mathbf{L} is the Laplacian of G and \mathbf{L}^- denotes its (pseudo)inverse.
- b. For a network graph G of interest to you, use the Gibbs sampling algorithm outlined in Section 8.3.2.2 to simulate realizations of \mathbf{X} according to the model described in part (a), with $\mu = \mathbf{0}$. Employing appropriate tools for visualization and numerical summary, explore the behavior of \mathbf{X} on G .

- c. Designate a portion of the vertices in your graph G to have values X_i that will be ‘missing.’ Modify your Gibbs sampling algorithm to predict these missing values, given observations at the remaining vertices, as described in Section 8.3.2.2. Explore the accuracy with which you are able to predict missing values, as a function of the vertices at which they arise. For example, consider the effect of vertex degree on the predictive accuracy of your algorithm.

8.2. Consider the method of kernel ridge regression outlined in Example 8.4. In practice, it is common to include an ‘intercept’ – or off-set parameter – in the model. As a result, rather than the optimization in (8.27), we solve the problem

$$\min_{\alpha_0, \alpha} \left[\left(\mathbf{x}^{obs} - \alpha_0 - \mathbf{K}^{(n)} \alpha \right)^T \left(\mathbf{x}^{obs} - \alpha_0 - \mathbf{K}^{(n)} \alpha \right) + \lambda \alpha^T \mathbf{K}^{(n)} \alpha \right] ,$$

where α_0 is the off-set parameter.

- a. Show that the solution $(\hat{\alpha}_0, \hat{\alpha})$ takes the form

$$\hat{\alpha}_0 = \bar{x} - (1/n) \mathbf{1}^T M \hat{\theta} \quad \text{and} \quad \hat{\alpha} = \Phi^{(n)} \Delta^{(n)}^{-1/2} \hat{\theta} ,$$

where

$$\hat{\theta} = [\mathbf{M}^T (\mathbf{I} - \mathbf{1}\mathbf{1}^T) \mathbf{M} + \lambda I]^{-1} \mathbf{M}^T \left(\mathbf{x}^{obs} - \bar{x} \mathbf{1} \right) ,$$

for $\mathbf{1}$ an $n \times 1$ vector of one’s and \bar{x} the average of the elements in \mathbf{x}^{obs} . Compare the above expression for $\hat{\theta}$ with that in (8.30).

- b. For binary classification problems, methods based on squared-error loss often can produce surprisingly good results (e.g., see Hastie, Tibshirani, and Friedman [194, Ch. 4.2]). For the protein function data featured in the case study of Section 8.5, apply the kernel ridge regression estimator in part (a). Specifically, convert the binary response \mathbf{x}^{obs} to values of -1 and $+1$, through the transformation $x_i^{obs} \rightarrow 2x_i^{obs} - 1$, and classify a vertex i as $+1$ if $\hat{\alpha}_0 + (\mathbf{K}^{(N_v, n)} \hat{\alpha})_i \geq 0$, and -1 , otherwise. Explore the appropriateness of various choices of kernel. To what extent does the performance of your classifier correlate with the kernel-target alignment $a(\mathbf{K}, \mathbf{x}\mathbf{x}^T)$?

8.3. The simulation of a network-based general epidemic process $\mathbf{X}(t)$, like that described in Section 8.6.1.2, is straightforward, using discrete-event simulation (DES) methods. Let $G = (V, E)$ be a network graph, and $\beta, \gamma > 0$, the infection and removal rates, respectively. At an arbitrary time $t \geq 0$, let \mathbf{x} denote the state of the process \mathbf{X} , where $x_i = 0, 1$, or 2 indicates that vertex $i \in V$ is in a susceptible, infected, or removed state at time t . Let $M_i(\mathbf{x})$ be the number of neighbors of i in G that are infected at time t . A general epidemic process may be simulated on G using the following algorithm:

initialize $t = 0, X_{i^*}(0) = 1$ for some $i^* \in V$, and $X_i(0) = 0$ for all $i \in V \setminus \{i^*\}$
while $\lambda > 0$
sum $\lambda = \sum_{i=1}^{N_v} \lambda_i$, where

$$\lambda_i = \begin{cases} \beta M_i(\mathbf{x}), & \text{if } x_i = 0 \\ \gamma, & \text{if } x_i = 1 \\ 0, & \text{otherwise} \end{cases}$$

draw $T \sim \text{Exponential}(\lambda)$
update $t = t + T$
draw $i \sim \{p_i\}_{i=1}^{N_v}$, where $p_i = \lambda_i / \lambda$
update \mathbf{X} at time t at vertex i , according to

$$X_i(t) = \begin{cases} 2, & \text{if } x_i = 1 \\ 1, & \text{if } x_i = 0 \end{cases}$$

end while

For various choices of random graph G , such as those underlying Figure 8.7), and using the above algorithm as a guide, write an appropriate piece of software to simulate a general epidemic process $\mathbf{X}(t)$ on G . Explore the effects on the induced processes $(N_S(t), N_I(t), N_R(t))$ of changes to the topology of G and to the rates β and γ . Use the expression for the basic reproduction number R_0 in (8.46) to inform your choice of settings.

Chapter 9

Analysis of Network Flow Data

Flows are at the heart of the form and function of many networks, and understanding their behavior is often a goal of primary interest. Here we consider problems of statistical estimation and prediction arising in connection with various types of measurements relating to network flows.

9.1 Introduction

Many networks serve as conduits – either literally or figuratively – for *flows*, in the sense that they facilitate the movement of something, such as materials, people, or information. For example, transportation networks (e.g., of highways, railways, and airlines) support flows of commodities and people, communication networks allow for the flow of data, and networks of trade relations among nations reflect the flow of capital. We will generically refer to that of which a flow consists as *traffic*.

Much of the quantitative work in the literature on flows is concerned with various types of problems involving, for example, questions of network design, provisioning, and routing, and their solutions have involved primarily tools in optimization and algorithms. We will not address such problems here; see Phillips and Garcia-Diaz [313] for a general overview of this area. Instead, we will concern ourselves – broadly speaking – with the statistical modeling and analysis of measurements relating to network flows. More specifically, we will look at a handful of problems in which regression-like models can be used to describe the association between one set of flow-related variables and another, and regression-based inference and prediction can be used to infer unknown characteristics of the network flows.

Let $G = (V, E)$ be a network graph. Since flows have direction, from an origin to a destination, formally G will be a digraph. Following the convention in this literature, we will refer to the (directed) edges in this graph as links. Traffic typically passes over multiple links in moving between origin and destination vertices. The manner in which traffic moves throughout the network is captured by a matrix, say \mathbf{B} , called the *routing matrix*, defined to have the same number of rows and columns

as links and origin-destination pairs in G , respectively. In the case that each origin-destination pair (i, j) has only a single route, from i to j , \mathbf{B} is a binary matrix, with the entry in the row corresponding to link e and the column corresponding to pair (i, j) being

$$B_{e;ij} = \begin{cases} 1, & \text{if link } e \text{ is traversed in going from } i \text{ to } j \\ 0, & \text{otherwise} \end{cases}, \quad (9.1)$$

That is, \mathbf{B} is in this case a matrix describing the incidence of routes with links. If multiple routes are possible, the entries of \mathbf{B} are instead fractions representing, for example, the proportion of traffic from i to j that is expected to use the link e . We will frequently find it convenient here in this chapter to assume, wherever it can aid in simplifying our presentation, that routes between origin-destination pairs are unique.

A quantity of fundamental interest in the study of network flows is the so-called *origin-destination* (OD) matrix, which we will denote by $\mathbf{Z} = [Z_{ij}]$, where Z_{ij} is the total volume of traffic flowing from an origin vertex i to a destination vertex j in a given period of time. The matrix \mathbf{Z} is also sometimes referred to as the *traffic matrix*. A number of other quantities of interest relate directly to \mathbf{Z} . For example, the *net out-flow* and *net in-flow* corresponding to vertices i and j can be represented through the sums

$$Z_{i+} = \sum_j Z_{ij} \quad \text{and} \quad Z_{+j} = \sum_i Z_{ij}, \quad (9.2)$$

respectively. Similarly, defining X_e to be the total flow over a given link $e \in E$, and letting $\mathbf{X} = (X_e)_{e \in E}$, the link totals in \mathbf{X} can be related to \mathbf{Z} through the expression $\mathbf{X} = \mathbf{BZ}$, where \mathbf{Z} now represents our traffic matrix written as a vector.

In addition to these various measures of traffic volume, the notion of a ‘cost’ c , usually associated either with paths or links, is also important. For example, the concept of *generalized cost* in a socioeconomic sense, is often invoked in transportation analysis, where it may be desired to understand and model the behavior of consumers of transportation resources, particularly the manner in which they make choices in their consumption. Similarly, the concept of *quality of service* (QoS) is important in the literature on computer network traffic analysis, where it is in the interests of both Internet network service providers and their consumers to, for example, monitor basic performance characteristics of the network, detect areas of congestion or anomalous behavior, and validate compliance with contractual service agreements.

Note that our notation above implicitly assumes a static snapshot of the flows in the network, a perspective that is convenient and adopted often in analyzing network flow data. But given the dynamic nature of flows, a time-varying perspective is generally more realistic and, depending on the context, may be more appropriate. Where relevant, we will signify this change by equipping the various quantities with a time index t . For example, a time-dependent traffic matrix will be denoted $\mathbf{Z}^{(t)}$. However, we will largely consider the routing matrix \mathbf{B} to be fixed in the ma-

Section	Measurements	Analysis Goal
9.2	OD flow volumes Z_{ij}	Model observed flow volumes Z_{ij}
9.3	Link volumes X_e	Predict unobserved OD flow volumes Z_{ij}
9.4	OD costs c_{ij}	Predict unobserved OD and link costs

Table 9.1 Summary of topics covered in Chapter 9, broken down by type of measurements taken and the goal of the statistical analysis.

terial that follows. This assumption is common and is generally justified in contexts where changes in routing occur on longer time scales than those associated with the dynamics of the underlying flows.

The statistical problems we consider in this chapter are organized largely according to the types of measurements taken, and the goal of the statistical analysis in which they are to be used, as summarized in Table 9.1. For example, in some contexts, it is possible to observe the entire traffic matrix \mathbf{Z} , and it is of interest to model these observations, both to obtain an understanding as to how potentially relevant factors (e.g., costs) affect flow volumes and to be able to make predictions of future flow volumes. A class of models commonly used for these purposes are the so-called gravity models, which we discuss in Section 9.2. In many contexts, however, it is difficult or impossible to observe the traffic matrix entries Z_{ij} directly, but comparatively easier to measure the link totals X_e . Recovery of the traffic matrix entries from these totals is then of central interest. We present a number of approaches to the problem of traffic matrix estimation in Section 9.3. Finally, there is the problem of modeling and inference of network cost parameters. In Section 9.4 we consider two related problems of this type, wherein it is assumed possible to obtain direct measurements of a cost c_{ij} for some subset of origin-destination pairs (i, j) , and the goal is to infer characteristics of this cost more broadly across the network.

9.2 Gravity Models

Gravity models are a class of models, developed largely in the social sciences, for describing aggregate levels of interaction among the people of different populations. They have traditionally been used most in areas like geography, economics, and sociology, for example, but also have found application in other areas of the sciences, such as hydrology and the analysis of computer network traffic. Although our interest in gravity models in this chapter will be largely for their use in contexts where the relevant traffic flows are over a network of sorts, we note that it is not necessary for the formulation of these models that a network be defined explicitly, and our exposition in this section will reflect this fact accordingly. In our use of gravity models later in the chapter, however, the role of a network will be made explicit.

The term ‘gravity model’ derives from the fact that, in analogy to Newton’s law of universal gravitation, it is assumed that the interaction among two populations

varies in direct proportion to their size, and inversely, with some measure of their separation. The concept goes back at least to the work of Carey [74] in the 1850's, but arguably was formulated in the strictest sense of the analogy by Stewart [368] in 1941. This area has been developed substantially in the past 50 years. Here we focus chiefly on providing an overview of a general version of the gravity model and corresponding methods of inference. For a comprehensive introduction to the topic, from the perspective of statistical modeling and inference, see the book by Sen and Smith [345], which itself contains references to numerous other works.

9.2.1 Model Specification

Suppose that \mathcal{I} and \mathcal{J} represent sets of origins and destinations, of cardinality $I = |\mathcal{I}|$ and $J = |\mathcal{J}|$, respectively, and Z_{ij} denotes – as defined in the introduction – a measure of the traffic flowing from $i \in \mathcal{I}$ to $j \in \mathcal{J}$ over a given period of time. The *general gravity model* specifies that the traffic flows Z_{ij} be in the form of counts, with independent Poisson distributions and mean functions of the form

$$\mathbb{E}(Z_{ij}) = h_O(i) h_D(j) h_S(\mathbf{c}_{ij}) , \quad (9.3)$$

where h_O , h_D , and h_S are positive functions, respectively, of the origin i , the destination j , and a vector \mathbf{c}_{ij} of K so-called separation attributes. The K elements of \mathbf{c}_{ij} are chosen to quantify some notion(s) of separation ascribed to the origin-destination pair (i, j) , often based on concepts of ‘distance’ or ‘cost’ of either a literal or figurative nature. The functions h_O and h_D are sometimes referred to as the *origin* and *destination functions*, respectively, while h_S is commonly called the *separation* or *deterrence function*. Often the separation function is constrained to be non-increasing in the elements of \mathbf{c}_{ij} .

An early and now classical example of the gravity model is that of Stewart [368], proposed in connection with his theory of ‘demographic gravitation,’ which specifies that

$$\mathbb{E}(Z_{ij}) = \gamma \pi_{O,i} \pi_{D,j} d_{ij}^{-2} , \quad (9.4)$$

where $\pi_{O,i}$ and $\pi_{D,j}$ are measures of the origin and destination population sizes, respectively, for two geographical regions i and j , and d_{ij} is a measure of distance between the centers of these regions. This formulation is completely analogous to Newton’s universal law, right down to the use of a ‘demographic gravitational constant’ γ . However, unlike Newton’s law, neither empirical evidence nor theoretical arguments suggest that this form of the gravity model is strictly accurate in practical contexts.

Many proposals have been made for alternative forms of origin, destination, and separation functions. For example, power functions have been proposed for the origin and destination functions of the form

$$h_O(i) = (\pi_{O,i})^\alpha \quad \text{and} \quad h_D(j) = (\pi_{D,j})^\beta , \quad (9.5)$$

with exponents $\alpha, \beta \geq 0$. If more than one relevant measure of population size (or some other similarly predictive quantity) is to be used, it is common to generalize the formulas for $h_O(i)$ and $h_D(j)$ in (9.5) to products of such power functions. Alternatively, the values $\{h_O(i)\}_{i \in \mathcal{I}}$ and $\{h_D(j)\}_{j \in \mathcal{J}}$ are often simply treated as a collection of $I + J$ unknown parameters.

Power functions also have been proposed for the separation function, but in the form

$$h_S(c_{ij}) = (c_{ij})^{-\theta} , \quad (9.6)$$

for scalar $c_{ij} \in \mathbb{R}$, with $\theta \geq 0$, so as to force the function to be non-increasing in c_{ij} . However, arguably more popular is the use of exponential functions of the general form

$$h_S(\mathbf{c}_{ij}) = \exp(\theta^T \mathbf{c}_{ij}) , \quad (9.7)$$

for vectors $\theta, \mathbf{c}_{ij} \in \mathbb{R}^K$.

These choices of origin, destination, and separation functions generalize the model of Stewart in a natural manner. They can be motivated using various arguments, based on perspectives ranging from sociophysics to economic utility theory to maximum entropy. See Sen and Smith [345] and Fotheringham and O'Kelly [147], for example, for extensive treatments of such arguments. We note, however, that these choices are also convenient, in that on the logarithmic scale, the mean function in (9.3) becomes linear in the unknown parameters. For example, specifying power origin and destination functions and an exponential separation function, we have

$$\log \mathbb{E}(Z_{ij}) = \alpha \log \pi_{O,i} + \beta \log \pi_{D,j} + \theta^T \mathbf{c}_{ij} . \quad (9.8)$$

This log-linear form facilitates the use of log-linear and linear regression methods for statistical inference on the model parameters, as we discuss below in Section 9.2.2.

To illustrate the preceding discussion, consider the following example.

Example 9.1 (Austrian Call Data). It is of interest to understand the spatial structure of telecommunication interactions among populations in different geographical regions. Such understanding is relevant to, for example, planning for government (de)regulation of the telecommunication sector and anticipating the influence of telecommunication technologies on regional development. It is natural to think of telecommunication patterns as flows, and gravity models are a standard tool for modeling these flows. The choice of relevant variables for constructing such models depends upon many factors, including the type of telecommunication and the level of spatial aggregation involved. Here we consider data on a network of inter-regional phone traffic (i.e., phone calls between geographical regions).

Fischer and Gopal [145] describe a set of data for phone traffic between 32 telecommunication districts in Austria throughout a period during the year 1991.

These data¹ consist of $32 \times 31 = 992$ flow measurements z_{ij} , $i \neq j = 1, \dots, 32$, capturing contact intensity over the period studied, and are shown in Figure 9.1. Also shown are data quantifying the gross regional product (GRP) for each region, which may serve as a proxy for economic activity and income, both of which are relevant to business and private phone calls. In addition, data reflecting a notion of road-based distance between regions are shown too.

All data are shown on a logarithmic scale (using base 10, for ease of interpretation). It is evident that there is a reasonably strong relationship between call volume and the origin GRP, destination GRP, and distance. Moreover, this relationship is fairly linear (i.e., the dotted lines, produced by ordinary least-squares, lie close to the solid lines, which are the result of a nonparametric smoother) and is increasing in origin and destination GRP and decreasing in distance. These observations suggest that a model of the form

$$\mathbb{E}(Z_{ij}) = \gamma (\pi_{O,i})^\alpha (\pi_{D,j})^\beta (c_{ij})^{-\theta} \quad (9.9)$$

might be reasonable, where $\pi_{O,i}$ is the GRP of origin i , $\pi_{D,j}$ is the GRP of destination j , and c_{ij} is the distance from origin i to destination j .

Note, however, that the flow volumes between regions vary dramatically in scale. Most origin-destination pairs exchanged a volume of traffic in the range of thousands to tens of thousands, but some had a volume in the hundreds of thousands, while a few had a volume of only a hundred or less. Such disparity in magnitude is not atypical of flow volume data and tends to make it difficult to achieve uniform accuracy across origin-destination pairs, as we will see later. \square

An alternative parameterization under which gravity models frequently are formulated is in terms of the so-called *interaction probabilities*

$$f_{ij} = \mathbb{E}(Z_{ij}/Z_{++} | Z_{++} > 0) , \quad (9.10)$$

where $Z_{++} = \sum_{i,j} Z_{ij}$. These ‘probabilities’ represent the expected relative frequency at which interactions are specifically ij -interactions. Under the general gravity model specification in (9.3), they can be expressed as

$$f_{ij} = \frac{h_O(i) h_D(j) h_S(\mathbf{c}_{ij})}{\sum_{i' \in \mathcal{I}, j' \in \mathcal{J}} h_O(i') h_D(j') h_S(\mathbf{c}_{i'j'})} . \quad (9.11)$$

In addition to the general gravity model in (9.3), there are various other related forms of gravity models. For example, *destination gravity models* pertain to the counts Z_{ij} to all destinations j from a given origin i . This perspective corresponds to considering the so-called *conditional destination probabilities*

¹ The original data are in units of erlang (i.e., number of phone calls, including faxes, times the average length of the call divided by the duration of the measurement period). For the purpose of illustration, we have converted these data to units of counts, by scaling by the minimum measurement value and rounding, to enable the types of log-linear statistical analyses described below in Section 9.2.2.

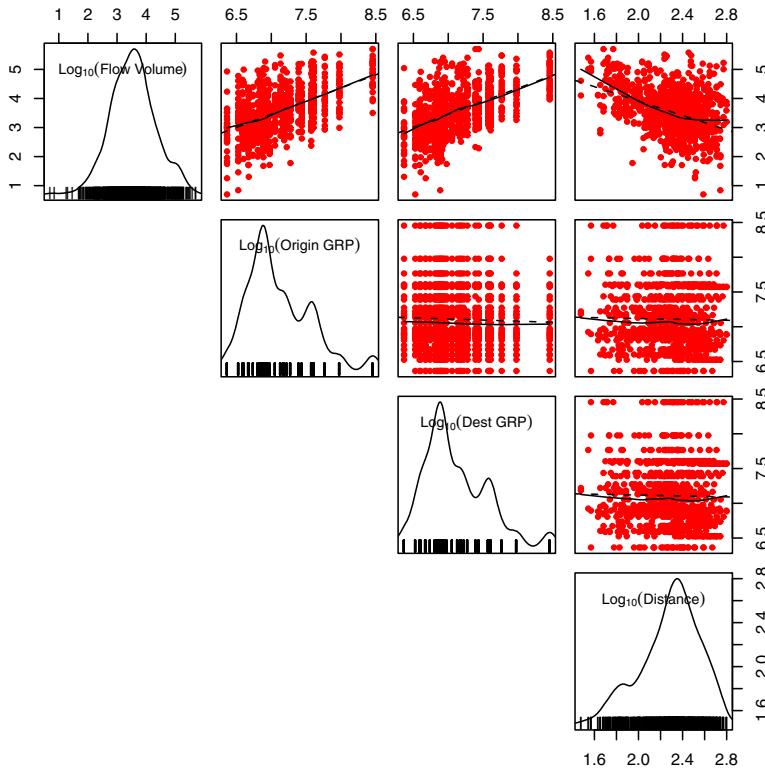


Fig. 9.1 Austrian call data. Scatterplots are shown for call flow volume versus each of origin GRP, destination GRP, and distance, along the top row, and for the latter three variables against each other, in the other rows. All axes are on log-log scales. Superimposed on each scatterplot are two lines, for descriptive purposes, showing fits based on simple linear regression (dotted) and a nonparametric smoother (solid). Density plots for each of the four variables are shown along the diagonal.

$$f_{j|i} = \frac{f_{ij}}{\sum_{j' \in \mathcal{J}} f_{ij'}} , \quad (9.12)$$

rather than the unconditional probabilities f_{ij} . In terms of the components in (9.3), these values can be expressed as

$$f_{j|i} = \frac{h_D(j) h_S(\mathbf{c}_{ij})}{\sum_{j' \in \mathcal{J}} h_D(j') h_S(\mathbf{c}_{ij'})} . \quad (9.13)$$

The set of *origin gravity models* can be specified analogously and pertain to *conditional origin probabilities* f_{ilj} , which can be expressed as in (9.13), but with $h_D(j)$ replaced by $h_O(i)$ in numerator and denominator.

For a comprehensive treatment of additional variations on the general gravity model, including versions that relax the assumption of independence between origin and destination effects implicit in the product form $h_O(i) \times h_D(j)$, see Sen and Smith [345].

9.2.2 Inference for Gravity Models

In light of the specification that the Z_{ij} be independent Poisson random variables with means $\mu_{ij} = \mathbb{E}(Z_{ij})$, statistical inference in the general gravity model is most naturally approached through likelihood-based methods. Our discussion here will focus on the model

$$\log \mu_{ij} = \alpha_i + \beta_j + \theta^T \mathbf{c}_{ij} , \quad (9.14)$$

where $\alpha_i = \log h_O(i)$, $\beta_j = \log h_D(j)$, and $\theta, \mathbf{c}_{ij} \in \mathbb{R}^K$. Other cases, such as when the origin and destination functions are parameterized, as in (9.8), may be handled similarly. These models are log-linear models, which in turn are a specific instance of the class of generalized linear models. See, for example, McCullagh and Nelder [272, Ch. 6].

Let $\mathbf{Z} = \mathbf{z}$ be an $(IJ) \times 1$ vector of observations of the flows Z_{ij} , which for convenience are usually ordered by origin i , and by destination j within origin i . The relevant portion of the Poisson log-likelihood for μ takes the form

$$\ell(\mu) = \sum_{i,j \in \mathcal{I} \times \mathcal{J}} z_{ij} \log \mu_{ij} - \mu_{ij} . \quad (9.15)$$

Substituting the gravity model (9.14), taking partial derivatives with respect to the parameters α_i , β_j , and θ_k , setting the resulting equations equal to zero, and simplifying, it can be shown that the maximum likelihood estimates for these parameters must yield estimates $\hat{\mu}_{ij} = \hat{\alpha}_i \hat{\beta}_j \exp(\hat{\theta}^T \mathbf{c}_{ij})$, for μ_{ij} satisfying the equations

$$\hat{\mu}_{i+} = z_{i+}, \text{ for } i \in \mathcal{I} \quad \text{and} \quad \hat{\mu}_{+j} = z_{+j}, \text{ for } j \in \mathcal{J} \quad (9.16)$$

$$\sum_{i,j \in \mathcal{I} \times \mathcal{J}} c_{ij;k} \hat{\mu}_{ij} = \sum_{i,j \in \mathcal{I} \times \mathcal{J}} c_{ij;k} z_{ij}, \text{ for } k = 1, \dots, K , \quad (9.17)$$

where $\hat{\mu}_{i+} = \sum_{j \in \mathcal{J}} \mu_{ij}$ and $\hat{\mu}_{+j} = \sum_{i \in \mathcal{I}} \mu_{ij}$, and the z_{i+} and z_{+j} are defined similarly. Here $c_{ij;k}$ denotes the k -th element of \mathbf{c}_{ij} . Under mild conditions, these estimates will be well defined, and furthermore, the values $\hat{\theta}_k$ and $\hat{\mu}_{ij}$ will be unique. The values $\hat{\alpha}_i$ and $\hat{\beta}_j$ will be unique only up to a constant, due to the fact that the

underlying model is over-parameterized by one degree of freedom.² Note that we are assuming here, without loss of generality, that origins i and destinations j for which $z_{i+} = 0$ or $z_{+j} = 0$ are dropped, as they contribute nothing to the analysis.

Various algorithms may be used to calculate the maximum likelihood estimates. Most straightforward is to use standard software for fitting log-linear models, usually available as an option in routines for fitting generalized linear models. These procedures are based on an iteratively re-weighted least-squares algorithm, like that used in logistic regression, which derives from application of the Newton-Raphson algorithm. See, for example, McCullagh and Nelder [272, Ch. 2.5] for details. Standard output includes parameter estimates and approximate standard errors, where the latter are driven by the usual arguments for asymptotic normality of maximum likelihood estimators. We illustrate through an analysis of the Austrian call data of Example 9.1.

Example 9.2 (Analysis of Austrian Call Data). We consider two models for these data. The first is that stated in (9.9), while the second is that stated in (9.14), but with the term $\theta^T c_{ij}$ replaced by $-\theta \log(\text{Distance})$, where θ now is a non-negative scalar. We will refer to the former model as our standard gravity model and the latter as our general gravity model.

Both standard and general gravity models were fit using the generic iteratively re-weighted least-squares method for generalized linear models, having specified a Poisson model. Based on the standard asymptotic tests, all variables in both models were judged to be significant at the 0.05 level – in fact, the p -value for nearly every variable was beyond machine precision. The fit for the general gravity model had a much higher likelihood than that for the standard gravity model, which is not surprising, given that the former incorporates 64 variables, compared to four variables in the latter. However, the extent of this difference would appear to go well beyond that attributable simply to an increased number of parameters, judging by the fact that the Akaike information criterion (AIC) statistic³ is reduced by almost 50% percent.

Since prediction of traffic volume is typically of most interest in this setting, we give in Figure 9.2 two pairs of plots characterizing the accuracy of traffic volume estimates under each model. The first pair of plots shows the fitted values $\hat{\mu}_{ij}$ versus

² More precisely, we can write (9.14) in the form $\log(\mu) = \mathbf{M}\gamma$, where \mathbf{M} is an $(IJ) \times (I+J+K)$ matrix, and $\gamma = (\alpha_1, \dots, \alpha_I, \beta_1, \dots, \beta_J, \theta_1, \dots, \theta_K)^T$ is an $(I+J+K) \times 1$ vector. The first $I+J$ columns of \mathbf{M} are binary vectors, indicating the appropriate origin and destination for each entry of μ , and are redundant in that both the first I and the next J sum to the unit vector. The last K columns correspond to the K variables defining the c_{ij} . Assuming that the latter are linearly independent of themselves and of the former, the rank of \mathbf{M} will be $(I+J-1)+K$. See Sen and Smith [345, Ch. 5.2].

³ The AIC statistic for a likelihood-based model, with k -dimensional parameter η , is defined as $AIC = -2\ell(\hat{\eta}) + 2k$, where $\ell(\eta)$ is the log-likelihood evaluated at η , and $\hat{\eta}$ is the maximum likelihood estimate of η . This statistic, as with others of its type, provides an estimate of the generalization error associated with the fitted model, in this case effectively by off-setting the assessment of how well the model fits the data by a measure of its complexity. See, for example, Hastie, Tibshirani, and Friedman [194, Ch. 7.5] for additional details.

observed flow volumes z_{ij} . These plots are presented on a log-log scale, due to the large dynamic range of the values involved. The relationship between the two quantities is found to be fairly linear for both models, and the variation around their linear trend, fairly uniform. However, we note that there is evidence to suggest that the standard model tends to over-estimate in somewhat greater frequency than the general model, particularly for medium- and low-volume flows. The second pair of plots shows the relative errors $(z_{ij} - \hat{\mu}_{ij}) / z_{ij}$ versus the flow volumes z_{ij} , again on a log-log scale. We see that for both models the relative error varies widely in magnitude. A large proportion of the flows are estimated with an error on the order of z_{ij} or less (i.e., the logarithm of relative error is less than zero), but a substantial number are estimated with an error on the order of up to ten times z_{ij} , and few others are even worse. In addition, we can see that, roughly speaking, the relative error decreases with volume. Finally, it is clear that for low volumes both models are inclined to over-estimate, while for higher volumes, they are increasingly inclined to under-estimate.

A concise relative comparison of the estimation accuracy of our standard and general gravity models can be obtained by plotting the empirical cumulative distribution functions of their (log) relative prediction errors together, as in Figure 9.3. As the CDF for the general model lies to the left of that for the standard model over most of the range of the relative error, it is clear that the former dominates the latter in accuracy. In fact, the general model produces estimates on the order of z_{ij} or less for 72% of the origin-destination pairs, as compared to 58% under the standard model. For both models, the average magnitude of the relative error for these pairs is just under 0.30. \square

Various other algorithms have also been developed for maximum likelihood estimation in the general gravity model. A case of some particular interest is that where the functions $h_S(c_{ij})$ in (9.3) are replaced simply by values $h_S(i, j)$, and the latter are fixed and known. Only the parameters α_i and β_j then remain to be estimated. The method of *iterative proportional fitting* (IPF), usually credited to Deming and Stephan [113] in the statistics literature, and a standard tool in the statistical analysis of contingency tables, can be used here. Specifically, under mild conditions, estimates of the form $\hat{\mu}_{ij} = \hat{\alpha}_i \hat{\beta}_j h_S(i, j)$ satisfying the constraints in (9.16) can be obtained through iteration of the expressions

$$\mu_{ij}^{(2m-1)} = \frac{\mu_{ij}^{(2m-2)} z_{i+}}{\mu_{i+}^{(2m-2)}} \quad (9.18)$$

$$\mu_{ij}^{(2m)} = \frac{\mu_{ij}^{(2m-1)} z_{+j}}{\mu_{+j}^{(2m-1)}} , \quad (9.19)$$

starting with some initial set of values $\mu_{ij}^{(0)} = \hat{\alpha}_i^{(0)} \hat{\beta}_j^{(0)} h_S(i, j)$, with $\hat{\alpha}_i^{(0)}, \hat{\beta}_j^{(0)} > 0$. See Sen and Smith [345, Ch. 5.3], for example, for additional background and justification in the context of gravity models. More general discussions can be found

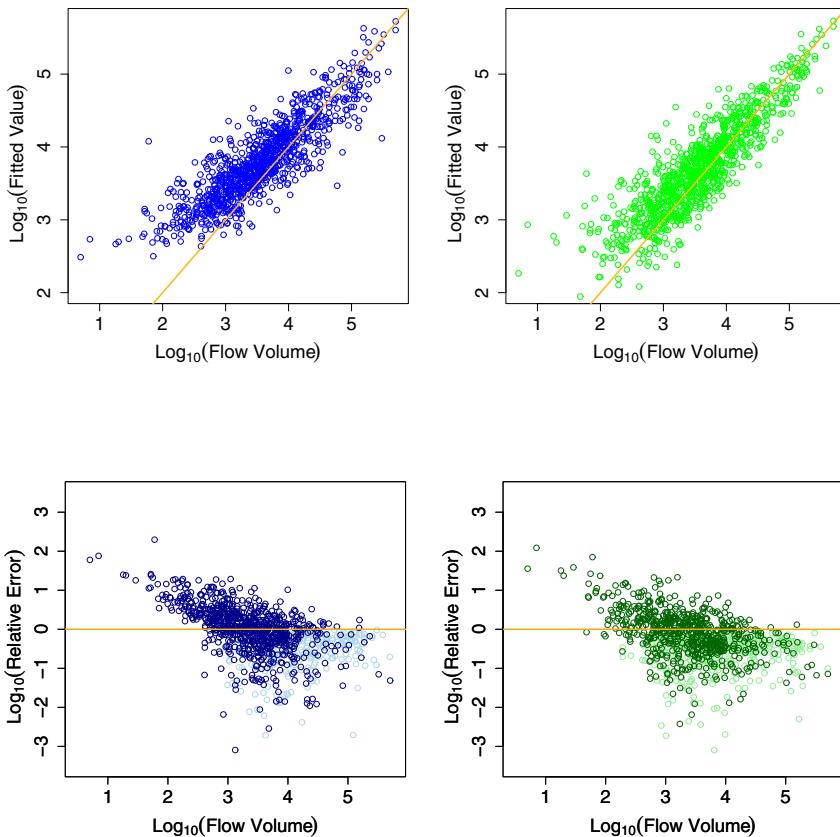


Fig. 9.2 Accuracy of estimates of traffic volume made by the standard (left, in blue) and general (right, in green) gravity models for the Austrian call data. Top: Fitted values versus flow volume. Bottom: Relative error versus flow volume, where light and dark points indicate under- and overestimation, respectively. All axes are on logarithmic scales, base ten. The lines $y = x$ and $y = 0$ are shown in yellow in the top and bottom sets of plots, respectively, for reference.

in most texts on log-linear models and the analysis of categorical data (e.g., Christensen [83, Ch. 3.3]).

The IPF algorithm can also be used for solving other related problems. For example, in the gravity model in (9.14), given a value for θ , such as an estimate $\hat{\theta}$ from past data, and anticipated values for the marginal flow totals Z_{i+} and Z_{+j} , the prediction of future values Z_{ij} can be approached using this algorithm. Similarly, the algorithm can be used within a larger, hybrid algorithm for obtaining maximum likelihood estimates of the α_i , β_j , and θ_k in (9.14). For instance, we can iterate between estimation of the α_i and β_j , on the one hand, and of θ , on the other, by calling the IPF algorithm for estimating α_i and β_j and a Newton-Raphson algorithm

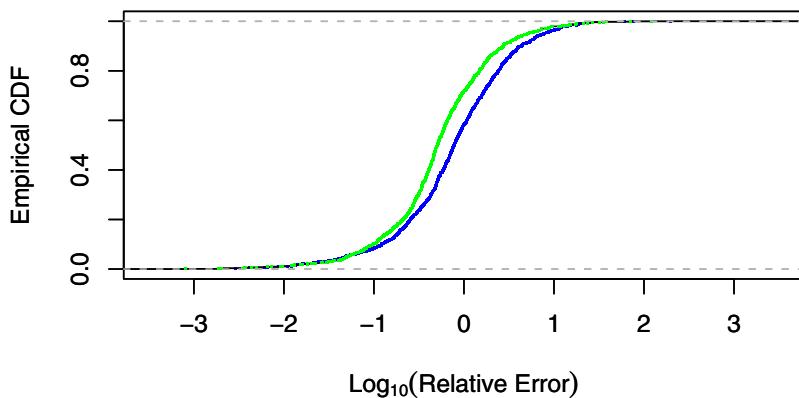


Fig. 9.3 Empirical CDF of the logarithm of the relative prediction error for the standard (blue) and general (green) gravity models in predicting call volume for the Austrian call data.

for estimating θ . Sen and Smith [345, Ch. 5] describe a number of algorithms of this nature. They also present evidence from a small numerical study that suggests such algorithms can be noticeably more computationally efficient than straightforward usage of standard generalized linear model fitting routines, particularly in problems with large I and J .

On a final note, it is also worth pointing out that procedures based on least squares can be used to obtain estimates for parameters in the general gravity model. Indeed, least-squares procedures were the norm for much of the history in this area, until the use of maximum-likelihood methods for generalized-linear models became computationally feasible and routine. Motivated by (9.3), least-squares procedures are generally based on models of the form

$$\log Z_{ij} \approx \alpha_i + \beta_j + \theta^T \mathbf{c}_{ij} + \varepsilon_{ij} , \quad (9.20)$$

where ε_{ij} is an error term. However, naive implementation of ordinary least-squares estimation is likely to give unsatisfactory results, particularly in settings where many of the means μ_{ij} are relatively small. Specifically, since $\mathbb{E}(\log Z_{ij}) \leq \log \mu_{ij}$, a bias can be expected in our estimates of the parameters. In addition, since $\mathbb{V}(\log Z_{ij})$ depends upon μ_{ij} , the usual assumption of constant variance underlying ordinary least squares is violated, and accompanying confidence statements and tests are suspect.

Standard corrective techniques from linear modeling can go some ways towards alleviating these problems. For example, we might replace the variables $\log(Z_{ij})$ on the left-hand side of (9.20) by $\tilde{Z}_{ij} = \log(Z_{ij} + 1/2)$, since it can be shown that $\mathbb{E}(\tilde{Z}_{ij})$ and $\mathbb{V}(\tilde{Z}_{ij})$ are equal to $\log \mu_{ij}$ and μ_{ij}^{-1} , respectively, up to terms of order $O(\mu_{ij}^{-2})$.

Application of a weighted least-squares procedure (e.g., Weisberg [399, Ch. 5.1]), with weights $w_{ij} \propto \mu_{ij}^{1/2}$, would then address the concerns about non-constant variance to a reasonable extent. Since these weights are not available, an iterative procedure can be used, starting for example with the observed $z_{ij}^{1/2}$ as weights and then using $\hat{\mu}_{ij}^{1/2}$ at the stages thereafter. Additionally, if our interest is focused primarily on the parameter θ , the overall dimension of the inference problem can be reduced by working not with the Z_{ij} , but rather with their centered versions $\tilde{Z}_{ij} - \tilde{Z}_{i\bullet} - \tilde{Z}_{\bullet j} + \tilde{Z}_{\bullet\bullet}$, where ‘•’ denotes averaging over the appropriate index.

Sen and Smith [345, Ch. 6] provide a detailed discussion of these topics. Least-squares techniques like these are still sometimes used for their simplicity. Nevertheless, all else being equal, it would seem preferable to use maximum likelihood techniques.

9.3 Traffic Matrix Estimation

Consider now the task of monitoring the flow volumes Z_{ij} on a network, with network graph G . In many types of networks, it is difficult – if not effectively impossible – to actually measure the Z_{ij} . For example, in monitoring road networks, various types of surveys are sometimes used (e.g., roadside surveys, home or destination surveys, etc.), but such devices can be highly inaccurate at small scales (e.g., due to small samples and sampling bias) and their large-scale usage is generally prohibitive in cost and effort. Furthermore, automated monitoring systems, based on collections of sensors, cameras, and the like, still face non-trivial technological hurdles prior to their successful large-scale deployment. On the other hand, in Internet networks carrying computer traffic, the network infrastructure itself can be equipped with the means to sample, store, and transmit (e.g., to a central location) such measurements. However, such capabilities are only relatively recent, and network service providers face some concern in enabling such measurements, in that they do wish to adversely affect the quality of service experienced by their customers.

Nevertheless, knowledge of the traffic flow volumes Z_{ij} is fundamental to a variety of network-oriented tasks. These include traffic management, network provisioning, and planning for network growth. Fortunately, in many of the same contexts in which measurement of the flow volumes Z_{ij} between origins and destinations is difficult, it is often relatively easy to measure the volumes X_e on network links. For example, in highway road networks, sensors may be positioned at the entrances to on- and off-ramps. Similarly, routers in an Internet network come equipped with the facility to monitor the data on incident links. In cases like these, we are then faced with a problem of predicting the Z_{ij} or, alternatively, estimating their means μ_{ij} , from the observed link counts $\mathbf{X} = (X_e)_{e \in E}$.

This is the *traffic matrix estimation* problem. It has received a great deal of attention in the transportation sciences literature, going back at least to the mid-1970’s, with seminal work by Robillard [327], van Zuylen and Willumsen [385], Bell [30],

and Cascetta [76]. Also, more recently, it has received substantial attention in the literature on computer network traffic analysis. The problem appears to have been formulated in this latter context, and at the same time introduced to the statistics literature, by Vardi [386], who dubbed it ‘network tomography.’

In any case of practical interest, the traffic matrix estimation problem will be highly under-constrained, in the sense that we effectively seek to invert the routing matrix \mathbf{B} in the relation $\mathbf{X} \approx \mathbf{BZ}$, and \mathbf{B} typically has many fewer rows (i.e., network links) than columns (i.e., origin-destination pairs). Various additional sources of information therefore typically are incorporated into the problem, which effectively serve to better constrain the set of possible solutions. Methods proposed in this area can be roughly categorized as *static* or *dynamic*, depending on whether they are aimed at estimating a traffic matrix for a single time period or successively over multiple time periods. We will discuss representative examples of both types.

Note that in what follows we assume that the routing matrix \mathbf{B} is known. In reality, its construction is a separate task in itself, the details of which vary by context. In computer traffic networks this task often can be done, for example, by processing tables of local routing protocols gathered from nodes throughout the network. Some discussion may be found in Crovella and Krishnamurthy [105, Ch. 5.3.2]. In contrast, in the field of transportation studies, the entries of the routing matrix typically are based on models, often fit to data, of human behavior. Ortúzar and Willumsen [306] provide a detailed introduction.

9.3.1 Static Methods

The various static methods proposed for traffic matrix estimation are similar in that they all ultimately involve the optimization of some objective function, usually subject to certain constraints. However, they can differ widely in the construction of and justification for this objective function and, to a lesser extent, the constraints imposed. Here we discuss three classes of methods, deriving from principles of least squares and Gaussian models, from Poisson models, and from principles of entropy minimization.

9.3.1.1 Methods Based on Least-Squares and Gaussian Models

Although traffic typically is discussed in units of ‘counts’ (e.g., number of vehicles per hour or number of bytes per second), some of the earliest proposed methods were based on least-squares and Gaussian measurement models. Much insight into the traffic matrix estimation problem can be gained from the least-squares perspective, and the Gaussian measurement model that accompanies least squares (whether it be made explicit or not) is often not unreasonable in high-count settings. So we begin our discussion within this context.

A simple but commonly adopted model for the link counts \mathbf{X} is one of the form

$$\mathbf{X} = \mathbf{B}\mu + \boldsymbol{\varepsilon} , \quad (9.21)$$

where $\mathbf{X} = (X_e)_{e \in E}$ is an $N_e \times 1$ vector of link counts, \mathbf{B} is an $N_e \times IJ$ routing matrix, as defined in (9.1), μ is an $IJ \times 1$ vector of expected flow volumes over all origin destination pairs $(i, j) \in \mathcal{I} \times \mathcal{J} \subseteq V^{(2)}$, and $\boldsymbol{\varepsilon}$ is an $N_e \times 1$ vector of errors, which we assume for the moment to be independent with mean zero and common variance σ^2 .

In principle, this formulation suggests that μ be estimated through ordinary least squares. However, typically $N_e \ll IJ$, and so the least-squares problem is poorly posed. More precisely, the vector μ is generally not estimable, in the sense that there is no matrix \mathbf{M} such that $\mathbb{E}(\mathbf{MX}) = \mu$. In fact, this issue can arise in even the smallest of systems, as illustrated by the following example.

Example 9.3 (Traffic Matrix Estimation on a Small Network (Robillard [327])). Figure 9.4 shows a simple network, consisting of five vertices and four links. There are two origin vertices (i.e., a and b), two destination vertices (i.e., c and d), and one intermediate vertex (i.e., v), resulting in the set $\{ac, ad, bc, bd\}$ of origin-destination pairs. In this case, the model (9.21) reduces to

$$\begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} \mu_{ac} \\ \mu_{ad} \\ \mu_{bc} \\ \mu_{bd} \end{bmatrix} + \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \varepsilon_3 \\ \varepsilon_4 \end{bmatrix} . \quad (9.22)$$

Although here $N_e = IJ = 4$, the rank of \mathbf{B} is nevertheless only 3, and therefore $\mathbf{B}^T \mathbf{B}$ is not invertible. As a result, for observations $\mathbf{X} = \mathbf{x}$, the normal equations $(\mathbf{B}^T \mathbf{B})\mu = \mathbf{B}^T \mathbf{x}$ corresponding to the least-squares problem

$$\min_{\mu} (\mathbf{x} - \mathbf{B}\mu)^T (\mathbf{x} - \mathbf{B}\mu) \quad (9.23)$$

will generally have an infinite number of possible solutions $\hat{\mu}$. \square

Although μ generally is not estimable in this model, under certain conditions the expected origin and destination volumes μ_{i+} and μ_{+j} are in fact estimable, and any solution $\hat{\mu}$ of the least-squares problem will produce the same estimates $\hat{\mu}_{i+}$ and $\hat{\mu}_{+j}$. Robillard [327] proposes using a gravity model for the expected flow volumes μ_{ij} . Specifically, he proposes a model in the form $\log \mu_{ij} = \alpha_i + \beta_j + \log \gamma_{ij}$, for pre-specified gravity constants γ_{ij} , and suggests estimating the α_i and β_j under the constraints that $\mu_{i+} = \hat{\mu}_{i+}$ and $\mu_{+j} = \hat{\mu}_{+j}$, using a nonlinear least-squares algorithm. The resulting solution $\hat{\mu}$ will be unique if the values $\hat{\mu}_{i+}, \hat{\mu}_{+j}$, and γ_{ij} are all positive.

Unfortunately, it has been observed that in practice gravity models often fit too poorly to produce good estimates $\hat{\mu}$. However, in some situations we have available some initial – although perhaps rough and inaccurate – set of origin-destination flow volume measurements, say $\mathbf{Z}^{(0)} = \mathbf{z}^{(0)}$. In this case, we might use these measurements, rather than a gravity model, to suitably constrain our estimate $\hat{\mu}$. Cascetta [76] proposes a method for doing so based on generalized least-squares.

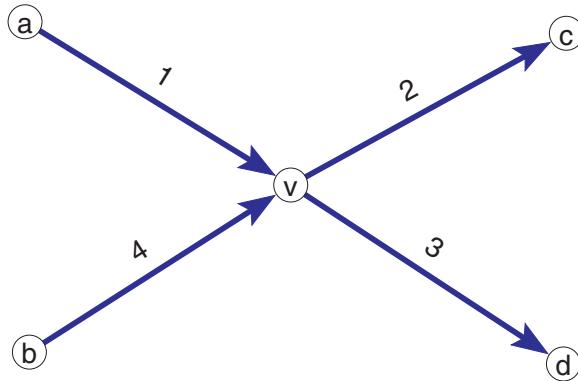


Fig. 9.4 A simple network illustrating the traffic matrix estimation problem.

Specifically, consider the model

$$\begin{bmatrix} \mathbf{Z}^{(0)} \\ \mathbf{x} \end{bmatrix} = \begin{bmatrix} \mathbf{I} \\ \mathbf{B} \end{bmatrix} \mu + \begin{bmatrix} \xi \\ \varepsilon \end{bmatrix}, \quad (9.24)$$

where \mathbf{I} is the $IJ \times IJ$ identity matrix, and ξ and ε are independent error vectors of dimension $IJ \times 1$ and $N_e \times 1$, respectively, with mean zero and covariance matrices Ψ and Σ , respectively. The solution to the corresponding generalized least-squares problem

$$\min_{\mu} \begin{bmatrix} \mathbf{z}^{(0)} - \mu \\ \mathbf{x} - \mathbf{B}\mu \end{bmatrix}^T \begin{bmatrix} \Psi^{-1} & 0 \\ 0 & \Sigma^{-1} \end{bmatrix} \begin{bmatrix} \mathbf{z}^{(0)} - \mu \\ \mathbf{x} - \mathbf{B}\mu \end{bmatrix} \quad (9.25)$$

takes the form

$$\hat{\mu} = (\Psi^{-1} + \mathbf{B}^T \Sigma^{-1} \mathbf{B})^{-1} (\Psi^{-1} \mathbf{z}^{(0)} + \mathbf{B}^T \Sigma^{-1} \mathbf{x}), \quad (9.26)$$

which is a linear combination of $\mathbf{z}^{(0)}$ and \mathbf{x} . Following standard linear model theory, under the model assumptions the estimator (9.26) is unbiased (i.e., $\mathbb{E}(\hat{\mu}) = \mu$) and its covariance $\mathbb{V}(\hat{\mu}) = (\Psi^{-1} + \mathbf{B}^T \Sigma^{-1} \mathbf{B})^{-1}$ is a minimum among all unbiased estimators.

The covariance may be used to accompany the estimator $\hat{\mu}$ with some quantification of the uncertainty involved in estimating μ , the exact nature of which is dictated

through the error covariances Ψ and Σ . In practice, Σ is often taken to be a diagonal matrix and the form of Ψ will depend on the nature of the sampling underlying the measurements $\mathbf{z}^{(0)}$. Cascetta and Nguyen [77, Sec. 2.2] offer some discussion on this topic, for example. The values of the entries in Σ and Ψ are often set using historical data (i.e., earlier sets of measurements $\mathbf{z}^{(0)}$) or previous estimates $\hat{\mu}$.

Note that it is possible – and, indeed, likely, in the case of low link volume counts – for the estimated expected flow volumes $\hat{\mu}_{ij}$ in (9.26) to be negative. Cascetta [76], and later Bell [31] in more detail, derive an analogous generalized least-squares estimate of μ under the constraint that $\mu_{ij} \geq 0$, for all pairs i, j . The result of incorporating these constraints is effectively to perturb the estimate $\hat{\mu}$ by adding another term to the expression in (9.26). The contribution of this term is increasingly more pronounced the more active these constraints become.

Although no distributional assumptions are necessary for the generalized least-squares framework above, from a likelihood-based perspective a Gaussian noise model is implicit in its formulation. Maher [267], adopting such a perspective, has proposed a Bayesian approach to the estimation of μ , based on the use of both a Gaussian measurement model and Gaussian prior. To illustrate, consider the measurement model in (9.21), and suppose we specify in addition that

$$\mu = \mu^{(0)} + \eta , \quad (9.27)$$

where $\mu^{(0)}$ is an $IJ \times 1$ prior mean vector for μ and η is an $IJ \times 1$ random vector, with independent components of mean zero and prior variance τ^2 . If it is further assumed that ε and η have independent multivariate Gaussian distributions, then it follows that the posterior distribution of μ , given observations $\mathbf{X} = \mathbf{x}$, is itself multivariate Gaussian with mean

$$\mathbb{E}(\mu | \mathbf{X} = \mathbf{x}) = \mu^{(0)} + \mathbf{B}^T (\mathbf{B}\mathbf{B}^T + \lambda I)^{-1} (\mathbf{x} - \mathbf{B}\mu^{(0)}) \quad (9.28)$$

and covariance matrix

$$\mathbb{V}(\mu | \mathbf{X} = \mathbf{x}) = \tau^2 \left[I - \mathbf{B}^T (\mathbf{B}\mathbf{B}^T + \lambda I)^{-1} \mathbf{B} \right] , \quad (9.29)$$

where $\lambda = \sigma^2 / \tau^2$. Maher [267] derives the analogous expressions under an assumption of general covariances for ε and η . The form of the resulting posterior mean parallels that of the generalized least-squares estimator (9.26).

Note that the form of $\hat{\mu}$ in (9.28) is such that it may be interpreted as starting with the prior mean $\mu^{(0)}$ and then ‘correcting’ it through a term driven by the error in the ‘prediction’ of \mathbf{x} by $\mathbf{B}\mu^{(0)}$. The covariance matrix in (9.29) can be used to fashion corresponding statements regarding the uncertainty of this estimate (e.g., point-wise posterior credible intervals for each μ_{ij}). The value λ acts as a smoothing parameter. In the case that $\lambda \rightarrow 1$, the estimate $\hat{\mu}$ behaves so that $\mathbf{B}\hat{\mu} \rightarrow \mathbf{x}$ (i.e., an equality constraint with the observed link volumes is enforced). On the other hand, as $\lambda \rightarrow 0$, the estimate $\hat{\mu}$ tends to the prior mean $\mu^{(0)}$.

9.3.1.2 Methods Based on Poisson Models

As was noted above, the flow volumes Z_{ij} typically are discussed in units of counts. While a Gaussian measurement model may be reasonable in high-count settings, it will likely not be appropriate if even just some of the expected flow volumes μ_{ij} are low. In this latter case, a Poisson model can be more appropriate.

Specifically, consider modeling the distribution of \mathbf{Z} , given μ , as a product of independent Poisson random variables i.e.,

$$\mathbb{P}(\mathbf{Z} = \mathbf{z} | \mu) = \prod_{ij} \mathbb{P}(Z_{ij} = z_{ij} | \mu_{ij}) = \prod_{ij} \frac{\exp(-\mu_{ij}) \mu_{ij}^{z_{ij}}}{z_{ij}!}. \quad (9.30)$$

Furthermore, suppose that $\mathbf{X} = \mathbf{BZ}$, indicating that no additional measurement error is incurred in observing the link volumes. The distribution of \mathbf{X} is therefore induced directly by that of \mathbf{Z} , and in general the elements of \mathbf{X} will not be independent. Note too that there is, of course, the same (usually substantial) reduction in dimension from \mathbf{Z} to \mathbf{X} as was observed at the beginning of the previous section on least squares methods, so that typically there are many values of \mathbf{z} that might solve the equations $\mathbf{x} = \mathbf{Bz}$, given observed link counts $\mathbf{X} = \mathbf{x}$.

It is perhaps somewhat surprising then that distinct values of μ will nevertheless yield distinct distributions for \mathbf{X} under fairly general conditions on \mathbf{B} . Specifically, Vardi [386] shows that if the columns of \mathbf{B} are all distinct, and each column has at least one non-zero element, then $\mathbb{P}(\mathbf{X} = \mathbf{x} | \mu) = \mathbb{P}(\mathbf{X} = \mathbf{x} | \tilde{\mu})$ implies that $\tilde{\mu}$ differs from μ by at most a simple rescaling.⁴ In principle, it would therefore seem that we can proceed in a standard manner with maximum likelihood estimation of the relative magnitudes of the μ_{ij} . However, potential difficulties still remain, as illustrated by the following toy example.

Example 9.4 (Difficulties with Poisson Maximum Likelihood (Vardi [386])). Recall the toy network of Example 9.3, depicted in Figure 9.4, and consider just that sub-network consisting of the three vertices $\{a, v, c\}$ and the two links 1 and 2. We will consider this network as having a total of three origin-destination pairs $\{av, vc, ac\}$. Its routing matrix is of the form

$$\mathbf{B} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}. \quad (9.31)$$

Assume model (9.30) and suppose that we observe the link volumes $\mathbf{x} = (x_1, x_2)^T = (1, 2)^T$. There are only two possible values for the corresponding flow volumes in this case: $\mathbf{z} = (0, 1, 1)^T$ and $\mathbf{z} = (1, 2, 0)^T$. The likelihood of the data can therefore be expressed as

$$\mathcal{L}(\mu) = \mathbb{P}(\mathbf{X} = (1, 2)^T)$$

⁴ That is, under these conditions, the parameter vector μ is identifiable. See also Singhal and Michailidis [354].

$$\begin{aligned}
&= \mathbb{P}(\mathbf{Z} = (0, 1, 1)^T) + \mathbb{P}(\mathbf{Z} = (1, 2, 0)^T) \\
&= (\mu_{ac}\mu_{vc} + \mu_{av}\mu_{vc}^2/2) \exp(-\mu_{ac} - \mu_{av} - \mu_{vc}) .
\end{aligned} \tag{9.32}$$

The maximum likelihood estimate $\hat{\mu}$ of μ will be the solution to the problem

$$\max_{\mu \geq 0} \ell(\mu) , \tag{9.33}$$

where $\ell(\mu) = \log \mathcal{L}(\mu)$ is the log-likelihood function. Straightforward calculations show that the likelihood score equations $\partial \ell(\mu)/\partial \mu = 0$ have a unique solution $\mu = (1, 2, 0)^T$. But, in fact, it can be shown that the value of μ that maximizes the likelihood in (9.32) is instead $\mu = (0, 1, 1)^T$. So the maximum likelihood solution differs from that which we would obtain through standard techniques, based on the likelihood score function. \square

The seeming paradox in this example illustrates the difficulty, under the specified Poisson model, of maximum likelihood estimation in the traffic estimation problem. This difficulty arises due to the possibility of solutions to the score equations at the boundary of the parameter space (i.e., where some subset of the elements of μ take on the value zero). Vardi further shows that the log-likelihood $\ell(\mu)$ is, in general, not even necessarily concave. He argues, however, that if there are multiple independent and identically distributed samples $\mathbf{x}_1, \dots, \mathbf{x}_n$, the likelihood is asymptotically concave in the limit of large n , assuming all elements of μ are strictly positive. Approximating this likelihood by that of a multivariate Gaussian, with mean $\mathbf{B}\mu$ and covariance $\mathbf{B}\mathbf{M}\mathbf{B}^T$, where $M = \text{diag}(\mu)$, he proposes a method-of-moments estimator, where these quantities are set equal to their empirical counterparts. But some care must be exercised in solving the resulting equations for $\hat{\mu}$, due to the possibility of algebraic inconsistencies. See Vardi [386, Sec. 3–4] for discussion and some proposed solutions.

Alternatively, the Poisson likelihood can be coupled with a prior distribution on μ , and inference on μ and \mathbf{Z} can be pursued through Bayesian methods. Similar to the Gaussian setting, incorporating an appropriately designed prior can have the effect of regularizing the traffic matrix estimation problem in the Poisson setting, which can lead to more stable solutions. In addition, the use of prior information – whether implicitly or explicitly – is almost obligatory when only a single sample $\mathbf{X} = \mathbf{x}$ is available.

Tebaldi and West [377] explore the Bayesian approach in some detail, where their goal is to conduct inference based on the joint posterior distribution $\mathbb{P}(\mathbf{Z}, \mu | \mathbf{X})$. In particular, they concentrate on the problem of simulating from this posterior using Markov chain Monte Carlo methods. Note that under the choice of independent priors on the components of μ , $\mathbb{P}(\mu) = \prod_{ij} \mathbb{P}(\mu_{ij})$, and we have

$$\mathbb{P}(\mu | \mathbf{X}, \mathbf{Z}) = \mathbb{P}(\mu | \mathbf{Z}) = \prod_{ij} \mathbb{P}(\mu_{ij} | Z_{ij}) . \tag{9.34}$$

Therefore, given \mathbf{Z} , it should be reasonably straightforward to simulate values of μ . So consider the conditional posterior distribution $\mathbb{P}(\mathbf{Z} | \mu, \mathbf{X})$. Since the system

of equations $\mathbf{X} = \mathbf{BZ}$ serves as a set of linear constraints on \mathbf{Z} , given $\mathbf{X} = \mathbf{x}$, these constraints must be incorporated explicitly when simulating from this posterior. The following example serves to illustrate this point.

Example 9.5 (Constraints in the Conditional Posterior (Tebaldi and West [377])). Recall the three-vertex network underlying Example 9.4, with routing matrix \mathbf{B} given in (9.31). Since $X_1 = Z_{av} + Z_{ac}$ and $X_2 = Z_{vc} + Z_{ac}$, it is clear that knowledge of $Z_{ac} = z_{ac}$, in addition to $\mathbf{X} = \mathbf{x}$, is sufficient to recover z_{av} and z_{vc} . Furthermore, under the assumption of independent prior distributions on the elements of μ , $\mathbb{P}(\mu_{av}, \mu_{vc}, \mu_{ac}) = \mathbb{P}(\mu_{av})\mathbb{P}(\mu_{vc})\mathbb{P}(\mu_{ac})$, and the marginal posterior distribution of Z_{ac} can be expressed in the form⁵

$$\begin{aligned}\mathbb{P}(Z_{ac} = z_{ac} | \mu, \mathbf{X} = \mathbf{x}) &\propto \mathbb{P}(Z_{ac} = z_{ac} | \mu_{ac}) \times \mathbb{P}(Z_{av} | \mu_{av}) I_{x_1 - z_{ac}}(Z_{av}) \\ &\quad \times \mathbb{P}(Z_{vc} | \mu_{vc}) I_{x_2 - z_{ac}}(Z_{vc}) ,\end{aligned}\quad (9.35)$$

where $I_s(r)$ is the indicator that $r = s$, for fixed s .

As a result of these two facts, it follows that

$$\mathbb{P}(Z_{ac} = z_{ac} | \mu, \mathbf{X} = \mathbf{x}) \propto \frac{\mu_{ac}^{z_{ac}}}{z_{ac}!} \frac{\mu_{av}^{x_1 - z_{ac}}}{(x_1 - z_{ac})!} \frac{\mu_{vc}^{x_2 - z_{ac}}}{(x_2 - z_{ac})!} , \quad (9.36)$$

for $z_{ac} \in \{0, 1, \dots, \min(x_1, x_2)\}$. Therefore, we can simulate from the full joint conditional posterior $\mathbb{P}(\mathbf{Z} | \mu, \mathbf{X} = \mathbf{x})$ in this problem by first drawing $Z_{ac} = z_{ac}$, according to (9.36) and then setting $z_{av} = x_1 - z_{ac}$ and $z_{vc} = x_2 - z_{ac}$. \square

More generally, if \mathbf{B} is of full rank N_e , we can re-order its columns so that $\mathbf{B} = [\mathbf{B}_1, \mathbf{B}_2]$, where \mathbf{B}_1 is a $N_e \times N_e$ non-singular matrix, and we can similarly write $\mathbf{Z}^T = (\mathbf{Z}_1^T, \mathbf{Z}_2^T)$, where $\mathbf{Z}_1 = \mathbf{B}_1^{-1}(\mathbf{X} - \mathbf{B}_2\mathbf{Z}_2)$. It then follows that the conditional posterior has the form

$$\mathbb{P}(\mathbf{Z} = \mathbf{z} | \mu, \mathbf{X} = \mathbf{x}) = \mathbb{P}(\mathbf{Z}_1 = \mathbf{z}_1 | \mathbf{Z}_2 = \mathbf{z}_2, \mu, \mathbf{X} = \mathbf{x}) \mathbb{P}(\mathbf{Z}_2 = \mathbf{z}_2 | \mu, \mathbf{X} = \mathbf{x}) , \quad (9.37)$$

where the distribution $\mathbb{P}(\mathbf{Z}_1 = \mathbf{z}_1 | \mathbf{Z}_2 = \mathbf{z}_2, \mu, \mathbf{X} = \mathbf{x})$ is concentrated at $\mathbf{z}_1 = \mathbf{B}_1^{-1}(\mathbf{x} - b\mathbf{B}_2\mathbf{z}_2)$ and

$$\mathbb{P}(\mathbf{Z}_2 = \mathbf{z}_2 | \mu, \mathbf{X} = \mathbf{x}) \propto \prod_{ij} \frac{\mu_{ij}^{z_{ij}}}{z_{ij}!} . \quad (9.38)$$

Compare this expression to (9.36).

Tebaldi and West [377, Sec. 3] describe how to sample from the distribution in (9.38) using Gibbs sampling,⁶ with particular attention given to designing efficient

⁵ Technically, this expression should also distinguish between the random variable μ and a realization of that random variable, analogous to the manner in which we distinguish between \mathbf{Z} and \mathbf{X} , on the one hand, and \mathbf{z} and \mathbf{x} , on the other. We suppress this detail here and immediately below, however, as it is cumbersome and the role of μ is secondary.

⁶ See the discussion in Section 8.3.2.2 and the references therein.

sampling strategies. Inference for the individual elements in \mathbf{Z} or μ then follows from the resulting Monte Carlo approximations to the appropriate marginal distributions $\mathbb{P}(Z_{ij} | \mathbf{X})$ and $\mathbb{P}(\mu_{ij} | \mathbf{X})$. In applications, these authors choose to use prior distributions $\mathbb{P}(\mu_{ij})$ that either are uniform on some interval $[0, m]$ or that follow a gamma distribution. Although simple and convenient, these choices appear to yield reasonable results in the examples studied.

Additional, earlier work on Bayesian approaches in the Poisson setting is surveyed in Cascetta and Nguyen [77], where some alternative choices of prior distributions are discussed and where it is sketched how inference on μ may be carried out in some cases through direct numerical optimization of the posterior $\mathbb{P}(\mu | \mathbf{X})$.

9.3.1.3 Methods Based on Entropy Minimization

Proposed in a somewhat different vein than the methods of the two classes above are a handful of methods based on the minimization of a particular form of entropy. Let μ be the vector of expected origin-destination flow volumes and $\mu^{(0)}$, a prior ‘guess’ for μ , and suppose that the two have been normalized so that $\mu_{++} = \mu_{++}^{(0)}$. The relative entropy ‘distance’ between μ and $\mu^{(0)}$ is given by

$$D(\mu || \mu^{(0)}) = \sum_{ij} \frac{\mu_{ij}}{\mu_{++}} \log \frac{\mu_{ij}}{\mu_{ij}^{(0)}} . \quad (9.39)$$

This quantity is also known as the Kullback-Liebler divergence and summarizes how similar the two ‘distributions’ $\{\mu_{ij}/\mu_{++}\}$ and $\{\mu_{ij}^{(0)}/\mu_{++}^{(0)}\}$ are to each other. It is always non-negative and will be equal to zero if and only if $\mu = \mu^{(0)}$. See, for example, Cover and Thomas [101, Ch. 2.3].

Entropy-based methods seek to minimize the expression in (9.39), subject to certain constraints that encourage fidelity to the observed data (i.e., $\mathbf{x} \approx \mathbf{B}\mu$). For example, an early proposal in this area, due to van Zuylen and Willumsen [385], effectively seeks to minimize (9.39) subject to the constraints $X_e = \sum_{ij} B_{e;ij} \mu_{ij}$, for $e \in E$. Formally, through the method of Lagrange multipliers, this problem is equivalent to the problem

$$\min_{\mu, \lambda_1, \dots, \lambda_{N_e}} D(\mu || \mu^{(0)}) + \sum_{e=1}^{N_e} \lambda_e \left(\sum_{ij} B_{e;ij} \mu_{ij} - x_e \right) , \quad (9.40)$$

where $\lambda_1, \dots, \lambda_{N_e}$ are the multipliers introduced to enforce the N_e constraints.

Differentiating the expression in (9.40) with respect to the μ_{ij} and setting the result to zero, we find that for fixed $\lambda = (\lambda_1, \dots, \lambda_{N_e})$, the elements of the resulting estimator have the form

$$\hat{\mu}_{ij}(\lambda) = \mu_{ij}^{(0)} \exp \left(-1 - \sum_{e \in E} \lambda_e B_{e;ij} \right) . \quad (9.41)$$

So the solution $\hat{\mu}$ is a multiplicative perturbation of the initial $\mu^{(0)}$. In practice, the values in λ are found using an iterative optimization algorithm starting, say, from $\lambda = \mathbf{1}$. The value $\mu^{(0)}$ is generally specified based on previous estimates or initial measurements $\mathbf{z}^{(0)}$. Note that a non-negative solution $\hat{\mu}$ is guaranteed if $\mu^{(0)} \geq 0$.

More recently, Zhang, Roughan, Lund, and Donoho [414] have instead proposed to optimize the relative entropy (9.39) subject to the constraint $(\mathbf{x} - \mathbf{B}\mu)^T(\mathbf{x} - \mathbf{B}\mu) \leq C$, for some constant $C > 0$, and $\mu \geq 0$. That is, the constraints of van Zuylen and Willumsen, with their strict equality, are relaxed to a single constraint on the overall accuracy of the prediction of \mathbf{x} by $\mathbf{B}\mu$. Formally, this optimization is equivalent to the problem

$$\min_{\mu \geq 0} (\mathbf{x} - \mathbf{B}\mu)^T(\mathbf{x} - \mathbf{B}\mu) + \lambda D(\mu || \mu^{(0)}) . \quad (9.42)$$

The expression in (9.42) is seen to be a penalized least-squares problem, like that in (2.39) of Chapter 2. Methods of this type are common in the literature on ill-posed inverse problems, where they are commonly referred to as *maximum entropy regularization*. Alternatively, interpreted from a Bayesian perspective, the estimator (9.42) is simply a maximum *a posteriori* estimate of μ , under the assumption of the Gaussian measurement model in (9.21) and a prior $f(\mu)$, where $\log f(\mu)$ behaves like $D(\mu || \mu^{(0)})$. This particular choice of prior can be viewed as an approximation to a multinomial prior for μ , with probabilities proportional to the values $\mu_{ij}^{(0)}$. See Cascetta and Nguyen [77, Sec. 2.3].

As the relative entropy is convex in its argument μ , the overall optimization is a convex optimization problem, for which existing software may be used, as Zhang, Roughan, Lund, and Donoho [414] discuss. These authors implement a particular version of this estimation strategy, under the additional specification that $\mu_{ij}^{(0)}$ follows a certain gravity model. We illustrate below, in the case study of Section 9.3.3, with a simple gravity model of the form $\mu_{ij}^{(0)} \propto \mu_{i+}^{(0)} \mu_{+j}^{(0)}$, where $\mu_{i+}^{(0)}$ and $\mu_{+j}^{(0)}$ are origin- and destination-specific net out-flow and in-flow volumes, respectively. In practice, there is also the issue of choosing the smoothing parameter λ in (9.42). Cross-validation, for example, may be used for this purpose. Zhang et al., however, report that their results are relatively insensitive to the value of λ and suggest that such careful tuning may be unnecessary.

9.3.2 Dynamic Methods

Suppose now that we are interested in recovering traffic matrices \mathbf{Z} , or their means μ , over a sequence of consecutive time periods $t = 1, \dots, \tau$. Specifically, given vectors $\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(\tau)}$ of observed link counts, we wish to infer the corresponding traffic matrices $\mathbf{z}^{(1)}, \dots, \mathbf{z}^{(\tau)}$ or their means $\mu^{(1)}, \dots, \mu^{(\tau)}$, where $\mathbf{x}^{(t)} \approx \mathbf{B}^{(t)} \mathbf{z}^{(t)}$, for possibly time-dependent routing matrices $\mathbf{B}^{(t)}$. Dynamic methods of traffic matrix estimation are designed for this purpose and may be categorized roughly according to whether they produce inferences simultaneously at all time periods or sequen-

tially. Simultaneous and sequential methods also are sometimes referred to as non-recursive and recursive, respectively. An overview of work in this area, from the field of transportation studies, may be found in Sherali and Park [349]. Similarly, Soule et al. [364] is a useful reference from the field of computer network traffic analysis.

The dynamic methods proposed to date seem to be predominantly based on principles of least squares. In addition, the majority of the methods require that the length of a typical trip time, from any given origin to any given destination, be substantially shorter than the length of each time interval t during which measurements were taken. This assumption has the advantage of simplifying the nature of the routing information that must be encoded in the routing matrices $\mathbf{B}^{(t)}$, since it effectively allows us to ignore the possibility that trips beginning in one time period actually end in a different time period. See Sherali and Park [349] for further discussion on this point.

Simultaneous methods of inference typically work from a model of the form $\mathbf{x}^{(t)} = \mathbf{B}^{(t)}\mu^{(t)} + \varepsilon^{(t)}$, in analogy to (9.21), and penalized least-squares criteria are often employed to estimate the values $\mu^{(t)}$. That is, we estimate the vector of means $(\mu^{(1)^T}, \dots, \mu^{(\tau)^T})^T$ as

$$\begin{aligned} (\hat{\mu}^{(1)^T}, \dots, \hat{\mu}^{(\tau)^T})^T &= \arg \min \quad \sum_{t=1}^{\tau} \left(\mathbf{x}^{(t)} - \mathbf{B}^{(t)}\mu^{(t)} \right)^T \left(\mathbf{x}^{(t)} - \mathbf{B}^{(t)}\mu^{(t)} \right) \\ &\quad + J \left[\left(\mu^{(1)^T}, \dots, \mu^{(\tau)^T} \right) \right], \end{aligned} \quad (9.43)$$

where the minimization is over all values of the collection of τ vectors $\mu^{(t)}$, possibly with the constraint that they have only non-zero elements. The penalty $J(\cdot)$ is needed because, based on the squared-error loss alone, estimation of the full collection of means $\mu^{(t)}$ reduces to τ separate ordinary least-squares sub-problems, and individually each of these sub-problems will of course be under-determined, as in the static case. A simple but common specification of the penalty is as a sum $J = \sum_{t=1}^{\tau} J^{(t)}$, where the $J^{(t)}$ are, for example, based on independent Gaussian priors (9.27), or entropy-based priors (9.39), say with $\mu^{(0)}$ replaced by time-indexed analogues $\mu^{(t,0)}$. In this case, the optimization in (9.43) then reduces to τ separate static inferences. See Cascetta [75], for example.

Note that from this perspective, any temporal correlations in the measurements $\mathbf{x}^{(t)}$ are effectively ignored. If we impose the additional assumption that the means $\mu^{(t)}$ are constant (i.e., that $\mu^{(t)} \equiv \mu$, for some μ), then the $\mathbf{x}^{(t)}$ may effectively be treated as replicates. Although this assumption is unlikely to be true when the length of the overall period of measurement is long, it can sometimes be a useful approximation over sufficiently short periods. Of course, in the least-squares setting, even with replicates the estimation of μ is not generally a well-posed problem. However, as was mentioned in Section 9.3.1.2, this estimation problem is well posed under the Poisson model, and for sufficiently large τ the method-of-moments estimator of Vardi [386] can be used. Similarly, Cao, Davis, Wiel, and Yu [73] show that the

estimation problem can also be well posed when the $\mathbf{x}^{(t)}$ are modeled according to a multivariate Gaussian distribution, with mean μ and covariance $\mathbf{B}\mathbf{M}\mathbf{B}^T$, where $\mathbf{M} = \text{diag}(\mu_{ij}^c)$, for some $c > 0$. Cao et al. propose a maximum likelihood estimator that can be interpreted as a weighted least-squares estimator, where the weights depend on the unknown elements μ_{ij} .

Sequential methods of traffic matrix estimation differ from simultaneous methods in that, not only do they try to account for correlations in the measurements $\mathbf{x}^{(t)}$ over time, they in fact seek to exploit these correlations. Most frameworks of this type can be viewed as variations on or extensions of the concept of Kalman filtering, as described in, for example, Nihan and Davis [303] and Soule, Salamatian, Nucci, and Taft [365]. Our description here parallels that of the latter authors.

In the Kalman filtering approach, the time-varying relationship among the means $\mu^{(t)}$ and the link counts $\mathbf{X}^{(t)}$ is modeled through a set of equations

$$\mu^{(t+1)} = \Xi^{(t)}\mu^{(t)} + \eta^{(t)} \quad (9.44)$$

$$\mathbf{X}^{(t)} = \mathbf{B}^{(t)}\mu^{(t)} + \varepsilon^{(t)}, \quad (9.45)$$

where the $\eta^{(t)}$ and the $\varepsilon^{(t)}$ are assumed to be zero-mean random vectors, with covariances $\Psi^{(t)}$ and $\Sigma^{(t)}$, respectively, and uncorrelated both with each other and among themselves across times t . The stochastic process defined in (9.45) is called the *observation process*, and that in (9.44), the *hidden process* or the *state process*. The matrices $\Xi^{(t)}$ govern the nature of the evolution of the hidden process.

The so-called ‘Kalman filter’ is a sequential, recursive algorithm for determining, at each time $t+1$, an optimal estimate of the state $\mu^{(t+1)}$, based on the observations $\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(t+1)}$, where this estimate is optimal in the sense that it is unbiased and has minimum variance among all unbiased estimators.⁷ Specifically, the Kalman filter algorithm alternates between making a prediction $\hat{\mu}^{t+1|t}$ of $\mu^{(t+1)}$, based on the first t observations, $\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(t)}$, and then updating (or ‘correcting’) that prediction to a value $\hat{\mu}^{t+1|t+1}$, to adjust for the discrepancy between the observation $\mathbf{x}^{(t+1)}$ and its predicted value $\mathbf{B}^{(t+1)}\hat{\mu}^{t+1|t}$. This second value $\hat{\mu}^{t+1|t+1}$ is what we use to define our estimate $\hat{\mu}^{(t+1)}$.

In more detail, let $\mathbf{P}^{t|t} = \mathbb{E} \left(\hat{\mu}^{t|t} - \mu^{(t)} \right) \left(\hat{\mu}^{t|t} - \mu^{(t)} \right)^T$ be the covariance matrix associated with our estimation error at time t , and let $\hat{\mu}^{0|0} = \mathbb{E}(\mu^{(0)})$ and $\mathbf{P}^{0|0}$ be initial values. In the ‘prediction step’ we compute the prediction

$$\hat{\mu}^{t+1|t} = \Xi^{(t)}\hat{\mu}^{t|t} \quad (9.46)$$

and its variance

$$\mathbf{P}^{t+1|t} = \Xi^{(t)}\mathbf{P}^{t|t}\Xi^{(t)^T} + \Psi^{(t)}. \quad (9.47)$$

In the ‘estimation step’ we then update this prediction, resulting in the estimate

⁷ That is, Kalman filtering produces a minimum variance unbiased estimator (MVUE), under the model specified in (9.44) and (9.45).

$$\hat{\mu}^{t+1|t+1} = \hat{\mu}^{t+1|t} + \mathbf{K}^{(t+1)} \left(\mathbf{x}^{(t+1)} - \mathbf{B}^{(t+1)} \hat{\mu}^{t+1|t} \right) , \quad (9.48)$$

where

$$\mathbf{K}^{(t+1)} = \mathbf{P}^{t+1|t} \mathbf{B}^{(t+1)T} \left[\mathbf{B}^{(t)} \mathbf{P}^{t+1|t} \mathbf{B}^{(t+1)T} + \Sigma^{(t+1)} \right]^{-1} . \quad (9.49)$$

At this stage, the covariance matrix of our estimation error also is updated, as

$$\mathbf{P}^{t+1|t+1} = \left[\mathbf{I} - \mathbf{K}^{(t+1)} \mathbf{B}^{(t+1)} \right] \mathbf{P}^{t+1|t} \left[\mathbf{I} - \mathbf{K}^{(t+1)} \mathbf{B}^{(t+1)} \right]^T + \mathbf{K}^{(t+1)} \Sigma^{(t+1)} \mathbf{K}^{(t+1)T} . \quad (9.50)$$

A derivation of these equations may be found in most graduate texts on statistical signal processing. For a thorough introduction to the topic of Kalman filtering see, for example, Chui and Chen [86]. Note the similarity between equation (9.48) here, in the dynamic setting, and equation (9.28) previously, in the static setting. In both cases, an initial estimate of μ is updated, based on the quality of predictions generated by that estimate for an observed $\mathbf{X} = \mathbf{x}$. In the static case, the prediction often comes from previous studies or a low-level sampling of the underlying origin-destination flows \mathbf{Z} , whereas in the dynamic cases, the prediction is generated from the measurements that are preceding in time.

There are certain practical issues that must be addressed in implementing the strategy described above. Most importantly, the matrices $\Xi^{(t)}$, $\Psi^{(t)}$, and $\Sigma^{(t)}$ must be determined. Typically these matrices are assumed to be constants Ξ , Ψ , and Σ over some (perhaps short) length of time, and their values, estimated from measurements. The manner of estimation varies to some extent with both the particulars of the model and the type of data available. In general, however, given, say, observations $\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(\tau)}$, some version of the expectation-maximization (EM) algorithm⁸ is used, following the method introduced in Shumway and Stoffer [352]. A concise description of the relevant details of the algorithm, as well as additional related references, may be found in the short survey by Ghahramani and Hinton [171]. When a set of auxiliary measurements of the flow volumes \mathbf{Z} can be obtained, as is sometimes possible, the task of estimation can be simplified, and use of the EM algorithm avoided, as we illustrate below in the case study of Section 9.3.3. Regardless of the manner in which estimates are generated, the algorithm periodically must be re-calibrated with new starting values and new estimates Ξ , Ψ , and Σ . In order to decide when to recalibrate, the error process $\mathbf{x}^{(t+1)} - \mathbf{B}^{(t+1)} \hat{\mu}^{(t+1)}$ can be monitored to see whether, for example, any component of it is found to exceed some pre-specified number of standard deviations during too many consecutive time intervals. The appropriate standard deviations are the square roots of the diagonal entries of the matrices $\mathbf{B}^{(t)} \mathbf{P}^{t+1|t} \mathbf{B}^{(t+1)T} + \Sigma^{(t+1)}$.

⁸ The EM algorithm is an iterative algorithm for maximizing a log-likelihood. It is designed to be used in contexts where it is difficult to work with the original likelihood, but where a more tractable likelihood can be obtained by ‘enlarging’ the data space, through appropriately defined latent variables. The algorithm iterates between an expectation (E) step and a maximization (M) step, and under appropriate conditions will converge to the maximum of the original likelihood. See, for example, Hastie, Tibshirani, and Friedman [194, Ch. 8.5] for an overview and additional references.

Other methods proposed for dynamic traffic matrix estimation in the spirit of Kalman filtering include the recursive least-squares estimators of Nihan and Davis [303] and the time-varying method of Cao, Davis, Wiel, and Yu [73].

9.3.3 Case Study: Internet Traffic Matrix Estimation

The network supporting Internet traffic can be thought of as a network of networks, a collection of so-called *autonomous systems* (ASes), each constituting a set of routers and communication links under the control of a single administrative entity. The latter, which are frequently Internet service providers (e.g., Sprint, AT&T, etc.), monitor their networks routinely for a whole host of purposes. For example, network operators need to be able to identify when a failure has occurred in the network (e.g., a link ‘goes down’), the extent of that failure, and if possible, the reason for it. Similarly, they need to be able to track the traffic loads across the network, so that routing may be adjusted as needed, to minimize congestion, in an effort to optimize the quality of service provided. In addition, information on traffic loads is necessary at the engineering and management levels for purposes of capacity planning. Finally, in light of the ever-present threat from malicious traffic, such as is generated by denial of service attacks and worms, traffic monitoring has come to be a critical component of security measures put in place by providers.

The availability of traffic matrices $\mathbf{Z}^{(t)}$ is fundamental to being able to conduct these sorts of monitoring operations. Historically, it has been relatively easy to obtain measurements on the net out-flows $Z_{i+}^{(t)}$ and net in-flows $Z_{+i}^{(t)}$ associated with each vertex i in an Internet network. Traffic is sampled at the routers corresponding to each vertex and then, at regular intervals, relevant information on the sampled traffic volume is sent to a central location. The recovery of traffic matrices from these measurements can be approached as a traffic matrix estimation problem. More recently, it is now possible to obtain direct measurements of traffic flow volumes in Internet networks. However, this option is frequently not exercised by service providers, for various reasons, including concerns about the volume of data generated and its potential to adversely impact the quality of service experienced by users of the network. Rather, these measurements more commonly are used to calibrate and improve the design of methods for Internet traffic matrix estimation.

In order to illustrate the application of the traffic matrix estimation methods presented in this chapter, in the context of the Internet, we present an analysis of data constructed from measurements of traffic flows on a ‘backbone’ network, which is analogous in its role to that of a nation-wide system of highways. Specifically, starting with actual origin-destination traffic volume measurements, and a routing matrix corresponding to the period of measurement, we manufacture a set of link count pseudo-measurements. We then apply an entropy-based method and the method of Kalman filtering to these pseudo-measurements, as representative static and dynamic methods, respectively, and compare their performance in recovering

the original traffic volume measurements. Our analysis is in the spirit of that in Soule et al. [364].

Our data are based on measurements of origin-destination flows on the Abilene network, taken continuously over a seven-day period, starting December 22, 2003. The relevant portion of the Abilene network, as it was at that time, may be conceptualized as a graph consisting of 11 vertices and 30 links (i.e., essentially a simplified version of that depicted in Figure 1.1). Each vertex corresponds to a ‘point-of-presence’ (PoP), encompassing a major metropolitan region, where traffic may enter or exit the network. The links represent systems of optical transportation technologies and routing devices, running from one PoP to another. There are two such links for every one of the 14 corresponding undirected edges shown in Figure 1.1. In addition, there are two other links associated with a connection between Kansas City and Sunnyvale that has since been discontinued (and hence is not shown on the more recent map in Figure 1.1). Data were obtained using the automated sampling protocols implemented on the Abilene routers, which resulted in the capture of roughly 1% of all packets.⁹ The size of a packet is measured in bytes, and the volume of traffic between each origin-destination PoP pair was therefore measured in units of total bytes counted. The traffic flow counts were aggregated to five-minute intervals, a standard device used to avoid issues of time synchronization across the network. See Lakhina et al. [245] for additional details of this nature.

We therefore have flow measurements $\{\mathbf{z}^{(t)}\}_{t=1}^{\tau}$ for a total of $\tau = 12 \times 24 \times 7 = 2,016$ consecutive time periods, across $11 \times 10 = 110$ origin-destination pairs in the network. Obtaining a 30×110 routing matrix for Abilene during this period, we create pseudo-measurements in the form of link counts $\mathbf{x}^{(t)} = \mathbf{B}\mathbf{z}^{(t)}$, for $t = 1, \dots, 2,016$. Our goal will be to recover the flow volumes $\mathbf{z}^{(t)}$ from the link counts $\mathbf{x}^{(t)}$. To illustrate the nature of the challenge we face, in Figure 9.5 are shown the byte counts for the week (in units of kilobytes per second) over the link running from Denver to Sunnyvale. Also shown are the flow volume counts for the two flows that are routed over this link, carrying traffic from Denver to Sunnyvale and from Denver to Los Angeles. We clearly can see certain characteristics of the two individual traffic flows in the composite link data, but the aggregation of the individual flows in constructing the link data is sufficient to make the recovery of these flows impossible in the absence of any additional information.

In solving our traffic estimation problem, our choice of entropy-based method is a version of the ‘tomography’ approach proposed by Zhang, Roughan, Lund, and Donoho [414]. This method, already described briefly in Section 9.3.1.3, is so named in reference to the facts that it utilizes a gravity model prior and that traffic matrix estimation in this area is referred to as a ‘network tomography’ problem, following Vardi [386]. For each $\mathbf{x} = \mathbf{x}^{(t)}$, our estimates of the values in $\mathbf{z} = \mathbf{z}^{(t)}$ are obtained by solving the optimization

⁹ Packets are the basic building blocks of Internet traffic, each consisting of a small ‘header,’ containing information like source and destination IP addresses, and a ‘payload,’ containing the data being transmitted, such as a portion of an email or a music download.

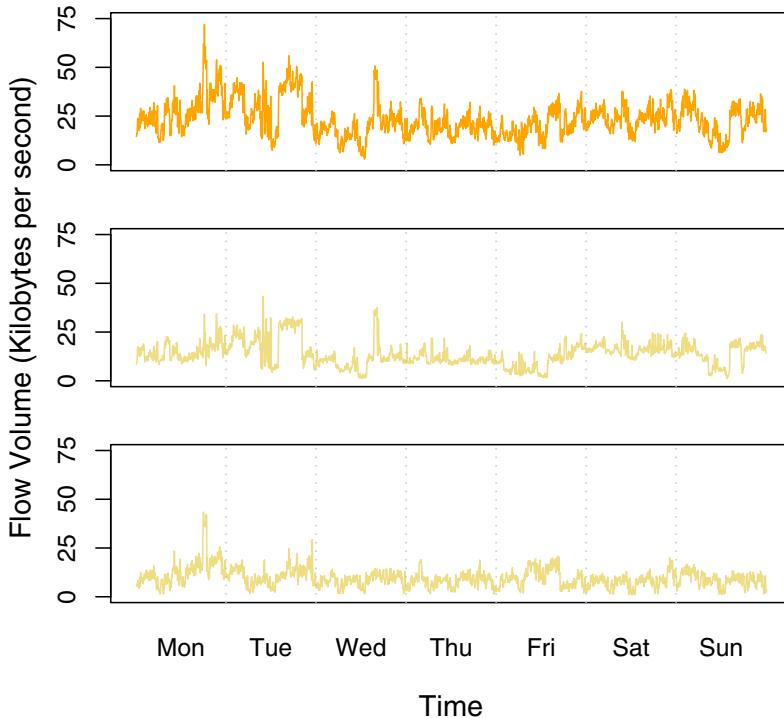


Fig. 9.5 Link counts (top, in orange) for the Denver to Sunnyvale link in the Abilene network, as compared to the origin-destination flow counts for the traffic passing over this link (in gold) from Denver to Sunnyvale (middle) and from Denver to Los Angeles (bottom).

$$\min_{\mathbf{z} \geq 0} (\mathbf{x} - \mathbf{Bz})^T (\mathbf{x} - \mathbf{Bz}) + \lambda \sum_{i,j} \frac{z_{ij}}{z_{++}^{(0)}} \log \left(\frac{z_{ij}}{z_{ij}^{(0)}} \right) , \quad (9.51)$$

where

$$z_{ij}^{(0)} = z_{i+}^{(0)} \times z_{+j}^{(0)} \quad (9.52)$$

is the product of the net out-flow and in-flow at vertices i and j , respectively, and $z_{++}^{(0)}$ is the total traffic in the network. The values $z_{i+}^{(0)}$, $z_{+j}^{(0)}$, and $z_{++}^{(0)}$ define what Zhang, Roughan, Lund, and Donoho [414] call a ‘simple gravity model’ prior,¹⁰ and can be

¹⁰ Zhang et al. actually propose and implement a more sophisticated gravity model prior, designed to better capture the effect on traffic volume of certain asymmetrical routing relationships. See also Erramilli, Crovella, and Taft [137].

obtained through appropriate summation of the elements in the vector \mathbf{x} of observed link counts. The optimization in (9.51) was performed using the *maxent* package in **MATLAB**, produced by Per Christian Hansen.¹¹ Following the recommendation of Zhang, Roughan, Lund, and Donoho [414], a value of $\lambda = (0.01)^2$ was used, although like those authors, our results were found to be rather robust to this choice.

Our implementation of the Kalman filtering method largely parallels that of Soule et al. [365]. Here we specify a model of the form

$$\mathbf{Z}^{(t+1)} = \Xi \mathbf{Z}^{(t)} + \boldsymbol{\eta}^{(t)} \quad (9.53)$$

$$\mathbf{X}^{(t)} = \mathbf{B} \mathbf{Z}^{(t)}, \quad (9.54)$$

where the $\boldsymbol{\eta}^{(t)}$ are assumed to be zero-mean random vectors, with common covariance Ψ . Note that the lack of an error term in the observation equation (9.54) is equivalent to using an error covariance matrix $\Sigma = 0$, in the notation of Section 9.3.2, for calculating our Kalman filter equations. The error term is omitted here due to the fact that no measurement error was injected during our construction of the pseudo-measurements $\mathbf{x}^{(t)} = \mathbf{B} \mathbf{z}^{(t)}$. The unknown parameters in our model are therefore just the two time-independent matrices Ξ and Ψ . These were estimated by reserving the first day of flow volume measurements, $\mathbf{z}^{(t)}$, $t = 1, \dots, 288$, and fitting independent models of the form

$$Z_{ij}^{(t+1)} = \Xi_{ii} Z_{ij}^{(t)} + \eta_{ij}^{(t)}, \quad (9.55)$$

separately for each pair (i, j) , using standard techniques.¹² Such usage of preliminary measurements mimics the idea of a service provider periodically exercising its capability to measure flow volumes directly, but choosing to do so only sparingly. Predictions of the flow volumes for the remaining six days were obtained through application of the appropriate Kalman filter equations to the link volumes $\mathbf{x}^{(t)}$, for $t = 289, \dots, 2,016$. Starting values were set using the mean and covariance of the first day flow volume measurements.

Note that our usage of these auxiliary measurements, to calibrate our model and initialize the prediction procedure, arguably gives the Kalman filtering method an unfair advantage over the tomogravity method. Soule et al. [364] have noted that tomogravity methods can suffer from a sizable estimation bias. Therefore, we also implemented a simple bias-corrected version of the tomogravity method described above, in which the original estimates were shifted uniformly in time by the amount of the average bias observed over the first day.

In Figure 9.6 we compare the performance of the tomogravity methods and the Kalman filtering method. Because the performance varies both as a function of time (i.e., temporally) and as a function of origin-destination flow (i.e., ‘spatially’) we have summarized the error associated with the methods in two ways. In order to compare temporal behavior, we have plotted the absolute error averaged over all

¹¹ Available at <http://www2.imm.dtu.dk/pch/Regutools/>.

¹² More precisely, imposing appropriate conditions on the Ξ_{ii} , each origin-destination flow was modeled and fit as a first-order auto-regressive, i.e., AR(1), time series. See, for example, Brockwell and Davis [66] or similar texts.

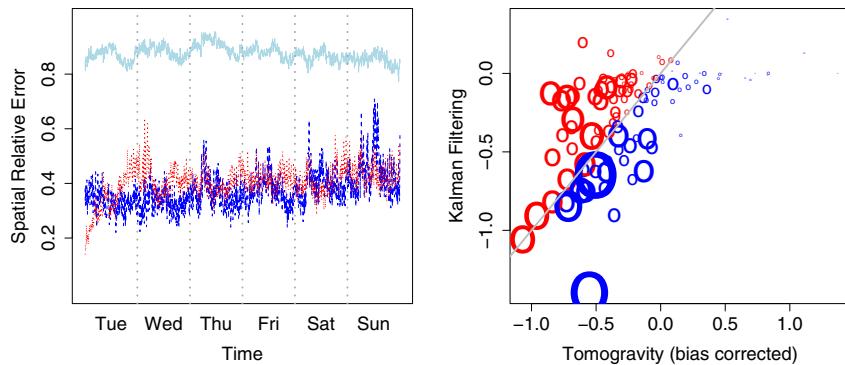


Fig. 9.6 Comparison of relative error (i.e., average absolute difference of predicted and actual origin-destination flow volumes, divided by average flow volume) for Abilene traffic flows. Left: Error aggregated over origin-destination pairs, as a function of time, for each of tomogravity (light blue), bias-corrected tomogravity (blue) and Kalman filtering (red). Right: Error aggregated over time, for each origin-destination pair, on a log-log scale, with area of symbol proportional to the mean volume of that flow, and colored according to which method had the larger relative error (tomogravity, blue; Kalman filtering, red).

origin-destination pairs, relative to the average flow volume, as a function of time. Similarly, in order to compare the spatial behavior, we have plotted absolute error averaged over time, again relative to the average flow volume, as a function of origin-destination. The temporal comparison shows that the tomogravity method, in its original form, consistently has a substantially greater relative error throughout the week. However, we can see too that with our simple bias correction, the two methods have comparable levels of error. Also evident from this plot is the fact that the Kalman filter performs its best at the start of the week, after which its performance degrades as time goes on. This behavior is not surprising, given the evolutionary nature of the algorithm, and could be ameliorated through the use of additional flow volume measurements to recalibrate the system (e.g., see Soule et al. [365]). In comparing the methods spatially, we see evidence that in general (but not always) the Kalman filtering method appears to do better than the (bias-corrected) tomogravity method at predicting the high- and low-volume traffic flows, but that the converse is true for moderate-volume flows.

Some representative examples are shown in Figure 9.7. Specifically, we show the predictions from the bias-corrected tomogravity and Kalman filtering methods for the traffic flows with the 2nd, 11th, 32nd, and 71st largest volumes, along with the flows themselves. The third of these corresponds to the Denver to Sunnyvale flow seen in the bottom of Figure 9.5. Predictions are plotted for the six days Tuesday through Sunday, omitting Monday. The performance is varied, with Kalman filtering outperforming tomogravity in the first and third cases (i.e., having relative errors of 0.0401 and 0.1247 versus 0.2806 and 0.4383, respectively), and vice versa, in

the second and fourth cases (i.e., 0.7482 and 0.8464 versus 0.1413 and 0.5927, respectively). Perhaps more interestingly, however, we see that while the Kalman filtering method captures the underlying dynamics of the flow to a reasonable extent in all four cases, this is not so for the tomogravity method, which completely misses the dynamics in the first and third cases. In fact, this phenomenon was observed in a large number of the cases, across the 110 traffic flows, where tomogravity was outperformed by Kalman filtering.

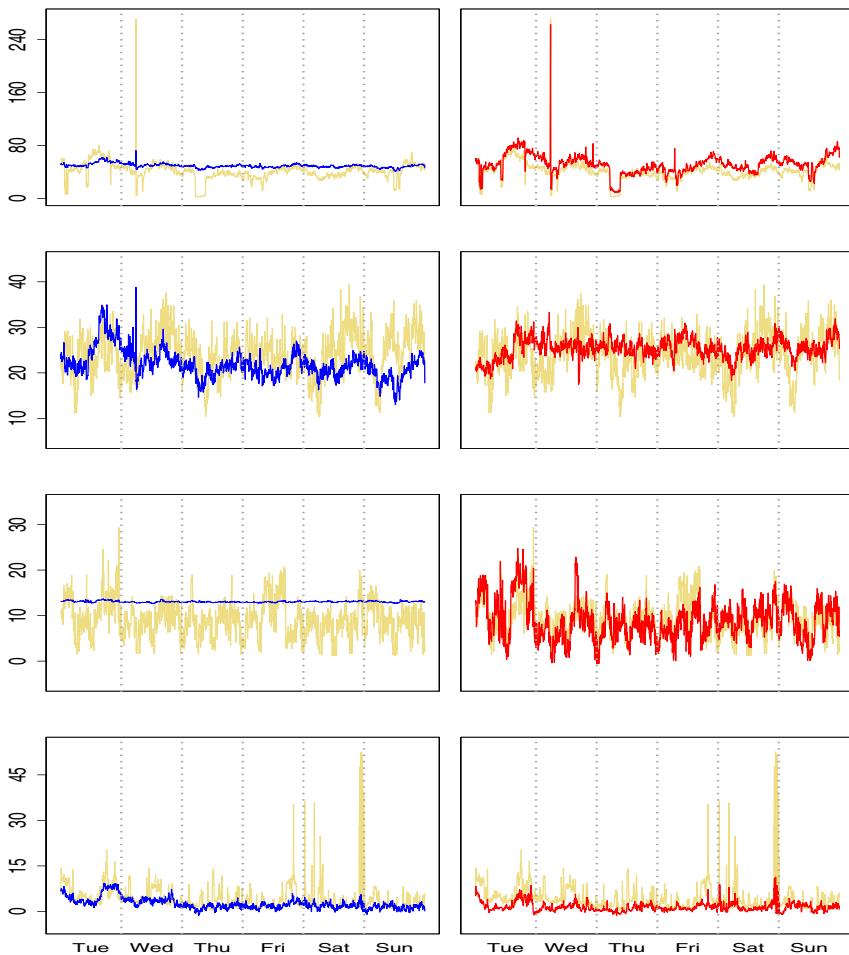


Fig. 9.7 Traffic flow volume predictions from bias-corrected tomogravity (left, in blue) and Kalman filtering (right, in red) methods, for four flows with volumes ranging from high (top) to low (bottom). Actual flow volumes are shown in yellow.

This distinction may be of greater or lesser importance depending on the application to which the estimated traffic matrices are to be put. For example, if the estimates are to be used as input to a system designed to detect anomalous traffic volumes (e.g., due to coordinated attacks on personal machines), it is important to correctly track the shape of the flow, since variations from typical conditions can be telling in this regard. On the other hand, for certain traffic engineering tasks (e.g., making routing changes in the networks to balance network load) or long-term planning for network capacity, it may be sufficient just to track the overall mean behavior successfully.

9.4 Estimation of Network Flow Costs

As was mentioned in Section 9.1, apart from flow volumes, the other major quantity of interest in transportation networks is – broadly speaking – cost. Depending on the context and the application, the term ‘cost’ might refer to a certain concrete quantity, such as the time, distance, or expense pertaining to a trip between locations in the network, or to a more abstract quantity, such as comfort. It might also be defined as a composite of a number of such individual quantities (i.e., as a so-called generalized cost).

Costs are usually discussed at two levels of network granularity: on paths and on links. A standard simplification relates the two through a linear model. For example, for simplicity, suppose that the paths routing between each origin-destination pair (i, j) are unique, so that the routing matrix \mathbf{B} reduces to a binary matrix. Then we would write

$$\mathbf{c} = \mathbf{B}^T \mathbf{x} , \quad (9.56)$$

where \mathbf{c} is an $IJ \times 1$ vector of path costs c_{ij} for travel from origins $i \in \mathcal{I}$ to destinations $j \in \mathcal{J}$, and \mathbf{x} here is a $N_e \times 1$ vector of costs $\{x_e\}_{e \in E}$ for the links on our network graph $G = (V, E)$. The expression in (9.56) states simply that the cost associated with each path is the sum of the costs of the links traversed. Such linearity assumptions are often well justified in practice. For instance, the cost in gasoline used to drive on a road network between two geographical locations can be thought of in terms of the gasoline used on each of the intermediary links. Similarly, the time required for a commodity to be moved along a path from one end of a network to another – sometimes called the *delay* – can be broken down according the delays on the relevant links.

In this section, we will focus on a particular class of problems involving the inference of costs, where we assume that it is possible to directly measure path costs c_{ij} for some subset of origin-destination pairs (i, j) in the network. Such measurements are often called *end-to-end measurements*. We will discuss two related problems, with the goal in the first being to infer characteristics of the link costs \mathbf{x} , and in the second, characteristics of the full set of path costs \mathbf{c} . The former is an instance of

what is often called *active network tomography*,¹³ while the second has been referred to as *network kriging*.¹⁴ In both problems, we will see that it is possible to exploit redundancies among the links traversed by the various paths to estimate the characteristics of interest. Such problems have been an area of major focus in the field of computer network traffic analysis, although they are in principle directly applicable to other types of transportation networks where the appropriate measurements can be made.

In fact, in the transportation sciences it is traditionally a somewhat different cost estimation problem that is encountered, within the context of the canonical *traffic assignment problem*. There, the goal is to infer (i.e., ‘assign’) the volume of traffic for all (non-unique) paths in the network, between all origin-destination pairs, using a traffic matrix \mathbf{z} as input. But a set of link costs is also included in the formulation of the problem, and typically is part of the inferred output. These costs are related to \mathbf{z} using particular functional forms deriving, for example, from consideration of the economic theory of supply and demand. A detailed discussion of this problem is beyond the scope of the book; we instead refer the reader to Ortúzar and Willumsen [306, Ch. 10] and Bell and Iida [32, Chs. 5–6].

9.4.1 Link Costs from End-to-end Measurements

Suppose that our interest is in characteristics of the link costs \mathbf{x} . In the transportation sciences, as we have just observed, link costs play a fundamental role in the traffic assignment problem. In computer network traffic analysis, they are basic indicators of the local level of performance of the network. Unfortunately, it is generally difficult to obtain measurements of link costs. On the other hand, it can sometimes be comparatively easier to measure the corresponding end-to-end costs on paths.

For example, consider the case where the cost of interest to us is delay (i.e., the time it takes to traverse from one point to another in the network). People generally do not keep track of the time it takes them, say, to move between points along the highway during their morning commute, but can likely report their total commute time on any given day. Alternatively, Internet service providers typically will not even know the topology of networks beyond their own (i.e., recall the case study in Section 7.4.5), much less be able to measure delays on links in those networks, and

¹³ The adjective ‘active’ is used here in reference to the fact that measurements of the path costs c_{ij} are generated by actively injecting traffic into the network, while the term ‘tomography’ is meant to evoke the image of tomographic medical imaging (e.g., of the human brain) in seeking to infer ‘internal’ network characteristics from these measurements. The problem of computer network topology identification, described in the case study of Section 7.4.5, is another example of active network tomography. In contrast, the label *passive network tomography* is sometimes applied to the traffic matrix estimation problem of Section 9.3, since the measured link counts are of traffic already flowing in the network.

¹⁴ The term ‘kriging’ is used in the geosciences in connection with methods of spatial interpolation that predict unknown values of a process from values observed at a subset of locations. See Cressie [102], for example, for an overview and additional references.

yet it is possible to obtain measurements of delays on paths from locations in their own network to others.

Formally, we can represent this situation by saying that we observe some subset $\mathbf{c}_s = (c_{i_1,j_1}, \dots, c_{i_s,j_s})^T$ of s of the IJ costs in \mathbf{c} . Denoting the corresponding $N_e \times s$ sub-matrix of \mathbf{B} by \mathbf{B}_s , in light of (9.56) we have $\mathbf{c}_s = \mathbf{B}_s^T \mathbf{x}$. Our goal is to infer some characteristic(s) of \mathbf{x} from the measurements in \mathbf{c}_s .

Work in this area can be roughly organized according to four aspects: (i) the type of cost and characteristics of that cost that are of interest, (ii) the mechanism by which end-to-end cost measurements are taken, (iii) the topology of the sub-network defined by the s origin-destination pairs measured, and the links traversed by the paths between them, and (iv) the statistical inference algorithm used. Of course, the specific choice of each these four aspects is often highly inter-related. Useful surveys include Coates, Hero, Nowak, and Yu [96] and Castro et al. [78].

9.4.1.1 End-to-end Measurements

We illustrate the nature of end-to-end measurements in the context where delay is the cost of interest.

Example 9.6 (Multicast Measurements of End-to-end Delay). We have seen in Section 7.4 how it is possible in computer networks to send probes (i.e., small packets of information) from a given origin to various destinations. The ‘multicast’ probes described in Example 7.4 are one such example. There the measurements we discussed were binary measurements, indicating whether or not a probe from an origin arrived at each particular destination. However, it is also possible to measure the delay involved with the travel of packets from the origin to the destinations (with the delay taken to be infinite in the case where the packet does not arrive).

Recall that a multicast probe sends a copy of the same packet from an origin simultaneously to a set of destinations, and that this process induces a sub-network of our network graph G that takes the form of a tree $T = (V_T, E_T)$, with root vertex r and N_l leaf vertices $R \subset V_T$. This tree corresponds to the logical topology underlying the flow of traffic on G from r to the vertices in R . Figure 9.8 shows an illustration, with a tree consisting of one root, $N_l = 6$ leaves, and three internal vertices.

A multicast delay experiment consists of sending n such probes into the network, and recording the delays $\{c_{r,j}^{(i)}\}_{i=1}^n$ for each pair (r, j) , where $j = 1, \dots, N_l$. The characteristic most commonly considered in this setting is the distribution of the link delays X_e , for each $e \in E_T$. It is standard to assume (after appropriate binning) a discrete form for this distribution, where the link delays are restricted to $b + 2$ intervals indexed by the set $\{0, q, \dots, bq, \infty\}$. Letting

$$\alpha_e(k) = \mathbb{P}(X_e = kq) , \quad (9.57)$$

our goal can be stated as that of inferring the N_e sets $\{\alpha_e(k)\}_{k=0}^b$ from the delay measurements.

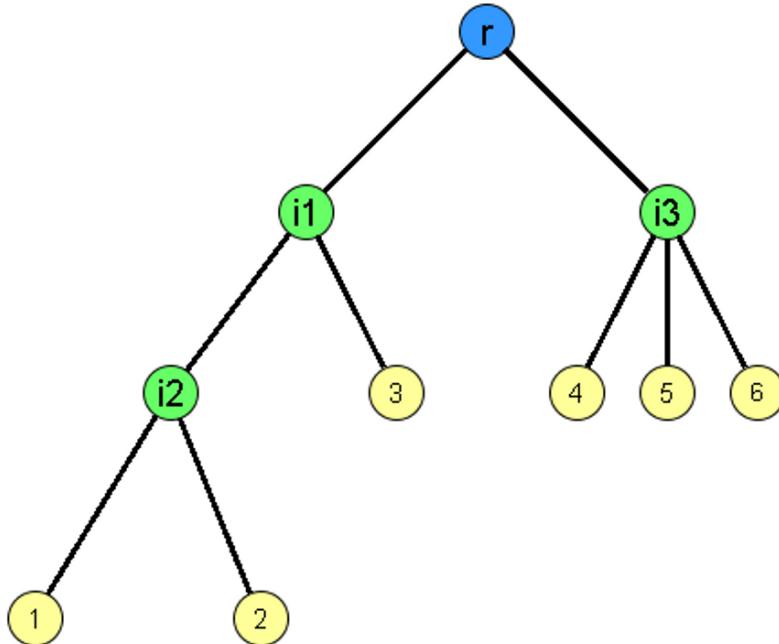


Fig. 9.8 Schematic illustration of a multicast tree for the link delay estimation problem.

In order to ensure identifiability of this inference problem, it is assumed that $\alpha_e(0) > 0$ for all $e \in E_T$, which means that for any given packet there is a positive probability on each link that it encounters ‘zero’ delay. Note that for this assumption to hold, it is generally necessary when defining the discrete delay distribution to compensate for some minimal level of delay common to traffic on all links. \square

The standard reference for the multicast delay problem is Lo Presti, Duffield, Horowitz, and Towsley [265]. Citing the possibly prohibitive overhead associated with multicast in some settings, the use of other types of probes has also been proposed, with the primary examples being *unicast* probes (e.g., Tsang, Coates, and Nowak [381]) and *bicast* probes (e.g., Liang and Yu [260]). The bicast scheme basically involves sending multicast probes to individual pairs of destinations. The unicast scheme, on the other hand, involves sending pairs of closely spaced packets from the origin to two separate destinations (i.e., similar to the ‘sandwich’ probes described in the case study of Section 7.4.5). In both cases, each such pair of destinations ends up having a simple binary tree associated with it, consisting of the root r , two leaf vertices, and a single internal vertex. Lawrence, Michailidis, and Nair [250] have introduced a general framework that allows flexibility in combining various of these probing techniques, called *flexicast*. Sufficient conditions for iden-

tifiability are discussed. There has also been some work on the use of probes from multiple origins. See, for example, Rabbat, Coates, and Nowak [320].

The other type of end-to-end measurements on which substantial work has been done – again, in the area of computer network traffic analysis – is measurements on packet loss. Let p_{ij} be the probability a packet is lost along the path from i to j , and p_e , the probability that it is lost along edge e . Then, since a packet is lost along a path if and only if it is lost along one of the constituent links, under the assumption that packet loss is independent across links we can write $1 - p_{ij} = \prod_{e \in E_{ij}} (1 - p_e)$, where E_{ij} is the set of links defining the relevant path. Taking logarithms on both sides and combining the resulting equations over all origin-destination pairs (i, j) , we obtain a linear system relating the log-rates $\log(1 - p_{ij})$ on paths to the log-rates $\log(1 - p_e)$ on links, analogous to that in (9.56). The structure of the problem of estimating the link loss rates p_e from packet loss measurements is closely related to the topology inference problem described in Section 7.4.1. See Caceres, Duffield, Horowitz, and Towsley [71].

9.4.1.2 Estimation of Link Cost Characteristics

Given path cost measurements \mathbf{c}_s on a subset of s paths, our hopes for accurately estimating some characteristic(s) of the link costs \mathbf{x} lie in the fact that in practice most links frequently are traversed by multiple paths. As a result of this sharing of links by paths, the path cost measurements \mathbf{c}_s are correlated in an advantageous manner. In particular, the costs for two paths sharing many links should be more similar than for two paths sharing fewer links, and furthermore, the difference in their costs should be reflective of the difference in costs on the links *not* shared.

For example, in the tree shown in Figure 9.8, the links from the root r to the internal vertices i_1 and i_3 are each traversed by paths to three of the six leaves, and the link joining the internal vertices i_1 and i_2 is traversed by paths to two of the leaves. Consider the paths from r to the leaf vertices 1 and 2. Because they differ only in their last link, sharing as they do the links $\{r, i_1\}$ and $\{i_1, i_2\}$, the difference in their path costs c_{r1} and c_{r2} reflects the difference in delays on the links $\{i_2, 1\}$ and $\{i_2, 2\}$.

Methods of inference in this area exploit these observations in various manners. Consider again, for instance, the problem of multicast-based estimation of delay, as introduced in Example 9.6.

Example 9.7 (Multicast Estimation of Link Delay Distribution). Lo Presti, Duffield, Horowitz, and Towsley [265] propose a framework for estimating the delay distributions $\{\alpha_e(k)\}_{k=0}^b$, using multicast measurements, that is qualitatively quite similar to that proposed by Duffield, Horowitz, Lo Presti, and Towsley [128] for tomographic network topology inference, as described in Example 7.5.

For each vertex $j \in V_T \setminus R$, let

$$\gamma_j(k) = \mathbb{P}\left(\min_{j' \in R(j)} C_{rj'} \leq kq\right) \quad (9.58)$$

be the probability that the shortest delay on paths from r to the leaves in $R(j)$ is at most kq , where $R(j)$ is the set of all leaf vertices in R that are descendants of j . Given our n observations $\{\mathbf{c}_s^{(i)}\}_{i=1}^n$, a natural estimator for the value $\gamma_j(k)$ in (9.58) is

$$\hat{\gamma}_j(k) = \frac{1}{n} \sum_{i=1}^n I_{\{\min_{j' \in R(j)} c_{rj'}^{(i)} \leq kq\}} . \quad (9.59)$$

The probabilities in (9.58) are analogous to the probabilities in (7.41), and as a result, the former probabilities – like the latter – can be related to the unknown parameters we seek to estimate in terms of a certain function that is invertible under appropriate conditions. Substituting the estimates (9.59) for (9.58), in a manner similar to our substitution of the estimates (7.44) for (7.41), and inverting, yields a collection of estimated delay distributions $\{\hat{\alpha}_e(k)\}_{k=0}^b$, for each link $e \in E_T$. The process of inversion, in this case, involves two steps, with a system of equations much like those in (7.43) being solved first, and then the second roots of a set of polynomial equations being found. There is a polynomial for each vertex $j \in V_T \setminus R$, and its order is equal to the out-degree of j . The expressions for these various equations are somewhat involved and therefore omitted here. Details may be found in Lo Presti, Duffield, Horowitz, and Towsley [265]. \square

The algorithm of Lo Presti et al. has the advantage of being fast and possessing good asymptotic properties. But a potential concern is the fact that in practice the polynomial second roots can sometimes be negative. An alternative is to use maximum likelihood methods but, unfortunately, the likelihood equations cannot be maximized in closed form. The expectation-maximization (EM) algorithm (i.e., see Footnote 8) can be implemented in a straightforward manner to maximize the likelihood, but tends to be extremely slow. Specifically, the algorithm scales like $O(b^{N_e})$, so that both increasing the number of bins b in our discretization of the delay distribution or increasing the network size N_e quickly becomes problematic. Generally we have some control over b , but little control over N_e . Some proposals have been put forth for speeding up the computations, yielding approximate maximum likelihood estimates, including the pseudo-likelihood method of Liang and Yu [260] and the ‘grafting’ method of Lawrence, Michailidis, and Nair [250].

Methods of estimation have also been proposed for the case where the cost of interest is the link loss rate. See, for example, Caceres, Duffield, Horowitz, and Towsley [71].

9.4.2 Path Costs from End-to-end Measurements

Suppose now that our main focus is on characteristics of some cost at the level of paths, rather than at the level of links. Although we are assuming an ability to measure this cost on paths, it is unusual in practice to be able to do so (or to want to do so) on *all* paths, since the number of origin-destination pairs IJ generally can be quite large (e.g., as large as $O(N_v^2)$ if all vertices can serve as both origins and

destinations). Fortunately, it is possible to recover substantial cost information on unmeasured paths from that on measured paths, due to redundancies among paths, if those that are measured are chosen in an appropriate fashion.

9.4.2.1 Interpolation of Path Costs

Consider the expression $\mathbf{c} = \mathbf{B}^T \mathbf{x}$ in (9.56). If we knew \mathbf{x} , and of course \mathbf{B} , we could recover any and all of the path costs c_{ij} trivially. But as we just saw in Section 9.4.1, it is rare that we will know \mathbf{x} . However, since our real interest is in \mathbf{c} , from a purely linear algebraic perspective it is not necessary to know \mathbf{x} itself, but rather only that portion of \mathbf{x} that lies in the row space of \mathbf{B}^T (i.e., the column space of \mathbf{B}). That is, recall that every such vector \mathbf{x} can be decomposed as

$$\mathbf{x} = \mathbf{x}_{\text{row}(\mathbf{B}^T)} + \mathbf{x}_{\text{null}(\mathbf{B}^T)} , \quad (9.60)$$

where the first vector in (9.60) lies in $\text{row}(\mathbf{B}^T)$, the space of all vectors in \mathbb{R}^{N_e} formed from linear combinations of the rows of \mathbf{B}^T , and the second vector, in $\text{null}(\mathbf{B}^T)$, its orthogonal complement. Then, by definition,

$$\mathbf{c} = \mathbf{B}^T \mathbf{x} = \mathbf{B}^T \mathbf{x}_{\text{row}(\mathbf{B}^T)} + \mathbf{B}^T \mathbf{x}_{\text{null}(\mathbf{B}^T)} = \mathbf{B}^T \mathbf{x}_{\text{row}(\mathbf{B}^T)} . \quad (9.61)$$

Exploiting this fact, Chen, Bindel, Song, and Katz [80] describe a method for recovering all of the costs c_{ij} in \mathbf{c} from measurements on just a subset of paths. More precisely, let \mathbf{c}_s denote the cost measurements on a subset of s chosen paths, and let \mathbf{c}_r be the unknown costs on the remaining $r = IJ - s$ paths, ordered such that $\mathbf{c} = (\mathbf{c}_s^T, \mathbf{c}_r^T)^T$. The method of Chen et al. allows us to recover \mathbf{c}_r from \mathbf{c}_s , by first recovering from \mathbf{c}_s that portion of \mathbf{x} that lies in $\text{row}(\mathbf{B}^T)$. As such, it is essentially a method of interpolation, and builds to some extent on earlier work of Shavitt, Sun, Wool, and Yenner [348] and of Nguyen and Thiran [300].

In more detail, let \mathbf{B}_s be an $N_e \times s$ sub-matrix of \mathbf{B} , with $s = \text{rank}(\mathbf{B})$, constructed so that each of its columns is independent. The selection of these s columns corresponds to a choice of s paths between s origin-destination pairs. The values in \mathbf{c}_s are the cost measurements on these paths. Define \mathbf{c}_r as above, and let \mathbf{B}_r be that $N_e \times r$ sub-matrix of \mathbf{B} such that $\mathbf{B} = [\mathbf{B}_s, \mathbf{B}_r]$. Consider the system of equations

$$\mathbf{c}_s = \mathbf{B}_s^T \mathbf{x}_{\text{row}(\mathbf{B}^T)} . \quad (9.62)$$

Generally, in practice $\text{rank}(\mathbf{B}) \leq N_e$, and so this system may be under-determined, in which case it will not have a unique solution. As is common in such situations, we choose from among all possible solutions that with the smallest possible length, which is unique and is obtained through the use of the Moore-Penrose pseudo-inverse. That is, we choose the solution $(\mathbf{B}_s^T)^- \mathbf{c}_s$ to (9.62), where $(\mathbf{B}_s^T)^- = \mathbf{B}_s (\mathbf{B}_s^T \mathbf{B}_s)^{-1}$. Combining this solution with (9.61), we recover the unknown path costs, as

$$\mathbf{c}_r = \mathbf{B}_r^T \mathbf{B}_s (\mathbf{B}_s^T \mathbf{B}_s)^{-1} \mathbf{c}_s . \quad (9.63)$$

This approach to obtaining \mathbf{c} can offer substantial savings in measurement overhead, compared to explicit measurement of cost on all paths. For instance, when every origin also acts as a destination, in which case $I = J$ and $IJ = I^2$, Chen, Bindel, Song, and Katz [80] conjecture that $\text{rank}(\mathbf{B}) = O(I \log I)$ and present a combination of numerical and theoretical arguments in support of this suggestion. This quantity represents almost an order of magnitude reduction over the nominal number of measurements and is essentially driven by the amount of redundancy in the links traversed by the various paths, according to the routing matrix \mathbf{B} . The particular choice of s paths with independent columns in \mathbf{B} that define \mathbf{B}_s can be made using algorithms for computing various so-called ‘rank-revealing decompositions,’ such as Gaussian elimination with complete pivoting, QR decomposition with column pivoting, and the singular value decomposition. See Golub and van Loan [181]. The overall algorithm of Chen, Bindel, Song, and Katz [80] uses the QR decomposition for both path selection and calculation of the Moore-Penrose pseudo-inverse.

9.4.2.2 Prediction of Path Cost Characteristics

Although the interpolation method just described is appealing when the necessary number of measurements are obtainable, note that if even one less measurement is taken than is necessary, equation (9.63) does not hold and the subset of unknown path costs \mathbf{c}_r cannot be recovered exactly. However, the basic recovery problem may be recast profitably as one of statistical prediction, and standard linear predictive methods may be utilized to obtain results that can still be surprisingly accurate, even when based on $s \ll \text{rank}(\mathbf{B})$ measurements. This approach was proposed by Chua, Kolaczyk, and Crovella [85], and its potential for success rests on the observation that in practice the *effective* rank of \mathbf{B} can be smaller than its nominal rank – often noticeably so.

The singular value decomposition (SVD) is a standard tool for obtaining insight into the effective rank of a matrix. Given a routing matrix \mathbf{B} , recall that using the SVD, we can re-write \mathbf{B}^T in the form $\mathbf{B}^T = \mathbf{U}\Lambda^{1/2}\mathbf{V}^T$, where \mathbf{V} is an $N_e \times N_e$ orthogonal matrix, containing the eigenvectors of $\mathbf{B}\mathbf{B}^T$, Λ is an $N_e \times N_e$ diagonal matrix, containing the corresponding eigenvalues, and \mathbf{U} is an $IJ \times N_e$ matrix for which $\mathbf{U}^T\mathbf{U} = \mathbf{I}$. Alternatively, we can write

$$\mathbf{B}^T = \sum_{k=1}^{N_e} \lambda_k^{1/2} \mathbf{u}_k \mathbf{v}_k^T , \quad (9.64)$$

where λ_k is the k -th largest diagonal entry of Λ , and \mathbf{u}_k and \mathbf{v}_k are the k -th columns of \mathbf{U} and \mathbf{V} , respectively. See Golub and van Loan [181, Ch. 2.5], for example.

The number of non-zero entries on the diagonal of Λ will be equal to $\text{rank}(\mathbf{B})$. So in fact equation (9.64) shows that \mathbf{B}^T may be expressed as the sum of $\text{rank}(\mathbf{B}) \leq N_e$ matrices $\mathbf{u}_k \mathbf{v}_k^T$, each of which itself is of rank one. However, in the event that some of the smaller values λ_k are ‘close’ to zero, we can truncate this sum even further and

obtain an approximation to \mathbf{B}^T that is still correspondingly ‘close’ to the original.¹⁵ It is in this sense that \mathbf{B} is then said to be of effectively lower rank than $\text{rank}(\mathbf{B})$. We illustrate these ideas through the following example.

Example 9.8 (Reduced Dimensionality in Abilene). Figure 9.9 shows the eigenvalues for the routing matrix corresponding to the Abilene network that was described in the case study of Section 9.3.3. Recall that there are 110 paths traversing just 30 directed links. The large gap between the second and third eigenvalues, and the resulting ‘knee’ in the spectrum, are indicative of there being substantially more linear dependence among the rows of \mathbf{B}^T (i.e., the columns of \mathbf{B}) than suggested by its nominal rank of 30. The overall decay in the spectrum of eigenvalues suggests that measurements on roughly five to ten paths, and perhaps as few as two paths, may be sufficient to recover useful information about path costs \mathbf{c} in this network.

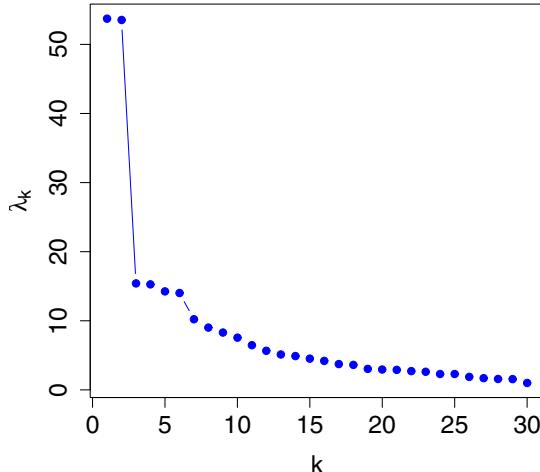


Fig. 9.9 Spectrum of eigenvalues for an Abilene routing matrix.

Chua, Kolaczyk, and Crovella [85] demonstrate an even more pronounced reduced dimensionality in the routing matrices for six additional networks, corresponding to commercial Internet service providers of various sizes. They also provide a general analytic result associating the eigenvalue decay for a routing matrix \mathbf{B} with the decay of the distribution of edge betweenness values, thus providing a

¹⁵ More precisely, let the truncation of (9.64) to the first K terms be denoted \mathbf{B}_K^T . Then the sum-of-squares of the entries (i.e., the squared Frobenius norm) of the difference $\mathbf{B}^T - \mathbf{B}_K^T$ is equal to $\lambda_{K+1} + \dots + \lambda_{\text{rank}(\mathbf{B})}$.

connection between the ideas in this section and the concept of centrality discussed in Section 4.2.2.

Additional insight into the network path properties on the Abilene network may be obtained through consideration of the eigenvectors in \mathbf{V} . For our purposes, these eigenvectors may be thought of as representing linearly independent ‘meta-paths’ in ‘link space’ (i.e., shared patterns of links common to the paths routed in this network), according to \mathbf{B} . These meta-paths are given an ordering through the magnitude of their corresponding eigenvalues, which essentially indicate for each meta-path the relative proportion of routed paths in the network that ‘match.’ Figure 9.10 shows visual representations of the first four eigenvectors of our Abilene routing matrix. These may be compared to the map of Abilene in Figure 1.1. The first eigenvector can be seen to concentrate along an east-west meta-path across the northern part of the network, while subsequent eigenvectors bring in successive refinements to this picture, with the second and third further emphasizing the connections to and within the meta-path indicated by the first, and the fourth introducing a second east-west meta-path along the southern part of the network. \square

Now consider the problem of predicting path costs in \mathbf{c} . Actually, we will consider a slightly more general problem – that of predicting an arbitrary linear summary of the path costs. Specifically, suppose we wish to predict a summary of the form $\mathbf{a}^T \mathbf{c}$, where \mathbf{a} is some known $IJ \times 1$ vector. Natural choices for \mathbf{a} are $a_{ij} \equiv 1/IJ$, for which $\mathbf{a}^T \mathbf{c}$ is the network-wide average path cost, and $a_{ij} = 1$ and $a_{i'j'} = 0$ when $(i', j') \neq (i, j)$, for which $\mathbf{a}^T \mathbf{c}$ is just the cost c_{ij} of the origin-destination pair (i, j) . Furthermore, assume that \mathbf{x} is the realization of a random vector \mathbf{X} , with mean μ and covariance Σ . From (9.56) it then follows that \mathbf{c} corresponds to a random vector \mathbf{C} that has mean and covariance $\mathbf{B}^T \mu$ and $\mathbf{B}^T \Sigma \mathbf{B}$, respectively. Finally, given a set of measured path costs \mathbf{c}_s , corresponding to $s \leq \text{rank}(\mathbf{B})$ independent paths in \mathbf{B}_s , and a predictor $p(\mathbf{c}_s)$ of $\mathbf{a}^T \mathbf{c}$, we will judge the quality of that predictor by the mean-squared prediction error $\mathbb{E}[(\mathbf{a}^T \mathbf{C} - p(\mathbf{C}_s))^2]$.

The best predictor under this criterion is known to be the conditional expectation

$$\mathbb{E}(\mathbf{a}^T \mathbf{C} | \mathbf{C}_s = \mathbf{c}_s) = \mathbf{a}_s^T \mathbf{c}_s + \mathbb{E}(\mathbf{a}_r^T \mathbf{C}_r | \mathbf{C}_s = \mathbf{c}_s) , \quad (9.65)$$

where $\mathbf{a} = (\mathbf{a}_s^T, \mathbf{a}_r^T)^T$ is partitioned in the same manner as \mathbf{c} . If we restrict our attention to predictors $p(\mathbf{c}_s)$ that are linear in \mathbf{c}_s , then this expression yields the best linear predictor (BLP), which can be shown to take the form

$$\widehat{(\mathbf{a}^T \mathbf{c})}_{BLP} = \mathbf{a}_s^T \mathbf{c}_s + \mathbf{a}_r^T \mathbf{B}_r^T \mu + \mathbf{a}_r^T \eta_* (\mathbf{c}_s - \mathbf{B}_s^T \mu) , \quad (9.66)$$

with η_* defined to be any solution to the equation $\eta_* \Omega_{ss} = \Omega_{rs}$, where $\Omega_{ss} = \mathbf{B}_s^T \Sigma \mathbf{B}_s$ and $\Omega_{rs} = \mathbf{B}_s^T \Sigma \mathbf{B}_r$. If no value for μ is available, such as might be obtained from historical data or a limited set of network-wide path cost measurements, it can be estimated from the measured path costs \mathbf{c}_s as

$$\hat{\mu} = [\mathbf{B}_s \Omega_{ss}^{-1} \mathbf{B}_s^T]^- \mathbf{B}_s \Omega_{ss}^{-1} \mathbf{c}_s , \quad (9.67)$$

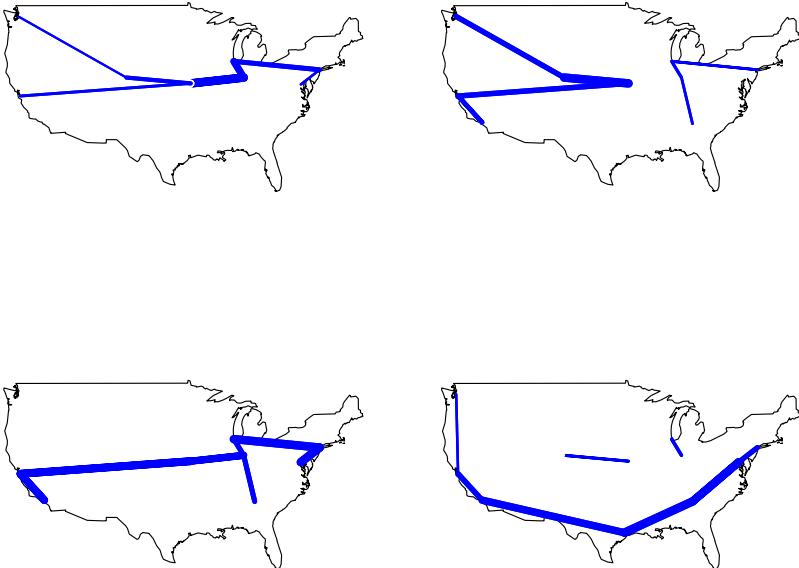


Fig. 9.10 Visual representation of the first four eigenvectors (top left and right, followed by bottom left and right, respectively) of an Abilene routing matrix. Each link is drawn with a thickness in proportion to the magnitude of its corresponding eigenvector component.

assuming Ω_{ss} is invertible, where \mathbf{M}^- again denotes the Moore-Penrose pseudo-inverse of a matrix \mathbf{M} . Substitution of (9.67) for μ in (9.66) can be shown to yield a predictor of the form

$$\widehat{(\mathbf{a}^T \mathbf{c})} = \mathbf{a}_s^T \mathbf{c}_s + \mathbf{a}_r^T \Omega_{rs} \Omega_{ss}^{-1} \mathbf{c}_s . \quad (9.68)$$

The predictor (9.68) and its derivation parallel closely what may be found in the geostatistics literature on spatial prediction (i.e., on kriging). See, for example, Christensen [84, pp. 225-227] for details. Accordingly, prediction using (9.68) has been referred to as ‘network kriging.’ If $\Sigma = \mathbf{I}$ and $s = \text{rank}(\mathbf{B})$, then this predictor effectively performs the interpolation of Chen, Bindel, Song, and Katz [80] in (9.63). In that case, of course, we recover the exact value of $\mathbf{a}^T \mathbf{c}$.

For the task of selecting just which s paths to measure costs upon, Chua, Koliczky, and Crovella [85] propose an algorithm based on the SVD of \mathbf{B} that is analogous to the QR -based algorithm of Chen, Bindel, Song, and Katz [80] mentioned

previously. They argue that such algorithms can be viewed as trying to select a subset of s paths that minimizes the mean-squared prediction error corresponding to the best linear predictor in (9.66), as a function of all possible subsets of s independent paths. They also characterize the theoretical performance that can be expected of network kriging. We illustrate here with the following numerical example, taken from their work, involving the prediction of path delay times.

Example 9.9 (Predicting Average Path Delay on Abilene). Chua, Kolaczyk, and Crovella [85] describe a process whereby they constructed a dataset of link delays $\mathbf{x}^{(t)}$ in the Abilene network, for every 10 minutes over a period of three days in 2003. From these they created a set of path-cost pseudo-measurements $\mathbf{c}^{(t)} = \mathbf{B}^T \mathbf{x}^{(t)}$, using the Abilene routing matrix \mathbf{B} of Example 9.8. The predictor (9.68) was then applied to a subset $\mathbf{c}_s^{(t)}$ of these path costs, separately for each time t , for various choices of $s \leq \text{rank}(\mathbf{B})$, where the paths measured were chosen according to the SVD-based algorithm mentioned just above. The covariance matrix Σ was taken to be diagonal, and was estimated as described in Chua, Kolaczyk, and Crovella [85].

Figure 9.11 shows the average path delay on Abilene predicted by network kriging, based on measurements from $s = 3, 5, 7$, or 9 paths, as a function of time period $t = 1, \dots, 432$, and the actual average path delay, which can be obtained from measurements on $s = \text{rank}(\mathbf{B}) = 30$ appropriately chosen paths. Two points are evident from examination of this figure. First, there clearly is a bias to the predictions, which becomes more pronounced as s decreases. Second, the predictions nevertheless capture much of the dynamics quite well, for all choices of s . The bias is to be expected, and is a result of the fact that \mathbf{B} is approximated by a matrix of rank $s < 30$. Practically speaking, information is lacking on links in the network that are traversed by none of the s measured paths.

However, it turns out that we can correct for this bias quite easily in these data if we allow ourselves to measure the delay on a set of 30 independent paths just once at the start of the three days studied. This use of auxiliary measurements is similar to that employed in the case study of Section 9.3.3 and represents a negligible addition to the overall network measurement load. Adjusting the predictions by the bias we observe in the first 10-minute time period produces predictions with an average of less than 1% absolute relative error, even with as little as $s = 3$ measured paths.

The accuracy of these predictions can in turn serve as the foundation for performing various high-level tasks. Chua, Kolaczyk, and Crovella [85] describe the use of the predicted average path delay in monitoring for anomalous levels of delay and in comparing the performance of two sub-networks. \square

Some extensions of the basic network kriging framework described here have been proposed. Qazi and Moors [317] have demonstrated that the use of principles of ridge regression (i.e., such as underlie equation (2.38) in Chapter 2) can help produce predictions with lower variance than those of (9.68). Coates, Pointurier, and Rabbat [98] introduce the use of Lasso regression and an alternative basis representation, within the context of a spatio-temporal approach. The incorporation of temporal correlations into the predictions is observed by these authors to noticeably

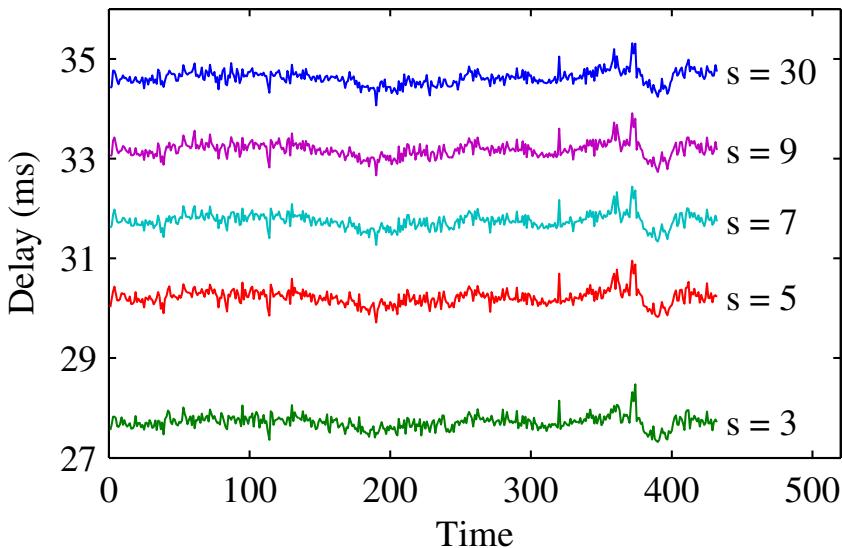


Fig. 9.11 Network kriging predictions of Abilene average delay over a period of three days.

decrease the bias, as might be expected. Finally, Sommers, Duffield, Barford, and Ron [363] describe a method for producing upper and lower bounds on quantiles of the distribution of costs on each path in the network. See also the work by Nguyen and Thiran [300, 301], who treat problems similar to those addressed in this section, but which involve binary measurements and arithmetic in the context of monitoring the ‘aliveness’ of paths in Internet networks.

9.5 Additional Related Topics and Reading

Besides those considered here, there are a number of other statistical topics relating to network flow data that have received nontrivial – and in some cases, substantial – attention in the literature. We briefly summarize a few of these now. As in the main body of this chapter, the examples we provide are dominated by contributions from the extremely active area of computer network traffic analysis, to the somewhat unfair exclusion of other areas in transportation more broadly.

The Poisson-like, and more generally, Markovian nature of the classical models for network flow data, of which the gravity models we discussed are perhaps the quintessential example, can in some contexts be quite insufficient. For example, while historically such models were considered reasonable for voice-data on standard telephone networks, like the Austrian call data we saw in Section 9.2, with the emergence of high-speed computer traffic networks, a new paradigm emerged. In particular, the celebrated paper of Leland, Taqqu, Willinger, and Wilson [254]

established that traffic of high temporal resolution on so-called local-area networks (LANs) was *statistically self-similar* (and therefore decidedly non-Markovian), in the sense that the temporal correlation structure of the underlying process was essentially indistinguishable from that of aggregated versions of itself. See the edited volume compiled by Park and Willinger [308], which covers a variety of developments building on this seminal work, as it relates to topics ranging from estimation and simulation, to queuing and congestion, to traffic control.

As the collection of network flow data becomes more and more automated, issues of sampling need to be examined. In computer networks, for example, routers are designed with the facility to sample from the massive flows of data streaming through them. A number of sampling designs are employed in this setting, including systematic sampling, random sampling, and stratified sampling. See, for example, Crovella and Krishnamurthy [105, Ch. 6.3.3] for discussion of such designs and the effect they can have on the types of statistical analyses to which such data are subjected. Similarly, the use of embedded devices (e.g., loop-detectors), cameras, and other related sensing devices is becoming increasingly common in the monitoring of highway traffic. Bickel et al. [38] offer a review of some of the statistical challenges that are emerging as a result, including issues of data quality and missing data.

Lastly, with respect to the measurement and inference of costs in networks, it should be noted that certainly not all costs of interest are amenable to linear formulations like that in (9.56). See, for instance, Shriram and Kaur [351] for a survey and evaluation of techniques for measuring the amount of end-to-end bandwidth available on computer network paths. Conversely, the exploitation of ‘spatial’ correlations among paths is not restricted to just problems involving cost. Lakhina et al. [245], for example, show that an analogous sense of reduced dimensionality exists for traffic flow volumes in backbone-level Internet networks, and use principal component analysis (PCA) to characterize typical temporal patterns across flows. This work in turn has served as the foundation for a series of papers by Lakhina and colleagues on the detection of computer network traffic anomalies using PCA-based methods, including Lakhina, Crovella, and Diot [244]. See also Zhang, Ge, Greenberg, and Roughan [415], whose work aims to address the problems of anomaly detection and network tomography within a single unified framework – which they dub *network anomography* – and which includes the PCA-based framework of Lakhina and colleagues as a special case.

Exercises

9.1. For the general gravity model, under the parameterization in (9.14), the log-likelihood (9.15), as a function of the parameter vectors $\alpha = (\alpha_1, \dots, \alpha_I)^T$, $\beta = (\beta_1, \dots, \beta_J)^T$, and $\theta = (\theta_1, \dots, \theta_K)^T$, takes the form

$$\ell(\alpha, \beta, \theta) = \sum_{i,j \in \mathcal{I} \times \mathcal{J}} z_{ij} (\alpha_i + \beta_j + \theta^T \mathbf{c}_{ij}) - \exp(\alpha_i + \beta_j + \theta^T \mathbf{c}_{ij}) .$$

Taking partial derivatives with respect to the elements α_i , β_j , and θ_k , setting the resulting equations equal to zero, and simplifying, show that the maximum likelihood estimates of α , β , and θ must satisfy the expressions in (9.16) and (9.17).

9.2. Consider the least-squares formulation of the static traffic matrix estimation problem, as introduced in Section 9.3.1.1. Recall that the goal in this problem is to recover the $IJ \times 1$ vector μ of expected flow volumes from the $N_e \times 1$ vector of link counts \mathbf{X} , where

$$\mathbf{X} = \mathbf{B}\mu + \varepsilon ,$$

with \mathbf{B} an $N_e \times IJ$ routing matrix and ε an $N_e \times 1$ vector of errors. Let η be an arbitrary $IJ \times 1$ vector. The value $\eta^T \mu$ is said to be *estimable* if there exists a matrix \mathbf{M} such that $\mathbb{E}(\mathbf{M}\mathbf{X}) = \eta^T \mu$.

- a. Show that $\eta^T \mu$ is estimable if and only if η is a solution to the equation

$$[\mathbf{I} - \mathbf{B}^T(\mathbf{B}\mathbf{B}^T)^- \mathbf{B}]\eta = 0 ,$$

where $(\mathbf{B}\mathbf{B}^T)^-$ denotes the Moore-Penrose pseudo-inverse of $\mathbf{B}\mathbf{B}^T$.

- b. For the simple network in Example 9.3, show that the expected origin volumes μ_{a+} and μ_{b+} and the expected destination volumes μ_{+c} and μ_{+d} are all estimable.
- c. If $\eta^T \mu$ is estimable, then given observed link volumes $\mathbf{X} = \mathbf{x}$, the best linear unbiased estimator¹⁶ of $\eta^T \mu$ is $\eta^T \hat{\mu}$, where $\hat{\mu} = (\mathbf{B}^T \mathbf{B})^- \mathbf{B}^T \mathbf{x}$. Show that, for the network in Example 9.3, the matrix $(\mathbf{B}^T \mathbf{B})^- \mathbf{B}^T$ can be expressed in the form

$$\frac{1}{4} \begin{bmatrix} 2 & 2 & -2 & -2 \\ 1 & -1 & 3 & 1 \\ -1 & 1 & 1 & 3 \\ 0 & 0 & 0 & 0 \end{bmatrix} .$$

Then, given observations $\mathbf{x} = (30, 27, 15, 10)^T$, show that

$$(\hat{\mu}_{a+}, \hat{\mu}_{b+}, \hat{\mu}_{+c}, \hat{\mu}_{+d}) = (30.5, 10.5, 26.5, 14.5) .$$

¹⁶ That is, an unbiased estimator that, among all other unbiased estimators, has minimal variance.

9.3. Consider the Poisson version of the static traffic matrix estimation problem, restricted to the subnetwork of the network in Figure 9.4 consisting of just the three vertices $\{a, v, c\}$ and the two links 1 and 2.

- a. Verify the result of Example 9.4, showing that upon observing the link counts $\mathbf{x} = (1, 2)^T$, the maximum likelihood estimate of the expected flow volumes μ is not given by that value for which the score equations $\partial \ell(\mu) / \partial \mu = 0$.
- b. Verify the results of Example 9.5, concluding that for this subnetwork we can simulate from the full joint conditional posterior $\mathbb{P}(\mathbf{Z} | \mu, \mathbf{X} = \mathbf{x})$ by first drawing $Z_{ac} = z_{ac}$ and then setting $z_{av} = x_1 - z_{ac}$ and $z_{vc} = x_2 - z_{ac}$.

9.4. Consider the problem of predicting path cost characteristics $\mathbf{a}^T \mathbf{c}$, based on measurements \mathbf{c}_s of a subset of path costs, as described in Section 9.4.2.2.

- a. Verify that substitution of (9.67) into (9.66) yields the predictor (9.68).
- b. Implement this predictor, and apply it to the Abilene delay data, in the manner of Example 9.9. Explore the accuracy with which you can predict the average delay, as you vary the choice of paths measured. Use the eigenvectors of the routing matrix to help guide your choice of measurements.
- c. The prediction of average path delay uses the weight vector \mathbf{a} with entries $a_{ij} = 1/IJ$. In order to predict the cost for an individual origin-destination pair (i, j) , we can let $a_{ij} = 1$ and $a_{i'j'} = 0$ for all $(i', j') \neq (i, j)$. Using the Abilene delay data, explore the accuracy with which you can predict the cost of individual unmeasured paths, as you vary the choice of paths measured.

Chapter 10

Graphical Models

A number of the probabilistic models seen in this book are members of the general class of graphical models. In this chapter, we briefly introduce this class of models and examine its relationship with some of those models encountered heretofore.

10.1 Introduction

We have seen a variety of techniques throughout this book for analyzing network data. Many of these techniques – especially in Chapters 6 through 9 – are based on probabilistic models. And among these, a number of them can be viewed profitably as specific instances of a particular class of probabilistic models called *graphical models*. We have chosen not to emphasize this connection previously, as it is not necessarily vital to the exposition of these models individually, nor to their direct application. Nevertheless, this connection is quite important at the level of conceptual, theoretical, and algorithmic foundations. Moreover, some understanding of the general graphical modeling framework is crucial for those interested in developing nontrivial extensions of these types of probabilistic models.

Therefore, in this final chapter we provide a brief introduction to the basic elements of the graphical modeling formalism. Our goal primarily is to make clear the connections between the class of graphical models and those instances of it that we have introduced previously, and to point out some of the implications of these connections. Graphical models have been studied in significant depth in statistics and related areas of computer science, and it is not our intent to provide a comprehensive overview of the topic. For a more in-depth coverage, there are various sources that the interested reader may consult. Useful survey articles include Jordan [216] and Heckerman [195], for example, while full-length treatments may be found in the books by Edwards [129], Lauritzen [248], and Whittaker [403].

10.2 Defining Graphical Models

A *graphical model* is a representation for a joint probability distribution in terms of a graph and a corresponding set of functions defined with respect to that graph. The graph encodes a set of conditional independence relations pertaining to the underlying random variables, which allows – under appropriate conditions – for the joint distribution of these random variables to be decomposed in product form (i.e., to be factorized). The specific nature of the individual terms in this product is dictated by the specified functions. The graphical models formalism provides a useful (and general) approach to handling joint probability distributions, even frequently those of high dimension, by enabling the application of techniques and results from graph theory to probability theory. The synergy that emerges from the union of these two areas has been found to be particularly useful with respect to questions of representation and algorithms.

To characterize graphical models as a subtopic of statistical analysis of network data (the topic of this book) would be unfair. Rather, graphical models are better viewed as a complementary topic – one which is distinct, but at times quite useful for the purposes of analyzing network data. The distinction lies in how we interpret the underlying graph. As an illustration, consider the simple three-vertex graph in Figure 10.1. From the perspective adopted in the first nine chapters of this book, we would associate the three vertices, the two edges, and the flows across these vertices and edges with a system of entities, such as people or computers, linked with respect to some notion of ‘relationship,’ such as friendship or physical cables, exchanging information of some type. Random variables would then be defined in correspondence with some measurement(s) on this system, and the graph would facilitate the indexing of these variables. Probability modeling might then be used to describe the behavior of these variables, and the graph-based indexing would enter the expressions for these models accordingly. On the other hand, from the perspective of graphical models, we would associate with the vertices a set of three arbitrary random variables – say X , Y , and Z , respectively, from left to right – and the graph would dictate an assumed conditional independence structure of a probability model for these random variables. In particular, the graph in Figure 10.1 is read to indicate that the random variable X is conditionally independent of Z , given Y .

In order to distinguish between these two notions in which graphs may enter into probability modeling, in this chapter we will use \tilde{G} to indicate a graph corresponding to a graphical model, and G to indicate a network graph, in the sense that it has been used throughout the rest of the book. Broadly speaking, there are two main sub-classes of graphical models, namely *directed graphical models* and *undirected graphical models*, which are based on directed acyclic graphs and undirected graphs \tilde{G} , respectively. The model mentioned just above is a simple example of an undirected graphical model. We will look at each of these model classes separately, beginning with that of directed graphical models.



Fig. 10.1 A simple three-vertex graph.

10.2.1 Directed Graphical Models

Let X_1, \dots, X_N be a collection of N random variables. A directed graphical model for this collection allows for the decomposition of the joint distribution \mathbb{P} of X_1, \dots, X_N , with respect to a set of conditional independence relations indexed by parent-child vertex combinations in a directed acyclic graph (DAG), say $\tilde{G} = (\tilde{V}, \tilde{E})$, where $\tilde{V} = \{1, \dots, N\}$. This decomposition in turn implies a certain Markov property among the random variables X_i .

More precisely, let \tilde{G} be a DAG and let $\{f(x_i | \mathbf{x}_{pa(i)})\}_{i \in \tilde{V}}$ be a set of conditional probability density (mass) functions for each random variable X_i , given its parents¹ $\mathbf{X}_{pa(i)}$. A directed graphical model defines a joint probability density (mass) function for \mathbb{P} in the form

$$f(x_1, \dots, x_N) = \prod_{i \in \tilde{V}} f(x_i | \mathbf{x}_{pa(i)}) . \quad (10.1)$$

Conversely, given a joint distribution function \mathbb{P} , we say that a pre-specified graphical model represents \mathbb{P} if (10.1) holds. More formally, we say in this case that \mathbb{P} admits a recursive factorization with respect to \tilde{G} .

An important implication of this factorization is that every variable X_i is conditionally independent, given its parents $\mathbf{X}_{pa(i)}$, of all other variables X_j that are neither the parents nor descendants of X_i . This property is known formally as the *Markov condition*, and in fact it can be shown that the factorization and this condition hold in an if-and-only-if manner. See, for example, Lauritzen [248, Ch. 3.2.2]. Informally, the Markov condition usually is interpreted as saying that each X_i , given its parents, is conditionally independent of its other ancestors.

¹ Recall that $pa(i)$ denotes the index set of the parents of i in \tilde{V} .

The practical implication of factorizations like that in (10.1) is to indicate for each variable X_i a certain subset $pa(i)$ of indices j whose random variables X_j directly ‘influence’ X_i . Since often in applications the size of this subset is, either by nature or design, substantially smaller than the total number of random variables N , this feature can have important algorithmic consequences when performing inference in graphical models. We will revisit this point below in Section 10.3.

Specific directed graphical models are distinguished from each other by (i) the topology of their underlying DAG and (ii) the form of their conditional density (mass) functions $f(x_i | \mathbf{x}_{pa(i)})$. A common – and algorithmically attractive – choice for the graph topology is a tree. As for the conditional density (mass) functions, for discrete random variables, multinomial conditional distributions are popular, while for continuous random variables, conditional Gaussian distributions are frequently used. We have already seen some examples of such directed graphical models in earlier chapters.

Example 10.1 (Tomographic Inference of Tree Topologies). From Section 7.4.1, recall the problem of inferring the topology of a tree $G_T = (V_T, E_T)$, with root r , from measurements taken at its leaves $R \subset V_T$. Our primary illustration of this problem was the task of inferring computer network topologies from multicast probe measurements. The outcome of whether or not a probe reached each of the N_l leaves in R , when sent from the root r , was represented by an N_l -tuple $\{X_1, \dots, X_{N_l}\}$. Moreover, to represent the full history of the probe within the tree G_T , we defined a cascade process $\{X_j\}_{j \in V_T}$, where $X_j = 1$ if vertex j received a copy of the probe, and $X_j = 0$, otherwise. This cascade process was then equipped with a particular probabilistic model dictating the manner in which the process was assumed to evolve from the root to the leaves.

Specifically, we assumed that $X_r = 1$ and, for each internal vertex k , that

$$\mathbb{P}(X_j = 1 | X_k = 1) = 1 - \mathbb{P}(X_j = 0 | X_k = 1) = \alpha_j , \quad (10.2)$$

where $0 < \alpha_j < 1$ for all j . Otherwise, if $X_k = 0$, then $X_j = 0$ for all descendants j of k . This model can be expressed in the form of a directed graphical model as follows. Let $\tilde{G} = (\tilde{V}, \tilde{E})$ be a directed version of the graph G_T , where $\tilde{V} = V_T$ and \tilde{E} has a directed edge if and only if E_T has an edge, with the direction of each such edge being away from the root and towards the leaves. Then we can write

$$\mathbb{P}(\mathbf{X} = \mathbf{x}) = \prod_{j \in \tilde{V}} \mathbb{P}(X_j = x_j | X_{pa(j)} = x_{pa(j)}) , \quad (10.3)$$

where

$$\mathbb{P}(X_j = x_j | X_{pa(j)} = x_{pa(j)}) = \begin{cases} 1, & \text{if } j = r \text{ and } x_j = 1 , \\ \alpha_j^{x_j} (1 - \alpha_j)^{1-x_j} , & \text{if } x_{pa(j)} = 1 , \\ 1, & \text{if } x_{pa(j)} = 0 \text{ and } x_j = 0 . \end{cases} \quad (10.4)$$

The expression for the conditional probabilities in (10.4) is defined in analogy to the term in (7.52). Note that $X_{pa(j)}$ in (10.3) is a random variable – instead of a random

vector, as in (10.1) – as a consequence of the fact that non-root vertices have only single parents in a tree. \square

Example 10.2 (Kalman Filtering). Recall that a traffic matrix summarizes the volume of traffic flowing between origin-destination pairs on a network graph $G = (V, E)$. In Section 9.3 we studied various methods for inferring characteristics of traffic matrices from measurements of link volumes $\mathbf{X} = (X_e)_{e \in E}$. Among the dynamic methods of traffic matrix estimation we discussed was the method of Kalman filtering. Under slightly modified conditions, the model underlying this method can be represented as a graphical model with respect to a particular type of binary tree.

To see this, recall that our model specified that the time-indexed link volumes $\mathbf{X}^{(t)}$ and the expected origin-destination traffic volumes $\mu^{(t)}$ evolve together with respect to the equations

$$\mu^{(t+1)} = \Xi^{(t)} \mu^{(t)} + \eta^{(t)} \quad (10.5)$$

$$\mathbf{X}^{(t)} = \mathbf{B}^{(t)} \mu^{(t)} + \varepsilon^{(t)} . \quad (10.6)$$

These equations are simply (9.44) and (9.45), repeated here for convenience. Our previous assumptions on the noise terms $\eta^{(t)}$ and $\varepsilon^{(t)}$ were that they be zero-mean random vectors, with covariances $\Psi^{(t)}$ and $\Sigma^{(t)}$, respectively, and that they be uncorrelated both with each other and among themselves across times t . If we now make the additional assumption that these terms have multivariate Gaussian distributions, then this lack of correlation is equivalent to independence, which in turn induces a set of conditional independence statements among the variables $\mu^{(t)}$ and $\mathbf{X}^{(t)}$.

Specifically, we have that $\mu^{(t+1)}$ is conditionally independent of $\mu^{(0)}, \dots, \mu^{(t-1)}$, given $\mu^{(t)}$, and furthermore, that given $\mu^{(t)}$, $\mathbf{X}^{(t)}$ is conditionally independent of all other terms $\mu^{(s)}$ and $\mathbf{X}^{(s)}$. Therefore, formally, we can write

$$f\left(\mu^{(0)}, \dots, \mu^{(\tau)}, \mathbf{x}^{(1)}, \dots, \mathbf{x}^{(\tau)}\right) = f\left(\mu^{(0)}\right) \prod_{t=1}^{\tau} f\left(\mu^{(t)} | \mu^{(t-1)}\right) f\left(\mathbf{x}^{(t)} | \mu^{(t)}\right) , \quad (10.7)$$

where the third probability density function on the right-hand side corresponds to a multivariate Gaussian distribution, with parameters

$$\mathbb{E}(\mathbf{X}^{(t)} | \mu^{(t)}) = \mathbf{B}^{(t)} \mu^{(t)} \quad \text{and} \quad \mathbb{V}(\mathbf{X}^{(t)} | \mu^{(t)}) = \Sigma^{(t)} , \quad (10.8)$$

and the second, to a multivariate Gaussian distribution, with parameters

$$\mathbb{E}(\mu^{(t)} | \mu^{(t-1)}) = \Xi^{(t-1)} \mu^{(t-1)} \quad \text{and} \quad \mathbb{V}(\mu^{(t)} | \mu^{(t-1)}) = \Psi^{(t-1)} . \quad (10.9)$$

The graph \tilde{G} corresponding to this factorization can be represented as a binary graph for which, among the left- and right-hand children of any parent, only the right-hand child has descendants. Figure 10.2 provides a visualization of this model for the case $\tau = 2$. More generally, this type of graph corresponds to the class of (*first order*) *hidden Markov models* (HMMs), of which our model is just one instance. For background on HMMs and related models, see MacDonald and Zucchini [266]. \square

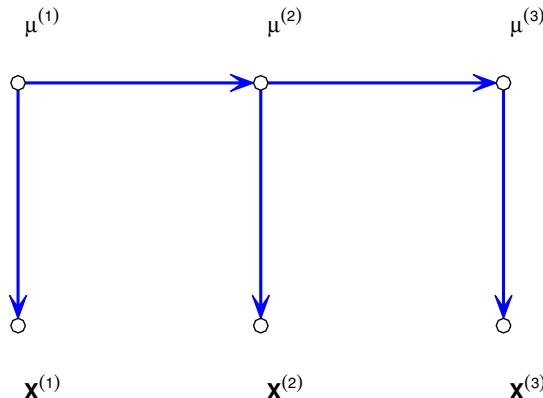


Fig. 10.2 Graphical representation of the model underlying Kalman filtering.

In considering these examples, two points are worth noting. First, the graph \tilde{G} defining a graphical model may or may not correspond to the ‘network graph’ G , as we have referred to it throughout this book. In Example 10.1, \tilde{G} and G are essentially the same. Indeed, one could argue that a sense of directedness was implicit throughout the exposition of Section 7.4.1. On the other hand, in Example 10.2, the graphs \tilde{G} and G are not at all the same, in that \tilde{G} corresponds to the temporal evolution of the relevant traffic volume random variables, while the network graph G embodies the ‘spatial’ system over which the traffic actually flows.

Second, although the graphical modeling formalism may not have been necessary to our previous exposition of the models in these two examples, it can be useful in conceptualizing extensions of the models. For example, in the multicast setting, we might treat the parameters α_j as random variables. In the simplest case, we could specify that the α_j be independent random variables, and independent of all X_j as well. As a result, since the X_j are then dependent not just on their parents but also on the corresponding α_j , our original DAG would be augmented to include additional vertices for each α_j and directed edges from each of these vertices to that of its respective X_j . More sophisticated models for the α_j would lead to graphs \tilde{G} with more involved topologies.

10.2.2 Undirected Graphical Models

Let $\tilde{G} = (\tilde{V}, \tilde{E})$ be an undirected graph. Recall that a clique is a complete sub-graph H , and that H is a maximal clique if no other such sub-graph contains it. Let \mathcal{C} be the set of maximal cliques in \tilde{G} . An undirected graphical model for a set of discrete random variables X_1, \dots, X_N is a model that specifies a probability mass function for their joint distribution \mathbb{P} in the form

$$f(x_1, \dots, x_N) = \left(\frac{1}{\kappa} \right) \prod_{H \in \mathcal{C}} \psi_H(\mathbf{x}_H) , \quad (10.10)$$

where $\psi_H(\mathbf{x}_H)$ is a so-called *potential function*, which is a positive (but otherwise arbitrary) function of the variables $\{x_i\}_{i \in H}$, and

$$\kappa = \sum_{\mathbf{x}} \prod_{H \in \mathcal{C}} \psi_H(\mathbf{x}_H) \quad (10.11)$$

is a normalization factor. The model is defined analogously when the X_i are continuous random variables.

The factorization in (10.10) can be shown to imply that \mathbb{P} satisfies a so-called *global Markov property*. That is, for $A, B, C \subset \tilde{V}$, the vector of random variables $\mathbf{X}_A = (X_i)_{i \in A}$ is independent of $\mathbf{X}_B = (X_i)_{i \in B}$, given $\mathbf{X}_C = (X_i)_{i \in C}$, if and only if the subset C separates the subsets A and B in the graph \tilde{G} , in the sense that every path between a vertex in A and a vertex in B intersects a vertex in C . This property in turn is equivalent to the so-called *pairwise Markov property*, which states that X_i is conditionally independent of X_j , given the other X_k , for $k \in \tilde{V} \setminus \{i, j\}$, if and only if there is no edge between i and j in \tilde{G} . In fact, the converse can be true as well here, in that, under appropriate conditions, if a distribution \mathbb{P} satisfies the pairwise Markov condition, it factorizes with respect to \tilde{G} , as in (10.10). This latter result is generally attributed to Hammersley and Clifford [191], who proved it for the case of discrete random variables. See Lauritzen [248, Ch. 3.2] for details on this and the rest of the above discussion.

Analogous to the case of directed graphical models, specific undirected graphical models are distinguished by the choice of the undirected graph \tilde{G} and the potential functions ψ_H . We have seen a number of examples of undirected graphical models in previous chapters.

Example 10.3 (Markov Random Fields). In Section 8.3.1 we introduced a class of Markov random field models on a network graph $G = (V, E)$, for a vector of attributes $\mathbf{X} = (X_i)_{i \in V}$ on the vertices $i \in V$. Specifically, a discrete random vector \mathbf{X} was said to be a Markov random field on G if its probability mass function obeyed the condition

$$\mathbb{P}(X_i = x_i | \mathbf{X}_{(-i)} = \mathbf{x}_{(-i)}) = \mathbb{P}(X_i = x_i | \mathbf{X}_{\mathcal{N}_i} = \mathbf{x}_{\mathcal{N}_i}) , \quad (10.12)$$

where recall that $\mathbf{X}_{(-i)}$ is the vector $(X_1, \dots, X_{i-1}, X_{i+1}, \dots, X_{N_v})^T$ and $\mathbf{X}_{\mathcal{N}_i}$ is the vector of all X_j for $j \in \mathcal{N}_i$, with \mathcal{N}_i defined to be the neighborhood of i in G .

The condition in (10.12) is equivalent to the global Markov condition. Furthermore, under appropriate conditions, the Hammersley-Clifford theorem establishes the equivalence of the full joint distribution of \mathbf{X} with the so-called Gibbs distribution in (8.5). See Besag [36]. In particular, recall that under such conditions we have

$$\mathbb{P}(\mathbf{X} = \mathbf{x}) = \left(\frac{1}{\kappa} \right) \exp \{U(\mathbf{x})\} , \quad (10.13)$$

where $U(\cdot)$ is the energy function and κ is a normalization factor (often called the partition function), and where the energy function can be decomposed as a sum over all (maximal) cliques $H \in \mathcal{C}$, in the form

$$U(\mathbf{x}) = \sum_{H \in \mathcal{C}} U_H(\mathbf{x}) . \quad (10.14)$$

In other words, the Markov random field model can be expressed in the form of (10.10), with $\tilde{G} = G$ and potential functions defined as

$$\psi_H(\mathbf{x}_H) = \exp \{U_H(\mathbf{x})\} . \quad (10.15)$$

Several choices of such potential functions were provided in Example 8.2. We note that undirected graphical models are often referred to synonymously with Markov random fields. \square

Example 10.4 (Gaussian Graphical Models). Recall from Section 7.3.3 our discussion of Gaussian graphical models for continuous attributes X_i corresponding to elements $i \in \{1, \dots, N\}$. In particular, suppose that $\mathbf{X} = (X_i)_{i=1}^N$ has a multivariate Gaussian distribution, with expectation $\mathbb{E}(\mathbf{X}) = \mu$ and variance $\mathbb{V}(\mathbf{X}) = \Sigma$, and let $\Omega = \Sigma^{-1}$ denote the precision matrix. The graph $G = (V, E)$ was defined so that $V = \{1, \dots, N\}$ and

$$E = \left\{ \{i, j\} \in V^{(2)} : \rho_{ij|V \setminus \{i, j\}} \neq 0 \right\} , \quad (10.16)$$

where

$$\rho_{ij|V \setminus \{i, j\}} = \frac{-\omega_{ij}}{\sqrt{\omega_{ii}\omega_{jj}}} \quad (10.17)$$

is the partial correlation of X_i and X_j adjusted for all other X_k , $k \in V \setminus \{i, j\}$. Since a partial correlation of zero is equivalent to conditional independence in the Gaussian setting, the graph G – the conditional independence graph – encodes the set of pairwise Markov relations among the X_i .

The impact of these conditional independence relations is evident, for example, in the log-likelihood, the quadratic component for which, without loss of generality, in the case of $\mu = 0$ takes the form

$$\mathbf{x}^T \boldsymbol{\Omega} \mathbf{x} = \sum_{i \in V} \omega_{ii} x_i^2 + 2 \sum_{\{i,j\} \in E} \omega_{ij} x_i x_j . \quad (10.18)$$

However, this expression does not itself directly yield a factorization like that in (10.10). Some additional work is necessary here, through which it can be shown that it is possible to write

$$f(\mathbf{x}) = \frac{\prod_{H \in \mathcal{C}} f(\mathbf{x}_H)}{\prod_{S \in \mathcal{S}} [f(\mathbf{x}_S)]^{v(S)}} , \quad (10.19)$$

where $f(\mathbf{x}_A)$ is the joint density of $\mathbf{X}_A = (X_i)_{i \in A}$, \mathcal{S} is a set of so-called separators indexed according to a particular ordering (i.e., a so-called perfect sequence) of the cliques H in \mathcal{C} , and $v(S)$ is the number of times the separator S occurs in the sequence. See Lauritzen [248, Ch. 5.3.1]. \square

Example 10.5 (Exponential Random Graph Models). The exponential random graph models (ERGMs) introduced in Section 6.5 provide a general approach to modeling the dependencies among edges in a network graph. Recall that these models specify an exponential family distribution for a random matrix \mathbf{Y} , where the latter describes the adjacencies in an undirected random graph G . And note that the expression for $\mathbb{P}(\mathbf{Y} = \mathbf{y})$ in (6.24) is qualitatively quite similar to that in (10.10). However, importantly, the summation in (6.24) is over an arbitrary set of ‘configurations’ (i.e., possible sets of edges among subsets of vertices in the network graph G), whereas that in (10.10) is specifically over maximal cliques in a conditional independence graph \tilde{G} . Under certain specifications, an ERGM will satisfy the appropriate Markov conditions needed to be equivalent to an undirected graphical model. On the other hand, many ERGM formulations do not satisfy these conditions, yet continue – by definition – to retain an exponential family form.

Frank and Strauss [155] introduced the notion of a pairwise Markov property to the modeling of random network graphs G , specifying that the edge random variables Y_{ij} and $Y_{j'j'}$ be conditionally independent, given the other random variables $Y_{k,l}$, if and only if they do not share a vertex. This condition can be represented using a conditional independence graph $\tilde{G} = (\tilde{V}, \tilde{E})$, such that each unordered pair of vertices $\{i, j\} \in V^{(2)}$ corresponds to a vertex in \tilde{V} , and pairs of vertices in \tilde{V} have an edge between them in \tilde{E} if and only if the corresponding variables in \mathbf{Y} are conditionally dependent. Using the Hammersley-Clifford theorem and the type of homogeneity assumption discussed in Section 6.5, Frank and Strauss establish that the distribution of \mathbf{Y} satisfies this pairwise Markov property if and only if

$$\mathbb{P}(\mathbf{Y} = \mathbf{y}) = \left(\frac{1}{\kappa} \right) \exp \left\{ \sum_{k=1}^{N_v-1} \theta_k S_k(\mathbf{y}) + \theta_\tau T(\mathbf{y}) \right\} , \quad (10.20)$$

where $S_1(\mathbf{y}) = N_e$ is the number of edges, $S_k(\mathbf{y})$ is the number of k -stars, and $T(\mathbf{y})$ is the number of triangles. Recall that such Markov random graphs also include the Bernoulli random graph, discussed in Example 6.6, as a special case.

More generally, other ERGM formulations, such as those based on the alternating k -stars and alternating k -triangles defined in (6.31) and (6.33), respectively, do not satisfy the pairwise Markov condition and hence cannot be represented as in (10.10). Nevertheless, it is possible to conceptualize these models – and others like them – within the framework of a somewhat broader notion of conditional independence, that of ‘partial conditional dependence.’ See Pattison and Robins [311] and Robins and Pattison [330] for details. \square

As in the case of directed graphical models, the framework of undirected graphical models can be useful in facilitating extensions of existing models. See, for example, Robins, Pattison, and Elliot [331], who offer an extension of the Hammersley-Clifford theorem in the context of joint and conditional models for a network adjacency matrix \mathbf{Y} and a vector \mathbf{X} of vertex attributes.

10.3 Inference for Graphical Models

There are a variety of tasks in graphical modeling that can be referred to, broadly speaking, as ‘inferential’ tasks – or perhaps more generally, as tasks of statistical learning. Such tasks can range from being relatively straightforward to intractable. The complexity of inferential tasks varies as a function of both graph topology (i.e., model complexity) and the task itself. From a computational point of view, statistical inference in this area is looked at as being fundamentally a problem of computing relevant conditional – or, more generally, marginal – probabilities. That is, given a graphical model, with graph $\tilde{G} = (\tilde{V}, \tilde{E})$ and joint probability mass (density) function f , we wish to compute

$$f(\mathbf{x}_A) = \sum_{\mathbf{x}' : \mathbf{x}'_A = \mathbf{x}_A} f(\mathbf{x}') , \quad (10.21)$$

for some $A \subset \tilde{V}$. Quantities like means and other values involving expectations can be obtained from such probabilities accordingly.

There are three main categories of methods for graphical model inference: exact methods, sampling methods, and variational methods. We characterize each of these methods briefly in the paragraphs that follow. For more details, see the survey article by Jordan [216], the edited volume by Jordan [215], and the many various references therein.

Exact methods of inference, as their name indicates, seek to compute the requisite marginal probabilities exactly. The primary hurdle to this computation is the summation (or, in the continuous case, integration) over all possible combinations of values for the variables $\{X_i\}_{i \in \tilde{V} \setminus A}$ not appearing in $f(\mathbf{x}_A)$. Counting terms in the summation, it is easy to see that if each X_i has a range of r values, the computational complexity of naively evaluating just one such probability can be seen to scale like $O(r^m)$, where $m = |\tilde{V} \setminus A|$. Therefore, algorithms in this area seek to increase computational efficiency through exploitation of the structure of the independence graph

\tilde{G} . The computational gains realized by such techniques are particularly important when more than one probability must be computed, for instance, when the computation is just one element of a larger optimization algorithm.

One well-known example of such algorithms is the *sum-product* algorithm, which allows for exact computation on tree-structured graphs. This algorithm leverages the fact that the joint probability mass function $f(\mathbf{x})$ can be written in product form (i.e., as in (10.1) or (10.10)), by employing a pattern of inter-changes of summation and product in marginalizing $f(\mathbf{x})$ to obtain $f(\mathbf{x}_A)$, so as to eliminate redundancy entailed by re-calculation of intermediate partial sums. A related algorithm, and essentially a special case of the sum-product algorithm, is *belief propagation* (Pearl [312] and Lauritzen and Spiegelhalter [249]), which performs exact inference on directed acyclic graphs. Also related is the *max-sum* algorithm, which uses an analogous strategy to maximize a probability function and return the corresponding argument. These types of algorithms are based on principles of dynamic programming and are sometimes called *message-passing* algorithms, since the computational process can be diagrammed as a sequence of steps in which ‘messages’ are ‘passed’ locally between adjacent vertices. A gentle introduction to these algorithms may be found in Bishop [39, Ch. 8.4], who also talks briefly about their extension for doing approximate inference in more general graphs (i.e., other than trees or DAGs).

Implementation of exact algorithms can require a certain amount of preliminary preparation. For example, they tend to be defined in terms of undirected graphs; their usage with directed graphs actually involves an initial conversion of the graph to a particular undirected graph called the *moral graph*. In addition, generally there is a choice in the manner in which the calculations are organized. For example, there can be multiple possible orders in which the alternating sums and products are performed in the sum-product algorithm. Finding an optimal ordering that minimizes the overall computational complexity is in general an *NP-hard* problem, but various heuristics exist. Jordan [216, Sec. 3.1] provides some discussion.

Exact algorithms are always in principle available, but in practice may be infeasible, depending on the order and structure of the graph underlying the graphical model. In such settings, methods of approximate inference become attractive. *Sampling methods* offer one such alternative, and Gibbs sampling algorithms are a standard choice in this area. Recall that the basic Gibbs sampling algorithm involves only the computation of univariate conditional probabilities of the form $f(x_i | \mathbf{x}_{(-i)})$. Because of the Markov properties associated with graphical models, these computations typically reduce to computing $f(x_i | \mathbf{x}_{\mathcal{N}_i})$. That is, the graphical model structure dictates a usually reduced set of variables to condition upon, for each variable X_i . We saw a detailed discussion of Gibbs sampling for inference of parameters and predictions in Markov random field models in Section 8.3.1. See Jordan [216, Sec. 3.2] for additional details and references.

Another class of methods of approximate inference in graphical models, and one that has seen some notable research activity in the past decade, is that of *variational methods*. These methods can be viewed as characterizing the calculation of probabilities as an optimization problem and proposing to solve a perturbed (and

more tractable) form of that problem, rather than the original problem itself. Such perturbed problems can be obtained, for example, by ignoring certain of the dependencies indicated by the graphical model and working instead with a probability distribution of simpler dependency structure. For details, we refer the reader to Jordan [216, Sec. 3.3] and the references therein.

10.4 Additional Related Topics and Reading

There are other representations for graphical models encountered in the literature besides the directed and undirected graphs we have focused upon. Two important examples are the class of *chain graphs*, which combine undirected and directed edges within a single graph, and *factor graphs*, which are bipartite graphs that provide a more nuanced factorization in the undirected case. Also, it should be noted that terminology varies somewhat in this area. For example, directed graphical models often are called *Bayesian networks*, although they are used in both frequentist and Bayesian settings. And, as was mentioned previously, undirected graphical models sometimes are referred to alternatively as Markov random fields. Finally, we mention that there are various related classes of probabilistic models that are relevant to the analysis of network data. The class of *probabilistic relational models*, for example, as developed largely by Koller and colleagues, can be viewed as an extension of directed graphical models to include relations as variables. A comprehensive overview of this more relatively recent class of models can be found in the volume edited by Getoor and Taskar [168].

Glossary of Notation

Graph Notation

$G = (V, E)$	A graph G , with vertex set V and edge set E .
N_v, N_e	Number of vertices; number of edges.
\mathcal{G}	A collection of graphs.
$V^m, V^{(m)}$	Set of ordered/unordered vertex m -tuples.
$V_{\setminus \{i,j\}}^{(m)}$	Set of unordered vertex m -tuples from $V \setminus \{i, j\}$.
$\mathbf{A} = [A_{ij}]_{i,j=1}^{N_v}$	Adjacency matrix.
B	Routing matrix (more generally, incidence matrix).
$\mathbf{L}, \tilde{\mathbf{L}}$	Laplacian and normalized Laplacian matrices.
$\{d_v\}_{v \in V}$	Degree sequence.
$\text{dist}(v, u)$	Geodesic distance between v and u .
$\bar{l}(G)$	Average geodesic distance on G .
$c_{Cl}(v), c_B(v), c_{Ei}(v)$	Closeness, betweenness, and eigenvector centrality of v .
$\text{den}(G)$	Density of G .
$\tau_3(G)$	Number of connected triples of vertices in G .
$\tau_\Delta(G)$	Number of triangles in G .
$\text{cl}(G), \text{cl}_T(G)$	Clustering and clustering-transitivity coefficient of G .
$\mathcal{C} = \{C_1, \dots, C_K\}$	A partition (e.g., of a set of vertices).
$\text{mod}(\mathcal{C})$	Modularity of the partition \mathcal{C} .
$\mathcal{N}(S)$	Neighborhood of vertices in S .
$\mathcal{N}_v = \mathcal{N}(\{v\})$	Neighborhood of vertex v .
$\mathcal{N}_v^+ = \mathcal{N}_v \cup \{v\}$	Union of vertex v with its neighborhood.
$\eta(G)$	Characteristic of G (e.g., average degree, clustering coefficient, etc.).
$G^* = (V^*, E^*)$	Graph obtained by sampling G .
$T = (V_T, E_T)$	A tree.

Probability and Statistics Notation

$\mathbb{P}(B)$	Probability of the event B .
X, \mathbf{X}	Random variable; random vector or matrix.
x, \mathbf{x}	Observation of X or \mathbf{X} .
$\mathbf{X}_{(-i)}, \mathbf{X}^{(-i)}$	Random vector \mathbf{X} without its i -th entry.
I_B	Indicator random variable for the event B .
F_X	Cumulative distribution function (CDF), i.e., $F_X(x) = \mathbb{P}(X \leq x)$.
f_X	Probability density (or mass) function.
$\mathbb{E}(X)$	Expected value (mean) of X ; sometimes written μ_X .
$\mathbb{V}(X)$	Variance of X i.e., $\mathbb{E}((X - \mu_X)^2)$
$\text{Cov}(X, Y)$	Covariance of X and Y i.e., $\mathbb{E}((X - \mu_X)(Y - \mu_Y))$
$\text{corr}(X, Y)$	Correlation of X and Y i.e., $\text{Cov}(X, Y)/(\mathbb{V}(X)\mathbb{V}(Y))^{1/2}$.
\bar{X}, \bar{x}	Sample mean; observed sample mean.
$\theta, \hat{\theta}$	Parameter; estimate of parameter.
$\text{se}(\hat{\theta})$	Standard error of $\hat{\theta}$ i.e., $\text{se}(\hat{\theta}) = \sqrt{\mathbb{V}(\hat{\theta})}$.
$\text{MSE}(\hat{\theta})$	Mean squared error of $\hat{\theta}$ i.e., $\text{MSE}(\hat{\theta}) = \mathbb{E}((\hat{\theta} - \theta)^2)$.
$\mathcal{L}(\theta)$	Likelihood function.
$\ell(\theta)$	Log-likelihood function i.e., $\ell(\theta) = \log \mathcal{L}(\theta)$.
RSS	Residual sum of squares.
FDR	False discovery rate.

Other Notation

$n!$	$n \times (n-1) \times \dots \times 2 \times 1$
$\binom{n}{k}$	$n!/(k!(n-k)!)$
\mathbb{R}	The real numbers.
$ S $	Number of elements in the set S .
$h_n = O(g_n)$	$ h_n/g_n $ is bounded for large n .
NP	A formal characterization in computer science of the difficulty of a problem, effectively indicating that it may be solved only in a number of computations growing exponentially with the size of the problem.
\mathbf{x}^T	Transpose of the vector \mathbf{x} .

References

- [1] “The R Project for Statistical Computing,” <http://www.r-project.org/>.
- [2] “Kyoto Encyclopedia of Genes and Genomes,” available at <http://www.genome.ad.jp/kegg/keggI.html>.
- [3] D. Achlioptas, A. Clauset, D. Kempe, and C. Moore, “On the bias of trace-route sampling,” *Proceedings of the 37th Annual ACM Symposium on Theory of Computing*, pp. 694–703, 2005.
- [4] L. Adamic and E. Adar, “Friends and neighbors on the Web,” *Social Networks*, vol. 25, no. 3, pp. 211–230, 2003.
- [5] W. Aiello, F. Chung, and L. Lu, “A random graph model for power law graphs,” *Experimental Mathematics*, vol. 10, no. 1, pp. 53–66, 2001.
- [6] R. Albert and A. Barabási, “Statistical mechanics of complex networks,” *Reviews of Modern Physics*, vol. 74, no. 1, pp. 47–97, 2002.
- [7] M. Altaf-Ul-Amin, K. Nishikata, T. Koma, T. Miyasato, Y. Shinbo, M. Arifuzzaman, C. Wada, M. Maeda, T. Oshima, H. Mori *et al.*, “Prediction of protein functions based on k -cores of protein-protein interaction networks and amino acid sequences,” *Genome Informatics*, vol. 14, pp. 498–499, 2003.
- [8] J. Alvarez-Hamelin, L. Dall’Asta, A. Barrat, and A. Vespignani, “Large scale networks fingerprinting and visualization using the k -core decomposition,” *Advances in Neural Information Processing Systems, NIPS*, vol. 5, p. 18, 2005.
- [9] N. Amenta, F. Clarke, and K. St. John, “A linear-time majority tree algorithm,” in *Proceedings of the Workshop on Algorithms in Bioinformatics, 2003*, ser. Lecture Notes in Computer Science, vol. 2812. Berlin: Springer-Verlag, 2003, pp. 216–227.
- [10] R. Anderson and R. May, *Infectious Diseases of Humans: Dynamics and Control*. Oxford: Oxford University Press, 1991.
- [11] T. Anderson, *An Introduction to Multivariate Statistical Analysis, Second Edition*. New York: John Wiley & Sons, Inc., 1984.
- [12] H. Andersson, “Limit theorems for a random graph epidemic model,” *Annals of Applied Probability*, vol. 8, no. 4, pp. 1331–1349, 1998.

- [13] H. Andersson and T. Britton, *Stochastic Epidemic Models and Their Statistical Analysis*. New York: Springer, 2000.
- [14] A. Argyriou, M. Herbster, and M. Pontil, “Combining graph Laplacians for semi-supervised learning,” *Advances in Neural Information Processing Systems*, vol. 18, 2006.
- [15] G. Bader and C. Hogue, “An automated method for finding molecular complexes in large protein interaction networks,” *BMC Bioinformatics*, vol. 4, no. 2, 2003.
- [16] J. Bader, A. Chaudhuri, J. Rothberg, and J. Chant, “Gaining confidence in high-throughput protein interaction networks,” *Nature Biotechnology*, vol. 22, no. 1, pp. 78–85, 2004.
- [17] F. Ball, D. Mollison, and G. Scalia-Tomba, “Epidemics with two levels of mixing,” *Annals of Applied Probability*, vol. 7, no. 1, pp. 46–89, 1997.
- [18] A. Barabási and R. Albert, “Emergence of scaling in random networks,” *Science*, vol. 286, no. 5439, pp. 509–512, 1999.
- [19] A. Barrat and M. Weigt, “On the properties of small-world network models,” *European Physical Journal B*, vol. 13, pp. 547–560, 2000.
- [20] A. Barrat, M. Barthélemy, and A. Vespignani, *Dynamical Processes on Complex Networks*. New York: Cambridge University Press, 2008.
- [21] J.-P. Barthélemy and F. McMorris, “The median procedure for n -trees,” *Journal of Classification*, vol. 3, pp. 329–334, 1986.
- [22] V. Batagelj, “Generalized Ward and related clustering problems,” in *Classification and Related Methods of Data Analysis*, H. Bock, Ed. Amsterdam: North-Holland, 1988, pp. 67–74.
- [23] V. Batagelj and U. Brandes, “Efficient generation of large random networks,” *Physical Review E*, vol. 71, no. 3, p. 36113, 2005.
- [24] V. Batagelj and A. Mrvar, “Pajek – Program for Large Network Analysis,” <http://vlado.fmf.uni-lj.si/pub/networks/pajek/>.
- [25] V. Batagelj, A. Mrvar, and M. Zaversnik, “Partitioning approach to visualization of large networks,” in *Proceedings of the 7th International Symposium on Graph Drawing (GD'99)*, ser. Lecture Notes in Computer Science, J. Kratochvíl, Ed., vol. 1731. Berlin: Springer-Verlag, 1999, pp. 90–98.
- [26] M. Baur, U. Brandes, M. Gaertler, and D. Wagner, “Drawing the AS graph in 2.5 dimensions,” in *Proceedings of the 12th International Symposium on Graph Drawing*, ser. Lecture Notes in Computer Science, J. Pach, Ed., vol. 3383. Berlin: Springer-Verlag, 2005, pp. 43–48.
- [27] N. Becker and T. Britton, “Statistical studies of infectious disease incidence,” *Journal of the Royal Statistical Society: Series B*, vol. 61, no. 2, pp. 287–307, 1999.
- [28] M. Belkin and P. Niyogi, “Laplacian eigenmaps for dimensionality reduction and data representation,” *Neural Computation*, vol. 15, no. 6, pp. 1373–1396, 2003.
- [29] M. Belkin, I. Matveeva, and P. Niyogi, “Regularization and semi-supervised learning on large graphs,” in *Proceedings of the 17th Annual Conference on Learning Theory (COLT)*, ser. Lecture Notes in Artificial Intelligence,

- J. Shawe-Taylor and Y. Singer, Eds., vol. 3120. Berlin Heidelberg: Springer-Verlag, 2004, pp. 624–638.
- [30] M. Bell, “The estimation of an origin-destination matrix from traffic counts,” *Transportation Science*, vol. 17, no. 2, pp. 198–217, 1983.
- [31] ——, “The estimation of origin-destination matrices by constrained generalized least squares,” *Transportation Research Part B*, vol. 25B, pp. 13–22, 1991.
- [32] M. Bell and Y. Iida, *Transportation Network Analysis*. New York: Wiley, 1997.
- [33] Y. Benjamini and Y. Hochberg, “Controlling the false discovery rate: a practical and powerful approach to multiple testing,” *Journal of the Royal Statistical Society, Series B*, vol. 57, no. 1, pp. 289–300, 1995.
- [34] Y. Benjamini and D. Yekutieli, “The control of the false discovery rate in multiple testing under dependency,” *Annals of Statistics*, vol. 29, no. 4, pp. 1165–1188, 2001.
- [35] J. Berger, B. Liseo, and R. Wolpert, “Integrated likelihood methods for eliminating nuisance parameters,” *Statistical Science*, vol. 14, no. 1, pp. 1–28, 1999.
- [36] J. Besag, “Spatial interaction and the statistical analysis of lattice systems,” *Journal of the Royal Statistical Society, Series B*, vol. 36, no. 2, pp. 192–236, 1974.
- [37] ——, “Statistical analysis of non-lattice data,” *The Statistician*, vol. 24, no. 3, pp. 179–195, 1975.
- [38] P. Bickel, C. Chen, J. Kwon, J. Rice, E. van Zwet, and P. Varaiya, “Measuring traffic,” *Statistical Science*, vol. 22, no. 4, pp. 581–597, 2007.
- [39] C. Bishop, *Pattern Recognition and Machine Learning*. Singapore: Springer, 2006.
- [40] A. Blum and S. Chawla, “Learning from labeled and unlabeled data using graph mincuts,” *Proceedings of the 18th International Conference on Machine Learning*, pp. 19–26, 2001.
- [41] S. Boccaletti, V. Latora, Y. Moreno, M. Chavez, and D. Hwang, “Complex networks: structure and dynamics,” *Physics Reports*, vol. 424, no. 4–5, pp. 175–308, 2006.
- [42] B. Bollobás, *Modern Graph Theory*. New York: Springer, 1998.
- [43] ——, *Random Graphs, Second Edition*. New York: Cambridge University Press, 2001.
- [44] B. Bollobás and O. Riordan, “The diameter of a scale-free random graph,” *Combinatorica*, vol. 24, no. 1, pp. 5–34, 2004.
- [45] ——, “Mathematical results on scale-free random graphs,” in *Handbook of Graphs and Networks: From the Genome to the Internet*, S. Bornholdt and H. Schuster, Eds. Weinheim: Wiley-VCH, 2002, pp. 1–34.
- [46] B. Bollobás, O. Riordan, J. Spencer, and G. Tusnady, “The degree sequence of a scale-free random graph process,” *Random Structures and Algorithms*, vol. 18, no. 3, pp. 279–290, 2001.

- [47] P. Bonacich, “Factoring and weighting approaches to status scores and clique identification,” *Journal of Mathematical Sociology*, vol. 2, no. 1, pp. 113–120, 1972.
- [48] S. Borgatti and M. Everett, “A graph-theoretic perspective on centrality,” *Social Networks*, vol. 28, no. 4, pp. 466–484, 2006.
- [49] M. Bourque, “Arbres de steiner et reseaux dont certains sommets sont à localisation variable,” Ph.D. dissertation, Université de Montréal, Québec, 1978.
- [50] K. Boyack, R. Klavans, and K. Börner, “Mapping the backbone of Science,” *Scientometrics*, vol. 64, no. 3, pp. 351–374, 2005.
- [51] U. Brandes, “A faster algorithm for betweenness centrality,” *Journal of Mathematical Sociology*, vol. 25, no. 2, pp. 163–177, 2001.
- [52] ——, “On variants of shortest-path betweenness centrality and their computation,” *Social Networks*, vol. 30, pp. 136–145, 2008.
- [53] ——, “Layout of graph visualizations,” Ph.D. dissertation, Department of Mathematics and Information, University of Konstanz, Germany, 1999.
- [54] U. Brandes and T. Erlebach, Eds., *Network Analysis: Methodological Foundations*, ser. Lecture Notes in Computer Science. New York: Springer-Verlag, 2005, vol. 3418.
- [55] U. Brandes and D. Fleischer, “Centrality measures based on current flow,” in *Proceedings of the 22nd Symposium on Theoretical Aspects of Computer Science (STACS'05)*, ser. Lecture Notes in Computer Science, V. Diekert and B. Durand, Eds., vol. 3404. Berlin Heidelberg: Springer-Verlag, 2005, pp. 533–544.
- [56] U. Brandes and C. Pich, “Eigensolver methods for progressive multidimensional scaling of large data,” in *Proceedings of the 14th International Symposium on Graph Drawing*, ser. Lecture Notes in Computer Science, M. Kaufmann and D. Wagner, Eds., vol. 4372. Berlin Heidelberg: Springer-Verlag, 2007.
- [57] ——, “Centrality estimation in large networks,” *International Journal of Bifurcation and Chaos*, vol. 17, no. 7, pp. 2303–2318, 2007.
- [58] U. Brandes and D. Wagner, “A Bayesian paradigm for dynamic graph layout,” in *Proceedings of the 5th International Symposium on Graph Drawing*, ser. Lecture Notes in Computer Science, G. DiBattista, Ed., vol. 1353. Berlin: Springer-Verlag, 1997, pp. 236–247.
- [59] ——, “Dynamic grid embedding with few bends and changes,” in *Proceedings of the 9th Annual International Symposium on Algorithms and Computation*, ser. Lecture Notes in Computer Science, K.-Y. Chwa and O. Ibarra, Eds., vol. 1533. Berlin: Springer-Verlag, 1998, pp. 89–98.
- [60] U. Brandes, P. Kenis, J. Raab, V. Schneider, and D. Wagner, “Explorations into the visualization of policy networks,” *Journal of Theoretical Politics*, vol. 11, no. 1, pp. 75–106, 1999.
- [61] U. Brandes, D. Delling, M. Gaertler, R. Görke, M. Hoefer, Z. Nikoloski, and D. Wagner, “On modularity clustering,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 20, no. 2, pp. 172–188, 2008.

- [62] L. Breiman, “Bagging predictors,” *Machine Learning*, vol. 24, pp. 123–140, 1996.
- [63] S. Brin and L. Page, “The anatomy of a large-scale hypertextual Web search engine,” *Computer Networks and ISDN Systems*, vol. 30, no. 1–7, pp. 107–117, 1998.
- [64] T. Britton and P. O’Neill, “Bayesian inference for stochastic epidemics in populations with random social structure,” *Scandinavian Journal of Statistics*, vol. 29, no. 3, pp. 375–390, 2002.
- [65] T. Britton, M. Nordvik, and F. Liljeros, “Modelling sexually transmitted infections: the effect of partnership activity and number of partners on R_0 ,” *Theoretical Population Biology*, vol. 72, pp. 389–399, 2007.
- [66] P. Brockwell and R. Davis, *Introduction to Time Series and Forecasting, Second Edition*. New York: Springer, 2002.
- [67] A. Broder, R. Kumar, F. Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, A. Tomkins, and J. Wiener, “Graph structure in the Web,” *Computer Networks*, vol. 33, no. 1–6, pp. 309–320, 2000.
- [68] D. Bryant, “A classification of consensus methods for phylogenetics,” *Bioconsensus: DIMACS Working Group Meetings on Bioconsensus, October 25–26, 2000 and October 2–5, 2001, DIMACS Center*, pp. 163–183, 2003.
- [69] J. Bunge and M. Fitzpatrick, “Estimating the number of species: a review,” *Journal of the American Statistical Association*, vol. 88, no. 421, pp. 364–373, 1993.
- [70] W. Burk, C. Steglich, and T. Snijders, “Beyond dyadic interdependence: actor-oriented models for co-evolving social networks and individual behaviors,” *International Journal of Behavioral Development*, vol. 31, no. 4, pp. 397–404, 2007.
- [71] R. Caceres, N. Duffield, J. Horowitz, and D. Towsley, “Multicast-based inference of network-internal loss characteristics,” *IEEE Transactions on Information Theory*, vol. 45, no. 7, pp. 2462–2480, 1999.
- [72] G. Caldarelli, R. Pastor-Satorras, and A. Vespignani, “Structure of cycles and local ordering in complex networks,” *European Physical Journal B*, vol. 38, pp. 183–186, 2004.
- [73] J. Cao, D. Davis, S. Wiel, and B. Yu, “Time-varying network tomography: router link data.” *Journal of the American Statistical Association*, vol. 95, no. 452, pp. 1063–1075, 2000.
- [74] H. Carey, *Principles of Social Science*. Philadelphia: Lippincott, 1858.
- [75] E. Cascetta, “Dynamic estimators of origin-destination matrices using traffic counts,” *Transportation Science*, vol. 27, no. 4, pp. 363–373, 1993.
- [76] ———, “Estimation of trip matrices from traffic counts and survey data: a generalized least squares estimator,” *Transportation Research Part B*, vol. 18B, pp. 289–299, 1984.
- [77] E. Cascetta and S. Nguyen, “A unified framework for estimating or updating origin/destination matrices from traffic counts,” *Transportation Research*, vol. 22B, pp. 437–455, 1988.

- [78] R. Castro, M. Coates, G. Liang, R. Nowak, and B. Yu, “Network tomography: recent developments,” *Statistical Science*, vol. 19, no. 3, pp. 499–517, 2004.
- [79] R. Castro, M. Coates, and R. Nowak, “Likelihood-based hierarchical clustering,” *IEEE Transactions on Signal Processing*, vol. 52, no. 8, pp. 2308–2321, 2004.
- [80] Y. Chen, D. Bindel, H. Song, and R. Katz, “An algebraic approach to practical and scalable overlay network monitoring,” *ACM SIGCOMM Computer Communication Review*, vol. 34, no. 4, pp. 55–66, 2004.
- [81] Y. Chen, P. Diaconis, S. Holmes, and J. Liu, “Sequential Monte Carlo methods for statistical analysis of tables,” *Journal of the American Statistical Association*, vol. 100, no. 469, pp. 109–121, 2005.
- [82] B. Chor and T. Tuller, “Finding a maximum likelihood tree is hard,” *Journal of the ACM*, vol. 53, no. 5, pp. 722–744, 2006.
- [83] R. Christensen, *Log-Linear Models and Logistic Regression*. New York: Springer, 1997.
- [84] ———, *Plane Answers to Complex Questions*. New York: Springer, 1987.
- [85] D. Chua, E. Kolaczyk, and M. Crovella, “Network kriging,” *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 12, pp. 2263–2272, 2006.
- [86] C. Chui and G. Chen, *Kalman Filtering with Real-Time Applications*. Berlin: Springer-Verlag, 1991.
- [87] F. Chung and L. Lu, “The average distances in random graphs with given expected degrees,” *Proceedings of the National Academy of Sciences USA*, vol. 99, no. 25, pp. 15,879–15,882, 2002.
- [88] ———, *Complex Graphs and Networks*. American Mathematical Society, 2006.
- [89] F. Chung, L. Lu, T. Dewey, and D. Galas, “Duplication models for biological networks,” *Journal of Computational Biology*, vol. 10, no. 5, pp. 677–687, 2003.
- [90] F. Chung, *Spectral Graph Theory*. American Mathematical Society, 1997.
- [91] A. Clauset and C. Moore, “Accuracy and scaling phenomena in Internet mapping,” *Physical Review Letters*, vol. 94, no. 1, p. 18701, 2005.
- [92] A. Clauset, M. Newman, and C. Moore, “Finding community structure in very large networks,” *Physical Review E*, vol. 70, no. 6, p. 66111, 2004.
- [93] W. Cleveland, *The Elements of Graphing Data*. Wadsworth, 1985.
- [94] ———, *Visualizing Data*. Hobart Press, 1993.
- [95] M. Coates, R. Castro, R. Nowak, M. Gadhiok, R. King, and Y. Tsang, “Maximum likelihood network topology identification from edge-based unicast measurements,” *Proceedings of the 2002 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, pp. 11–20, 2002.
- [96] M. Coates, A. Hero, R. Nowak, and B. Yu, “Internet tomography,” *IEEE Signal Processing Magazine*, vol. 19, no. 3, pp. 47–65, 2002.
- [97] M. Coates, M. Rabbat, and R. Nowak, “Merging logical topologies using end-to-end measurements,” in *Proceedings of the 3rd ACM SIGCOMM Conference on Internet Measurement*. New York: ACM, 2003, pp. 192–203.

- [98] M. Coates, Y. Pointurier, and M. Rabbat, “Compressed network monitoring for IP all-optical networks,” in *Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement*. New York: ACM, 2007, pp. 241–252.
- [99] C. Cooper and A. Frieze, “A general model of Web graphs,” *Random Structures and Algorithms*, vol. 22, no. 3, pp. 311–335, 2003.
- [100] T. Cormen, C. Leiserson, R. Rivest, and C. Stein, *Introduction to Algorithms*. Cambridge, MA: MIT Press, 2003.
- [101] T. Cover and J. Thomas, *Elements of Information Theory*. New York: John Wiley & Sons, 1991.
- [102] N. Cressie, *Statistics for Spatial Data*. New York: Wiley-Interscience, 1993.
- [103] N. Cristianini, J. Shawe-Taylor, A. Elisseeff, and J. Kandola, “On kernel-target alignment,” *Advances in Neural Information Processing Systems 14*, 2002.
- [104] N. Cristianini, J. Kandola, A. Elisseeff, and J. Shawe-Taylor, “On kernel-target alignment,” in *Innovations in Machine Learning: Theory and Applications*, D. Holmes and L. Jain, Eds. New York: Springer-Verlag, 2006.
- [105] M. Crovella and B. Krishnamurthy, *Internet Measurement*. New York: Wiley, 2006.
- [106] D. Daley and J. Gani, *Epidemic Modeling*. New York: Cambridge University Press, 1999.
- [107] L. Dall’Asta, I. Alvarez-Hamelin, A. Barrat, A. Vázquez, and A. Vespignani, “Exploring networks with traceroute-like probes: theory and simulations,” *Theoretical Computer Science*, vol. 355, no. 1, pp. 6–24, 2006.
- [108] J. N. Darroch and D. Ratcliff, “A note on capture-recapture estimation,” *Biometrics*, vol. 36, pp. 149–153, 1980.
- [109] G. Davidson, B. Wylie, and K. Boyack, “Cluster stability and the use of noise in interpretation of clustering,” *IEEE Symposium on Information Visualization*, pp. 23–30, 2002.
- [110] 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, no. 18, pp. 3565–3574, 2004.
- [111] W. de Nooy, A. Mrvar, and V. Batagelj, *Exploratory Social Network Analysis with Pajek*. New York: Cambridge University Press, 2005.
- [112] D. de Solla Price, “Networks of scientific papers,” *Science*, vol. 149, no. 3683, pp. 510–515, 1965.
- [113] W. Deming and F. Stephan, “On a least squares adjustment of a sampled frequency table when the expected marginal totals are known,” *Annals of Mathematical Statistics*, vol. 11, no. 4, pp. 427–444, 1940.
- [114] N. Demiris and P. O’Neill, “Bayesian inference for stochastic multitype epidemics in structured populations via random graphs,” *Journal of the Royal Statistical Society Series B*, vol. 67, no. 5, pp. 731–745, 2005.
- [115] A. Dempster, “Covariance selection,” *Biometrics*, vol. 28, no. 1, pp. 157–175, 1972.
- [116] M. Deng, T. Chen, and F. Sun, “An integrated analysis of protein function prediction,” *Journal of Computational Biology*, vol. 11, pp. 463–475, 2004.

- [117] G. di Battista, P. Eades, R. Tamassia, and I. Tollis, *Graph Drawing*. Englewood Cliffs, NJ: Prentice Hall, 1999.
- [118] R. Diestel, *Graph Theory, Third Edition*. Heidelberg: Springer-Verlag, 2005.
- [119] A. Dobra, C. Hans, B. Jones, J. Nevin, G. Yao, and M. West, “Sparse graphical models for exploring gene expression data,” *Journal of Multivariate Analysis*, vol. 90, no. 1, pp. 196–212, 2004.
- [120] D. Donoho and C. Grimes, “Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data,” *Proceedings of the National Academy of Sciences*, vol. 100, no. 10, pp. 5591–5596, 2003.
- [121] P. Doreian and F. Stokman, Eds., *Evolution of Social Networks*. New York: Gordon and Breach, 1997.
- [122] P. Doreian, V. Batagelj, and A. Ferligoj, *Generalized Blockmodeling*. New York: Cambridge University Press, 2004.
- [123] S. Dorogovtsev and J. Mendes, *Evolution of Networks: From Biological Nets to the Internet and WWW*. Oxford: Oxford University Press, 2003.
- [124] S. Dorogovtsev, J. Mendes, and A. Samukhin, “Structure of growing networks with preferential linking,” *Physical Review Letters*, vol. 85, no. 21, pp. 4633–4636, 2000.
- [125] J. Doyle, D. Alderson, L. Li, S. Low, M. Roughan, S. Shalunov, R. Tanaka, and W. Willinger, “The ‘robust yet fragile’ nature of the Internet,” *Proceedings of the National Academy of Sciences*, vol. 102, no. 41, pp. 14 497–14 502, 2005.
- [126] H. Drees, L. de Haan, and S. Resnick, “How to make a Hill plot,” *The Annals of Statistics*, vol. 28, no. 1, pp. 254–274, 2000.
- [127] M. Drton and M. Perlman, “Model selection for Gaussian concentration graphs,” *Biometrika*, vol. 91, no. 3, pp. 591–602, 2004.
- [128] N. Duffield, J. Horowitz, F. Lo Presti, and D. Towsley, “Multicast topology inference from measured end-to-end loss,” *IEEE Transactions on Information Theory*, vol. 48, no. 1, pp. 26–45, 2002.
- [129] D. Edwards, *Introduction to Graphical Modelling*. New York: Springer, 2000.
- [130] B. Efron, “Correlation and large-scale simultaneous significance testing,” *Journal of the American Statistical Association*, vol. 102, no. 477, p. 93, 2007.
- [131] B. Efron and R. Tibshirani, *An Introduction to the Bootstrap*. London: Chapman & Hall, 1993.
- [132] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani, “Least angle regression,” *Annals of Statistics*, vol. 32, pp. 407–451, 2004.
- [133] D. Eppstein and J. Wang, “Fast approximation of centrality,” *Journal of Graph Algorithms and Applications*, vol. 8, no. 1, pp. 39–45, 2004.
- [134] P. Erdős and A. Rényi, “On random graphs,” *Publ. Math. Debrecen*, vol. 6, no. 290, pp. 290–297, 1959.
- [135] ———, “On the evolution of random graphs,” *Publ. Math. Inst. Hung. Acad. Sci.*, vol. 5, pp. 17–61, 1960.

- [136] ——, “On the strength of connectedness of a random graph,” *Acta. Math. Acad. Sci. Hungar.*, vol. 12, pp. 261–267, 1961.
- [137] V. Erramilli, M. Crovella, and N. Taft, “An independent-connection model for traffic matrices,” *Proceedings of the 6th ACM SIGCOMM Conference on Internet Measurement*, pp. 251–256, 2006.
- [138] W. Esty, “A normal limit law for a nonparametric estimator of the coverage of a random sample,” *The Annals of Statistics*, vol. 11, no. 3, pp. 905–912, 1983.
- [139] M. Everett and S. Borgatti, “The centrality of groups and classes,” *Journal of Mathematical Sociology*, vol. 23, pp. 181–202, 1999.
- [140] J. Faith, B. Hayete, J. Thaden, I. Mogno, J. Wierzbowski, G. Cottarel, S. Kasif, J. Collins, and T. Gardner, “Large-scale mapping and validation of *Escherichia coli* transcriptional regulation from a compendium of expression profiles,” *PLoS Biology*, vol. 5, no. 1, p. e8, 2007.
- [141] J. Felsenstein, *Inferring Phylogenies*. Sunderland, MA: Sinear Associates, 2004.
- [142] ——, “Maximum likelihood and minimum-step methods of phylogenetic reconstruction from discrete datfor estimating evolutionary trees from data on discrete characters,” *Systematic Zoology*, vol. 22, pp. 240–249, 1973.
- [143] ——, “Evolutionary trees from DNA sequences: a maximum likelihood approach,” *Journal of Molecular Evolution*, vol. 17, pp. 368–376, 1981.
- [144] M. Fiedler, “Algebraic connectivity of graphs,” *Czechoslovak Mathematical Journal*, vol. 23, no. 98, pp. 298–305, 1973.
- [145] M. Fischer and S. Gopal, “Artificial neural networks: a new approach to modeling interregional telecommunication flows,” *Journal of Regional Science*, vol. 34, no. 4, pp. 503–527, 1994.
- [146] L. Ford and D. Fulkerson, *Flows in Networks*. Princeton, NJ: Princeton University Press, 1962.
- [147] A. Fotheringham and M. O’Kelly, *Spatial Interaction Models: Formulations and Applications*. The Netherlands: Kluwer Academic Publishers, 1989.
- [148] O. Frank, “Survey sampling in graphs,” *Journal of Statistical Planning and Inference*, vol. 1, pp. 235–264, 1977.
- [149] ——, “Estimation of graph totals,” *Scandinavian Journal of Statistics*, vol. 4, pp. 81–89, 1977.
- [150] ——, “Estimation of the number of connected components in a graph by using a sampled subgraph,” *Scandanavian Journal of Statistics*, vol. 5, pp. 177–188, 1978.
- [151] ——, “Sampling and estimation in large social networks,” *Social Networks*, vol. 1, no. 1, pp. 91–101, 1978.
- [152] ——, “Network sampling and model fitting,” in *Models and Methods in Social Network Analysis*, P. Carrington, J. Scott, and S. Wasserman, Eds. New York: Cambridge University Press, 2004.
- [153] ——, “Statistical inference in graphs,” Ph.D. dissertation, Stockholm University, 1971.

- [154] O. Frank and T. Snijders, “Estimating the size of hidden populations using snowball sampling,” *Journal of Official Statistics*, vol. 10, pp. 53–67, 1994.
- [155] O. Frank and D. Strauss, “Markov graphs,” *Journal of the American Statistical Association*, vol. 81, no. 395, pp. 832–842, 1986.
- [156] L. Freeman, “A set of measures of centrality based on betweenness,” *Sociometry*, vol. 40, no. 1, pp. 35–41, 1977.
- [157] ———, “Centrality in social networks: Conceptual clarification,” *Social Networks*, vol. 1, no. 3, pp. 215–239, 1979.
- [158] ———, “Visualizing social networks,” *Journal of Social Structure*, vol. 1, no. 1, p. 4, 2000.
- [159] L. Freeman, S. Borgatti, and D. White, “Centrality in valued graphs: A measure of betweenness based on network flow,” *Social Networks*, vol. 13, no. 2, pp. 141–154, 1991.
- [160] J. Friedman, “On bias, variance, 0/1-loss, and the curse-of-dimensionality,” *Data Mining and Knowledge Discovery*, vol. 1, pp. 55–77, 1997.
- [161] J. Friedman, T. Hastie, and R. Tibshirani, “Sparse inverse covariance estimation with the graphical lasso,” *Biostatistics*, vol. 9, no. 3, p. 432, 2008.
- [162] N. Friedman, M. Linial, I. Nachman, and D. Pe’er, “Using Bayesian networks to analyze expression data,” *Journal of Computational Biology*, vol. 7, no. 3–4, pp. 601–620, 2000.
- [163] A. Fronczak, J. Holyst, M. Jedynak, and J. Sienkiewicz, “Higher order clustering coefficients in Barabási–Albert networks,” *Physica A*, vol. 316, no. 1–4, pp. 688–694, 2002.
- [164] T. Fruchterman and E. Reingold, “Graph drawing by force-directed placement,” *Software – Practice and Experience*, vol. 21, no. 11, pp. 1129–1164, 1991.
- [165] E. Gansner, Y. Koren, and S. North, “Graph drawing by stress majorization,” in *Proceedings of the 12th International Symposium on Graph Drawing*, ser. Lecture Notes in Computer Science, J. Pach, Ed., vol. 3383. Berlin: Springer-Verlag, 2005, pp. 239–250.
- [166] T. Gardner and J. Faith, “Reverse-engineering transcription control networks,” *Physics of Life Reviews*, vol. 2, no. 1, pp. 65–88, 2005.
- [167] S. Geman and D. Geman, “Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 6, no. 6, pp. 721–741, 1984.
- [168] L. Getoor and B. Taskar, Eds., *Introduction to Statistical Relational Learning*. Cambridge, MA: MIT Press, 2007.
- [169] L. Getoor, N. Friedman, D. Koller, and B. Taskar, “Learning probabilistic models of relational structure,” *Journal of Machine Learning Research*, vol. 3, pp. 679–707, 2002.
- [170] C. Geyer and E. Thompson, “Constrained Monte Carlo maximum likelihood for dependent data,” *Journal of the Royal Statistical Society, Series B*, vol. 54, no. 3, pp. 657–699, 1992.
- [171] Z. Ghahramani and G. Hinton, “Parameter estimation for linear dynamical systems,” University of Toronto, Tech. Rep. CRG-TR-96-2, 1996.

- [172] D. Gibson, J. Kleinberg, and P. Raghavan, “Inferring Web communities from link topology,” *Hypertext 98: The Ninth ACM Conference on Hypertext and Hypermedia, June 20–24, 1998, Pittsburgh*, 1998.
- [173] E. Gilbert, “Random graphs,” *Annals of Mathematical Statistics*, vol. 30, no. 4, pp. 1141–1144, 1959.
- [174] M. Girvan and M. Newman, “Community structure in social and biological networks,” *Proceedings of the National Academy of Sciences*, vol. 99, no. 12, pp. 7821–7826, 2002.
- [175] C. Gkantsidis, M. Mihail, and E. Zegura, “Spectral analysis of Internet topologies,” in *Proceedings of the 22nd Annual INFOCOM Conference*, 2003, pp. 364–374.
- [176] C. Godsil and G. Royle, *Algebraic Graph Theory*. New York: Springer, 2001.
- [177] A. Goldberg and R. Tarjan, “A new approach to the maximum-flow problem,” *Journal of the ACM*, vol. 35, no. 4, pp. 921–940, 1988.
- [178] D. Goldberg and F. Roth, “Assessing experimentally derived interactions in a small world,” *Proceedings of the National Academy of Sciences*, vol. 100, no. 8, pp. 4372–4376, 2003.
- [179] M. Goldstein, S. Morris, and G. Yen, “Problems with fitting to the power-law distribution,” *European Physical Journal B*, vol. 41, pp. 255–258, 2004.
- [180] A. Golightly and D. Wilkinson, “Bayesian inference for stochastic kinetic models using a diffusion approximation,” *Biometrics*, vol. 61, no. 3, pp. 781–788, 2005.
- [181] G. Golub and C. van Loan, *Matrix Computations, Third Edition*. Baltimore: Johns Hopkins University Press, 1996.
- [182] R. Gomory and T. Hu, “Multi-terminal network flows,” *Journal of the Society for Industrial and Applied Mathematics*, vol. 9, no. 4, pp. 551–570, 1961.
- [183] I. Good, “The population frequencies of species and the estimation of population parameters,” *Biometrika*, vol. 40, pp. 237–264, 1953.
- [184] S. Gopal, “The evolving social geography of blogs,” in *Societies and Cities in the Age of Instant Access*, H. Miller, Ed. Berlin: Springer, 2007, pp. 275–294.
- [185] A. Gordon, “A review of hierarchical classification,” *Journal of the Royal Statistical Society A*, vol. 150, no. 2, pp. 119–137, 1987.
- [186] J. Gross and J. Yellen, *Graph Theory And Its Applications*. Boca Raton, FL: Chapman & Hall/CRC, 1999.
- [187] C. Gu, *Smoothing Spline Anova Models*. New York: Springer-Verlag, 2002.
- [188] J. Guare, *Six Degrees of Separation: A Play*. New York: Vintage, 1990.
- [189] S. Gupta, R. Anderson, and R. May, “Networks of sexual contacts: implications for the pattern of spread of HIV,” *AIDS*, vol. 3, no. 12, pp. 807–817, 1989.
- [190] P. Hage and H. Harary, *Structural Models in Anthropology*. New York: Cambridge University Press, 1983.
- [191] J. Hammersley and P. Clifford, “Markov fields on finite graphs and lattices,” 1971, unpublished manuscript.

- [192] J. Han, D. Dupuy, N. Bertin, M. Cusick, and M. Vidal, “Effect of sampling on topology predictions of protein-protein interaction networks,” *Nature Biotechnology*, vol. 23, pp. 839–844, 2005.
- [193] M. Handcock, “Assessing degeneracy in statistical models of social networks,” Center for Statistics and the Social Sciences, University of Washington, Tech. Rep. No. 39, 2003.
- [194] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*. New York: Springer, 2001.
- [195] D. Heckerman, “A tutorial on learning with Bayesian networks,” in *Learning in Graphical Models*, M. Jordan, Ed. Cambridge, MA: MIT Press, 1999, pp. 301–354.
- [196] J. Heesterbeek, “A brief history of R_0 and a recipe for its calculation,” *Acta Biotheretica*, vol. 50, pp. 189–204, 2002.
- [197] F. Hernández-Campos, J. Marron, G. Samorodnitsky, and F. Smith, “Variable heavy tails in Internet traffic,” *Performance Evaluation*, vol. 58, no. 2-3, pp. 261–284, 2004.
- [198] S. Hill, D. Agarwal, R. Bell, and C. Volinsky, “Building an effective representation for dynamic networks,” *Journal of Computational and Graphical Statistics*, vol. 15, no. 3, pp. 584–608, 2006.
- [199] P. Hoff, “Bilinear mixed-effect models for dyadic data,” *Journal of the American Statistical Association*, vol. 100, pp. 286–295, 2005.
- [200] ———, “Multiplicative latent factor models for description and prediction of social networks,” *Computational and Mathematical Organization Theory*, 2007.
- [201] ———, “Modeling homophily and stochastic equivalence in symmetric relational data,” *Advances in Neural Information Processing Systems, NIPS*, 2008.
- [202] P. Hoff, A. Raftery, and M. Handcock, “Latent space approaches to social network analysis,” *Journal of the American Statistical Association*, vol. 97, no. 460, pp. 1090–1098, 2002.
- [203] P. Holland and S. Leinhardt, “An exponential family of probability distributions for directed graphs (C/R: P51-65),” *Journal of the American Statistical Association*, vol. 76, pp. 33–50, 1981.
- [204] S. Holmes, “Phylogenies: An overview,” in *Statistics and Genetics*, ser. IMA, E. Halloran and S. Geisser, Eds. New York: Springer-Verlag, 1999, vol. 81.
- [205] ———, “Statistics for phylogenetic trees,” *Theoretical Population Biology*, vol. 63, pp. 17–32, 2003.
- [206] H. Hotelling, “New light on the correlation coefficient and its transforms,” *Journal of the Royal Statistical Society, Series B*, vol. 15, no. 2, pp. 193–232, 1953.
- [207] J. Huelsenbeck, F. Ronquist, R. Nielsen, and J. Bollback, “Bayesian inference of phylogeny and its impact on evolutionary biology,” *Science*, vol. 294, no. 5550, pp. 2310–2314, 2001.
- [208] D. Hunter, “Curved exponential family models for social networks,” *Social Networks*, vol. 29, no. 2, pp. 216–230, 2007.

- [209] D. Hunter and M. Handcock, “Inference in curved exponential family models for networks,” *Journal of Computational and Graphical Statistics*, vol. 15, no. 3, pp. 565–583, 2006.
- [210] D. Huson and D. Bryant, “Application of phylogenetic networks in evolutionary studies,” *Molecular Biology and Evolution*, vol. 23, no. 2, pp. 254–267, 2006.
- [211] E. Ising, “Beitrag zur theorie des ferromagnetismus,” *Zeit. fur Physik*, vol. 31, pp. 253–258, 1925.
- [212] M. Jackson, “A survey of models of network formation: stability and efficiency,” in *Group Formation in Economics: Networks, Clubs, and Coalitions*, G. Demange and M. Wooders, Eds. New York: Cambridge University Press, 2005.
- [213] A. Jain, M. Murty, and P. Flynn, “Data clustering: a review,” *ACM Computing Surveys*, vol. 31, no. 3, pp. 264–323, 1999.
- [214] R. Johnson and D. Wichern, *Applied Multivariate Statistical Analysis, Fifth Edition*. Upper Saddle River, NJ: Pearson Education, 2001.
- [215] M. Jordan, Ed., *Learning in Graphical Models*. Cambridge, MA: MIT Press, 1999.
- [216] M. Jordan, “Graphical models,” *Statistical Science*, vol. 19, no. 1, pp. 140–155, 2004.
- [217] T. Kamada and S. Kawai, “An algorithm for drawing general undirected graphs,” *Information Processing Letters*, vol. 31, no. 1, pp. 7–15, 1989.
- [218] N. Kashtan, S. Itzkovitz, R. Milo, and U. Alon, “Efficient sampling algorithm for estimating subgraph concentrations and detecting network motifs,” *Bioinformatics*, vol. 20, no. 11, pp. 1746–1758, 2004.
- [219] R. Kasturirangan, “Multiple scales in small-world graphs,” Massachusetts Institute of Technology, Tech. Rep. AIM-1663, 1999.
- [220] L. Katz, “A new status index derived from sociometric analysis,” *Psychometrika*, vol. 18, no. 1, pp. 39–43, 1953.
- [221] M. Kaufmann and D. Wagner, Eds., *Drawing Graphs*. Berlin: Springer, 1998.
- [222] M. Keeling, “The implications of network structure for epidemic dynamics,” *Theoretical Population Biology*, vol. 67, no. 1, pp. 1–8, 2005.
- [223] M. Keeling and K. Eames, “Networks and epidemic models,” *Journal of the Royal Society Interface*, vol. 2, pp. 295–307, 2005.
- [224] D. Kempe, J. Kleinberg, and A. Kumar, “Connectivity and inference problems for temporal networks,” *Journal of Computer and System Sciences*, vol. 64, no. 4, pp. 820–842, 2002.
- [225] W. Kermack and A. McKendrick, “A contribution to the mathematical theory of epidemics,” *Proceedings of the Royal Society of London. Series A*, vol. 115, no. 772, pp. 700–721, 1927.
- [226] J. Kim, P. Krapivsky, B. Kahng, and S. Redner, “Infinite-order percolation and giant fluctuations in a protein interaction network,” *Physical Review E*, vol. 66, no. 5, p. 55101, 2002.

- [227] G. Kimeldorf and G. Wahba, "Some results on Tchebycheffian spline functions," *Journal of Mathematical Analysis and Applications*, vol. 33, pp. 82–95, 1971.
- [228] R. Klavans and K. Boyack, "Quantitative evaluation of large maps of science," *Scientometrics*, vol. 68, no. 3, pp. 475–499, 2006.
- [229] J. Kleinberg, "Navigation in a small world," *Nature*, vol. 406, no. 6798, p. 845, 2000.
- [230] ———, "Authoritative sources in a hyperlinked environment," *Journal of the ACM*, vol. 46, no. 5, pp. 604–632, 1999.
- [231] ———, "The small-world phenomena: an algorithmic perspective," in *Proceedings of the 32nd Annual ACM Symposium on Theory of Computing*. New York: Association of Computing Machinery, 2000, pp. 163–170.
- [232] J. Kleinberg, S. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins, "The web as a graph: measurements, models, and methods," in *Proceedings of the International Conference on Combinatorics and Computing*, ser. Lecture Notes in Computer Science, vol. 1627. Berlin: Springer, 1999, pp. 1–18.
- [233] E. Kolaczyk, D. Chua, and M. Barthélémy, "Group betweenness and co-betweenness: inter-related notions of coalition centrality," *Social Networks*, 2009, (to appear).
- [234] R. Kondor and J. Lafferty, "Diffusion kernels on graphs and other discrete input spaces," *Proceedings of the ICML*, 2002.
- [235] D. König, *Theorie der Endlichen und Unendlichen Graphen*. American Mathematical Society, 1950.
- [236] J. Koskinen and T. Snijders, "Bayesian inference for dynamic social network data," *Journal of Statistical Planning and Inference*, vol. 137, pp. 3930–3938, 2007.
- [237] G. Kossinets, "Effects of missing data in social networks," *Social Networks*, vol. 28, no. 3, pp. 247–268, 2006.
- [238] G. Kossinets and D. Watts, "Empirical analysis of an evolving social network," *Science*, vol. 311, no. 5757, pp. 88–90, 2006.
- [239] M. Kramer, E. Kolaczyk, and H. Kirsch, "Emergent network topology at seizure onset in humans," *Epilepsy Research*, vol. 79, pp. 173–186, 2008.
- [240] P. Krapivsky, S. Redner, and F. Leyvraz, "Connectivity of growing random networks," *Physical Review Letters*, vol. 85, no. 21, pp. 4629–4632, 2000.
- [241] R. Kumar, P. Raghavan, S. Rajagopalan, D. Sivakumar, A. Tomkins, and E. Upfal, "Stochastic models for the web graph," *Proceedings of the 42nd Annual IEEE Symposium on the Foundations of Computer Science*, vol. 41, pp. 57–65, 2000.
- [242] R. Kumar, P. Raghavan, J. Novak, and A. Tomkins, "On the bursty evolution of Blogspace," in *Proceedings of the 14th International World Wide Web Conference*, 2003.
- [243] A. Lakhina, J. Byers, M. Crovella, and P. Xie, "Sampling biases in IP topology measurements," *Proceedings of the 22nd Annual INFOCOM 2003*, 2003.
- [244] A. Lakhina, M. Crovella, and C. Diot, "Diagnosing network-wide traffic anomalies," in *Proceedings of the ACM SIGCOMM'04*, 2004, pp. 219–230.

- [245] A. Lakhina, K. Papagiannaki, C. Crovella, M. Diot, E. Kolaczyk, and N. Taft, “Structural analysis of network traffic flows,” in *Proceedings of the ACM Sigmetrics Conference*, 2004.
- [246] G. Lanckriet, N. Cristianini, P. Bartlett, L. El Ghaoui, and M. Jordan, “Learning the kernel matrix with semidefinite programming,” *Journal of Machine Learning Research*, vol. 5, pp. 27–72, 2004.
- [247] G. Lanckriet, N. Cristianini, M. Jordan, and W. Nobel, “A statistical framework for genomic data fusion,” *Bioinformatics*, vol. 20, pp. 2626–2635, 2004.
- [248] S. Lauritzen, *Graphical Models*. Oxford: Oxford University Press, 1996.
- [249] S. Lauritzen and D. Spiegelhalter, “Local computations with probabilities on graphical structures and their application to expert systems,” *Journal of the Royal Statistical Society*, vol. 50, no. 2, pp. 157–224, 1988.
- [250] E. Lawrence, G. Michailidis, and V. Nair, “Network delay tomography using flexicast experiments,” *Journal of the Royal Statistical Society: Series B*, vol. 68, no. 5, pp. 785–813, 2006.
- [251] E. Lazega, *The Collegial Phenomenon: The Social Mechanisms of Cooperation Among Peers in a Corporate Law Partnership*. Oxford: Oxford University Press, 2001.
- [252] E. Lazega and P. Pattison, “Multiplexity, generalized exchange and cooperation in organizations: a case study,” *Social Networks*, vol. 21, no. 1, pp. 67–90, 1999.
- [253] S. Lee, P. Kim, and H. Jeong, “Statistical properties of sampled networks,” *Physical Review E*, vol. 73, no. 1, p. 16102, 2006.
- [254] W. Leland, M. Taqqu, W. Willinger, D. Wilson, and M. Bellcore, “On the self-similar nature of ethernet traffic (extended version),” *IEEE/ACM Transactions on Networking*, vol. 2, no. 1, pp. 1–15, 1994.
- [255] J. Leskovec, J. Kleinberg, and C. Faloutsos, “Graphs over time: densification laws, shrinking diameters, and possible explanations,” in *Proceedings of the 11th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2005.
- [256] L. Li, D. Alderson, W. Willinger, and J. Doyle, “A first-principles approach to understanding the Internet’s router-level topology,” *ACM SIGCOMM Computer Communication Review*, vol. 34, no. 4, pp. 3–14, 2004.
- [257] M. Li and L. Zhang, “Twist-rotation transformations of binary trees and arithmetic expressions,” *Journal of Algorithms*, vol. 32, no. 2, pp. 155–166, 1999.
- [258] S. Li, *Markov Random Field Modeling in Computer Vision*. New York: Springer-Verlag, 1995.
- [259] F. Liang, K. Mao, M. Liao, S. Mukherjee, and M. West, “Non-parametric Bayesian kernel models,” Department of Statistical Science, Duke University, Tech. Rep., 2007.
- [260] G. Liang and B. Yu, “Maximum pseudo likelihood estimation in network tomography,” *IEEE Transactions on Signal Processing*, vol. 51, no. 8, pp. 2043–2053, 2003.

- [261] D. Liben-Nowell and J. Kleinberg, “The link prediction problem for social networks,” in *Proceedings of the 12th International Conference on Information and Knowledge Management*, 2003.
- [262] Y. Lin and H. Zhang, “Component selection and smoothing in multivariate nonparametric regression,” *Annals of Statistics*, vol. 34, no. 5, pp. 2272–2297, 2006.
- [263] R. Little and D. Rubin, *Statistical Analysis with Missing Data, Second Edition*. New York: Wiley & Sons, Inc., 2002.
- [264] J. Liu, *Monte Carlo Strategies in Scientific Computing*. New York: Springer, 2002.
- [265] F. Lo Presti, N. Duffield, J. Horowitz, and D. Towsley, “Multicast-based inference of network-internal delay distributions,” *IEEE/ACM Transactions on Networking*, vol. 10, no. 6, pp. 761–775, 2002.
- [266] I. MacDonald and W. Zucchini, *Hidden Markov Models and Other Models for Discrete-valued Time Series*. London: Chapman & Hall, 1997.
- [267] M. Maher, “Inferences on trip matrices from observations on link volumes: a Bayesian statistical approach,” *Transportation Research, Part B*, vol. 17, no. 6, pp. 435–447, 1983.
- [268] T. Margush and F. McMorris, “Consensus n -trees,” *Bulletin of Mathematical Biology*, vol. 43, pp. 239–244, 1981.
- [269] P. Marsden, “Recent developments in network measurement,” in *Models and Methods in Social Network Analysis*, P. Carrington, J. Scott, and S. Wasserman, Eds. Cambridge University Press, 2004.
- [270] S. Maslov and K. Sneppen, “Specificity and stability in topology of protein networks,” *Science*, vol. 296, no. 5569, pp. 910–913, 2002.
- [271] S. Maslov, K. Sneppen, and A. Zaliznyak, “Pattern detection in complex networks: correlation profile of the Internet,” *arxiv preprint cond-mat/0205379*, 2002.
- [272] P. McCullagh and J. Nelder, *Generalized Linear Models*. London: Chapman & Hall/CRC, 1989.
- [273] J. McDonald, P. Smith, and J. Forster, “Markov chain Monte Carlo exact inference of social networks,” *Social Networks*, vol. 29, pp. 127–136, 2007.
- [274] B. McKay and N. Wormald, “The degree sequence of a random graph. I. The models,” *Random Structures and Algorithms*, vol. 11, pp. 97–117, 1997.
- [275] N. Meinshausen and P. Bühlmann, “High-dimensional graphs and variable selection with the Lasso,” *Annals of Statistics*, vol. 34, no. 3, pp. 1436–1462, 2006.
- [276] M. Mihail, “Conductance and convergence of Markov chains – a combinatorial treatment of expanders.” in *Proceedings of the 30th Annual Symposium of the Foundations of Computer Science (FOCS)*, 1989, pp. 526–531.
- [277] S. Milgram, “The small world problem,” *Psychology Today*, vol. 2, no. 1, pp. 60–67, 1967.
- [278] R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, and U. Alon, “Network motifs: simple building blocks of complex networks,” *Science*, vol. 298, no. 5594, pp. 824–827, 2002.

- [279] R. Milo, N. Kashtan, S. Itzkovitz, M. Newman, and U. Alon, “On the uniform generation of random graphs with prescribed degree sequences,” *eprint arXiv: cond-mat/0312028v1*, 2003.
- [280] M. Mitzenmacher, “A brief history of generative models for power law and lognormal distributions,” *Internet Mathematics*, vol. 1, no. 2, pp. 226–251, 2004.
- [281] B. Mohar, “The Laplacian spectrum of graphs,” in *Graph Theory, Combinatorics, and Applications*, Y. Alavi, G. Chartrand, O. Oellermann, and A. Schwenk, Eds. New York: Wiley, 1991, vol. 2, pp. 871–898.
- [282] M. Molloy and B. Reed, “A critical point for random graphs with a given degree sequence,” *Random Structures and Algorithms*, vol. 6, no. 2/3, pp. 161–180, 1995.
- [283] ——, “The size of the giant component of a random graph with a given degree sequence,” *Combinatorics, Probability and Computing*, vol. 7, no. 3, pp. 295–305, 1998.
- [284] R. Monasson, “Diffusion, localization, and dispersion relations on ‘small-world’ lattices,” *European Physical Journal B*, vol. 12, pp. 555–567, 1999.
- [285] J. Moody, D. McFarland, and S. Bender-deMoll, “Dynamic network visualization,” *American Journal of Sociology*, vol. 110, no. 4, pp. 1206–1241, 2005.
- [286] T. Murali, C. Wu, and S. Kasif, “The art of gene function prediction,” *Nature Biotechnology*, vol. 24, no. 12, pp. 1474–1475, 2006.
- [287] N. Nariai, E. Kolaczyk, and S. Kasif, “Probabilistic protein function prediction from heterogeneous genome-wide data,” *PLoS ONE*, vol. 2, no. 3, p. e337, 2007.
- [288] M. Newman, “Assortative mixing in networks,” *Physical Review Letters*, vol. 89, no. 20, p. 208701, 2002.
- [289] ——, “Detecting community structure in networks,” *European Physical Journal B*, vol. 38, pp. 321–330, 2004.
- [290] ——, “Ego-centered networks and the ripple effect,” *Social Networks*, vol. 25, no. 1, pp. 83–95, 2003.
- [291] ——, “Finding community structure in networks using the eigenvectors of matrices,” *Physical Review E*, vol. 74, no. 3, p. 36104, 2006.
- [292] ——, “Spread of epidemic disease on networks,” *Physical Review E*, vol. 66, no. 1, p. 16128, 2002.
- [293] ——, “Mixing patterns in networks,” *Physical Review E*, vol. 67, no. 2, p. 26126, 2003.
- [294] ——, “Fast algorithm for detecting community structure in networks,” *Physical Review E*, vol. 69, p. 066133, 2004.
- [295] ——, “A measure of betweenness centrality based on random walks,” *Social Networks*, vol. 27, no. 1, pp. 39–54, 2005.
- [296] ——, “The structure and function of complex networks,” *SIAM Review*, vol. 45, pp. 167–256, 2003.
- [297] M. Newman and M. Girvan, “Finding and evaluating community structure in networks,” *Physical Review E*, vol. 69, no. 2, p. 26113, 2004.

- [298] M. Newman and D. Watts, “Renormalization group analysis of the small-world network model,” *Physics Letters A*, vol. 263, no. 4–6, pp. 341–346, 1999.
- [299] M. Newman, S. Strogatz, and D. Watts, “Random graphs with arbitrary degree distributions and their applications,” *Physical Review E*, vol. 64, no. 2, p. 26118, 2001.
- [300] H. Nguyen and P. Thiran, “Active measurement for multiple link failures: Diagnosis in IP networks,” in *Proceedings of the 5th International Passive and Active Measurement Workshop*, ser. Lecture Notes in Computer Science, C. Barakat and I. Pratt, Eds., vol. 3015. Berlin: Springer-Verlag, 2004, pp. 185–194.
- [301] ——, “Binary versus analogue path monitoring in IP networks,” in *Passive and Active Network Measurement*, ser. Lecture Notes in Computer Science, vol. 3431. Berlin / Heidelberg: Springer, 2005, pp. 97–107.
- [302] J. Ni, H. Xie, S. Tatikonda, and Y. Yang, “Network routing topology inference from end-to-end measurements,” in *Proceedings of the 27th IEEE Infocom*, 2008, pp. 36–40.
- [303] N. Nihan and G. Davis, “Recursive estimation of origin-destination matrices from input/output counts,” *Transportation Research Part B*, vol. 21B, pp. 149–163, 1987.
- [304] K. Nowicki and T. Snijders, “Estimation and prediction for stochastic block-structures,” *Journal of the American Statistical Association*, vol. 96, no. 455, pp. 1077–1087, 2001.
- [305] S. Ohno, *Evolution by Gene Duplication*. New York: Springer Verlag, 1970.
- [306] J. Ortúzar and L. Willumsen, *Modeling Transport, Third Edition*. West Sussex, England: John Wiley & Sons, 2001.
- [307] L. Page, S. Brin, R. Motwani, and T. Winograd, “The pagerank citation ranking: bringing order to the web,” Technical Report, Stanford Digital Library Technologies Project, 1998.
- [308] K. Park and W. Willinger, Eds., *Self-similar Network Traffic and Performance Evaluation*. New York: John Wiley & Sons, 2000.
- [309] R. Pastor-Satorras and A. Vespignani, *Evolution and Structure of the Internet: A Statistical Physics Approach*. New York: Cambridge University Press, 2004.
- [310] R. Pastor-Satorras, A. Vázquez, and A. Vespignani, “Dynamical and correlation properties of the internet,” *Physical Review Letters*, vol. 87, no. 25, p. 258701, 2001.
- [311] P. Pattison and G. Robins, “Neighborhood-based models for social networks,” *Sociological Methodology*, vol. 32, no. 1, pp. 301–337, 2002.
- [312] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. San Francisco: Morgan Kaufmann Publishers, 1988.
- [313] D. Phillips and A. Garcia-Diaz, *Fundamentals of Network Analysis*. Englewood Cliffs, NJ: Prentice-Hall, Inc., 1981.

- [314] A. Popescul and L. Ungar, "Statistical relational learning for link prediction," in *Proceedings of the Workshop on Learning Statistical Models from Relational Data at IJCAI-2003*, 2003.
- [315] C. Priebe, J. Conroy, D. Marchette, and Y. Park, "Scan statistics on Enron graphs," in *Proceedings of the SIAM International Conference on Data Mining, Workshop on Link Analysis, Counterterrorism, and Security*, 2005.
- [316] R. Puzi, Y. Elovici, and S. Dolev, "Fast algorithm for successive computation of group betweenness centrality," *Physical Review E*, vol. 76, p. 56709, 2007.
- [317] S. Qazi and T. Moors, "Practical issues of statistical path monitoring in overlay networks with large, rank-deficient routing matrices," 2008, (submitted).
- [318] M. Rabbat, M. Figueiredo, and R. Nowak, "Genomic network tomography," in *Proceedings of the 32nd IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2007.
- [319] ———, "Network inference from co-occurrences," *IEEE Transactions on Information Theory*, 2008, (in press).
- [320] M. Rabbat, M. Coates, and R. Nowak, "Multiple source Internet tomography," *IEEE Journal on Selected Areas in Communication*, vol. 24, no. 12, pp. 2221–2234, 2006.
- [321] J. Ramsay, G. Hooker, D. Campbell, and J. Cao, "Parameter estimation for differential equations: a generalized smoothing approach," *Journal of the Royal Statistical Society, Series B*, vol. 69, no. 5, pp. 741–796, 2007.
- [322] A. Rao, R. Jana, and S. Bandyopadhyay, "A Markov chain Monte Carlo method for generating random $(0, 1)$ -matrices with given marginals," *Sankhya, Series A*, vol. 58, pp. 225–242, 1996.
- [323] S. Ratnasamy and S. McCanne, "Inference of multicast routing trees and bottleneck bandwidths using end-to-end measurements," in *Proceedings of the IEEE Infocom '99*, New York, 1999.
- [324] H. Robbins and S. Monro, "A stochastic approximation method," *Annals of Mathematical Statistics*, vol. 22, pp. 400–407, 1951.
- [325] C. Robert and G. Casella, *Monte Carlo Statistical Methods, Second Edition*. New York: Springer, 2005.
- [326] J. Roberts, "Simple methods for simulating sociomatrices with given marginal totals," *Social Networks*, vol. 22, no. 3, pp. 273–283, 2000.
- [327] P. Robillard, "Estimating the O-D matrix from observed link volumes," *Transportation Research*, vol. 9, pp. 123–128, 1975.
- [328] G. Robins and M. Morris, "Advances in exponential random graph (p^*) models," *Social Networks*, vol. 29, no. 2, pp. 169–172, 2007.
- [329] G. Robins, T. Snijders, P. Wang, M. Handcock, and P. Pattison, "Recent developments in exponential random graph (p^*) models for social networks," *Social Networks*, vol. 29, no. 2, pp. 192–215, 2007.
- [330] G. Robins and P. Pattison, "Interdependencies and social processes: generalized dependence structures," in *Models and Methods in Social Network Analysis*, P. Carrington, J. Scott, and S. Wasserman, Eds. New York: Cambridge University Press, 2004.

- [331] G. Robins, P. Pattison, and P. Elliott, “Network models for social influence processes,” *Psychometrika*, vol. 66, no. 2, pp. 161–190, 2001.
- [332] D. Robinson and L. Foulds, “Comparison of phylogenetic trees,” *Mathematical Biosciences*, vol. 53, no. 1–2, pp. 131–147, 1981.
- [333] S. Ross, *Introduction to Probability Models, Eighth Edition*. New York: Academic Press, 2003.
- [334] S. Roweis and L. Saul, “Nonlinear dimensionality reduction by locally linear embedding,” *Science*, vol. 290, no. 5500, pp. 2323–2326, 2000.
- [335] H. Rue and L. Held, *Gaussian Markov Random Fields: Theory and Applications*. Boca Raton, FL: Chapman & Hall/CRC, 2005.
- [336] G. Sabidussi, “The centrality index of a graph,” *Psychometrika*, vol. 31, pp. 581–683, 1966.
- [337] K. Sachs, O. Perez, D. Pe’er, D. Lauffenburger, and G. Nolan, “Causal protein-signaling networks derived from multiparameter single-cell data,” *Science*, vol. 308, no. 5721, pp. 523–529, 2005.
- [338] H. Salgado, A. Santos-Zavaleta, S. Gama-Castro, D. Millán-Zárate, E. Díaz-Pereido, F. Sánchez-Solano, E. Pérez-Rueda, C. Bonavides-Martínez, and J. Collado-Vides, “RegulonDB (version 3.2): transcriptional regulation and operon organization in *Escherichia coli* K-12,” *Nucleic Acids Research*, vol. 29, no. 1, pp. 72–74, 2001.
- [339] C. Särndal, B. Swensson, and J. Wretman, *Model Assisted Survey Sampling*. New York: Springer-Verlag, 1992.
- [340] J. Schäfer and K. Strimmer, “An empirical Bayes approach to inferring large-scale gene association networks,” *Bioinformatics*, vol. 21, no. 6, pp. 754–764, 2005.
- [341] ———, “A shrinkage approach to large-scale covariance matrix estimation and implications for functional genomics,” *Statistical Applications in Genetics and Molecular Biology*, vol. 4, no. 1, p. A32, 2005.
- [342] B. Schölkopf and A. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. Cambridge, MA: MIT Press, 2002.
- [343] D. Scholtens and R. Gentleman, “Making sense of high-throughput protein-protein interaction data,” *Statistical Applications in Genetics and Molecular Biology*, vol. 3, no. 1, p. A39, 2004.
- [344] J. Scott, *Social Network Analysis: A Handbook, Second Edition*. Newbury Park, CA: Sage Publications, 2004.
- [345] A. Sen and T. Smith, *Gravity Models of Spatial Interaction Behavior*. Berlin: Springer-Verlag, 1995.
- [346] T. Sen, A. Kloczkowski, and R. Jernigan, “Functional clustering of yeast proteins from the protein-protein interaction network,” *BMC Bioinformatics*, vol. 7, pp. 355–367, 2006.
- [347] R. Sharan, I. Ulitsky, and R. Shamir, “Network-based prediction of protein function,” *Molecular Systems Biology*, vol. 3, p. 88, 2007.

- [348] Y. Shavitt, X. Sun, A. Wool, and B. Yener, "Computing the unmeasured: an algebraic approach to Internet mapping," in *Proceedings of the IEEE INFOCOM*, 2001, pp. 1646–1654.
- [349] H. Sherali and T. Park, "Estimation of dynamic origin–destination trip tables for a general network," *Transportation Research Part B*, vol. 35, no. 3, pp. 217–235, 2001.
- [350] B. Shipley, *Cause and Correlation in Biology: A User's Guide to Path Analysis, Structural Equations, and Causal Inference*. New York: Cambridge University Press, 2002.
- [351] A. Shriram and J. Kaur, "Empirical evaluation of techniques for measuring available bandwidth," in *Proceedings of the IEEE INFOCOM 2007*, 2007, pp. 2162–2170.
- [352] R. Shumway and D. Stoffer, "An approach to time series smoothing and forecasting using the em algorithm," *Journal of Time Series Analysis*, vol. 3, no. 4, pp. 253–264, 1982.
- [353] H. Simon, "On a class of skew distribution functions," *Biometrika*, vol. 42, pp. 425–440, 1955.
- [354] H. Singhal and G. Michailidis, "Identifiability of flow distributions from link measurements with applications to computer networks," *Inverse Problems*, vol. 23, no. 5, pp. 1821–1849, 2007.
- [355] V. Smith, J. Yu, T. Smulders, A. Hartemink, E. Jarvis, and K. Friston, "Computational inference of neural information flow networks," *PLoS Computational Biology*, vol. 2, no. 11, pp. 1436–1449, 2007.
- [356] A. Smola and R. Kondor, "Kernels and regularization on graphs," in *Proceedings of the 16th Annual Conference on Learning Theory (COLT)*, 2003.
- [357] T. Snijders, "Markov chain Monte Carlo estimation of exponential random graph models," *Journal of Social Structure*, vol. 3, no. 2, pp. 1–40, 2002.
- [358] ——, "Statistical methods for network dynamics," in *Proceedings of the XLIII Scientific Meeting, Italian Statistical Society*, S. Luchini *et al.*, Eds., 2006, pp. 281–296.
- [359] ——, "Enumeration and simulation methods for 0 – 1 matrices with given marginal totals," *Psychometrika*, vol. 57, pp. 397–417, 1991.
- [360] T. Snijders, P. Pattison, G. Robins, and M. Handcock, "New specifications for exponential random graph models," *Sociological Methodology*, vol. 36, no. 1, pp. 99–153, 2006.
- [361] T. Snijders, C. Steglich, and M. Schweinberger, "Modeling the co-evolution of networks and behavior," in *Longitudinal Models in the Behavioral and Related Sciences*, K. van Montfort, H. Oud, and A. Satorra, Eds. New York: Lawrence Erlbaum, 2007, pp. 41–71.
- [362] R. Sokal and C. Michener, "A statistical method for evaluating systematic relationships," *University of Kansas Science Bulletin*, vol. 28, pp. 1409–1438, 1958.
- [363] J. Sommers, P. Barford, N. Duffield, and A. Ron, "Accurate and efficient SLA compliance monitoring," in *Proceedings of the ACM SIGCOMM*, 2007.

- [364] A. Soule, A. Lakhina, N. Taft, K. Papagiannaki, K. Salamatian, A. Nucci, M. Crovella, and C. Diot, “Traffic matrices: balancing measurements, inference and modeling,” *Proceedings of the 2005 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, pp. 362–373, 2005.
- [365] A. Soule, K. Salamatian, A. Nucci, and N. Taft, “Traffic matrix tracking using Kalman filters,” in *Proceedings of the ACM Sigmetrics, Special Issue on the First ACM Sigmetrics Workshop on Large Scale Network Inference*, 2005, pp. 24–31.
- [366] D. Spielman and S. Teng, “Spectral partitioning works: planar graphs and finite element meshes,” *Linear Algebra and Its Applications*, vol. 421, no. 2–3, pp. 284–305, 2007.
- [367] C. Stark, B. Breitkreutz, T. Reguly, L. Boucher, A. Breitkreutz, and M. Tyers, “BioGRID: a general repository for interaction datasets,” *Nucleic Acids Research*, pp. D535–539, 2006.
- [368] J. Stewart, “An inverse distance variation for certain social influences,” *Science*, vol. 93, no. 2404, pp. 89–90, 1941.
- [369] M. Stoer and F. Wagner, “A simple min-cut algorithm,” *Journal of the ACM*, vol. 44, no. 4, pp. 585–591, 1997.
- [370] J. Storey, “A direct approach to false discovery rates,” *Journal of the Royal Statistical Society Series B*, vol. 64, no. 3, pp. 479–498, 2002.
- [371] D. Strauss, “Clustering on colored lattices,” *Journal of Applied Probability*, vol. 14, pp. 135–143, 1977.
- [372] D. Strauss and M. Ikeda, “Pseudolikelihood estimation for social networks,” *Journal of the American Statistical Association*, vol. 85, no. 409, pp. 204–212, 1990.
- [373] M. Stumpf and C. Wiuf, “Sampling properties of random graphs: the degree distribution,” *Physical Review E*, vol. 72, no. 3, p. 36118, 2005.
- [374] M. Stumpf, C. Wiuf, and R. May, “Subnets of scale-free networks are not scale-free: sampling properties of networks,” *Proceedings of the National Academy of Sciences*, vol. 102, no. 12, pp. 4221–4224, 2005.
- [375] K. Sugiyama, S. Tagawa, and M. Toda, “Methods for visual understanding of hierarchical system structures,” *IEEE Transactions on Systems, Man and Cybernetics*, vol. 11, no. 2, pp. 109–125, 1981.
- [376] B. Taskar, M. Wong, P. Abbeel, and D. Koller, “Link prediction in relational data,” *Advances in Neural Information Processing Systems*, vol. 16, 2004.
- [377] C. Tebaldi and M. West, “Bayesian inference on network traffic using link count data,” *Journal of the American Statistical Association*, vol. 93, pp. 557–573, 1998.
- [378] J. Tenenbaum, V. Silva, and J. Langford, “A global geometric framework for nonlinear dimensionality reduction,” *Science*, vol. 290, no. 5500, pp. 2319–2323, 2000.
- [379] S. Thompson, *Sampling, Second Edition*. New York: Wiley & Sons, Inc., 2002.

- [380] R. Tibshirani, “Regression shrinkage and selection via the lasso,” *Journal of the Royal Statistical Society, Series B*, vol. 58, pp. 267–288, 1996.
- [381] Y. Tsang, M. Coates, and R. Nowak, “Network delay tomography,” *IEEE Transactions on Signal Processing*, vol. 51, no. 8, pp. 2125–2136, 2003.
- [382] E. Tufte, *The Visual Display of Quantitative Information*. Graphics Press, 1983.
- [383] J. Tukey, *Exploratory Data Analysis*. New York: Addison-Wesley, 1977.
- [384] P. Turán, “On an extremal problem in graph theory,” *Matematikai és Fizikai Lapok*, vol. 48, pp. 436–452, 1941.
- [385] H. van Zuylen and L. Willumsen, “The most likely trip matrix estimated from traffic counts,” *Transportation Research, Part B*, vol. 14, no. 3, pp. 281–294, 1980.
- [386] Y. Vardi, “Network tomography: estimating source-destination traffic intensities from link data,” *Journal of the American Statistical Association*, vol. 91, no. 433, pp. 365–377, 1996.
- [387] A. Vázquez, R. Pastor-Satorras, and A. Vespignani, “Large-scale topological and dynamical properties of the Internet,” *Physical Review E*, vol. 65, no. 6, p. 66130, 2002.
- [388] F. Viger, A. Barrat, L. Dall’Asta, C. Zhang, and E. Kolaczyk, “What is the real size of a sampled network? The case of the Internet,” *Physical Review E*, vol. 75, no. 5, p. 56111, 2007.
- [389] G. Wahba, *Spline Models for Observational Data*. Philadelphia: SIAM, 1990.
- [390] M. Wainwright, P. Ravikumar, and J. Lafferty, “High-dimensional graphical model selection using ℓ_1 -regularized logistic regression,” *Advances in Neural Information Processing Systems (NIPS)*, vol. 19, p. 1465, 2007.
- [391] J. Ward, “Hierarchical grouping to optimize an objective function,” *Journal of the American Statistical Association*, vol. 58, no. 301, pp. 236–244, 1963.
- [392] L. Wasserman, *All of Statistics: A Concise Course in Statistical Inference*. New York: Springer, 2004.
- [393] S. Wasserman and K. Faust, *Social Network Analysis: Methods and Applications*. New York: Cambridge University Press, 1994.
- [394] S. Wasserman and P. Pattison, “Logit models and logistic regressions for social networks: I. An introduction to Markov graphs and p^* ,” *Psychometrika*, vol. 61, no. 3, pp. 401–425, 1996.
- [395] D. Watts, *Six Degrees: The Science of a Connected Age*. New York: WW Norton & Company, 2003.
- [396] D. Watts and S. Strogatz, “Collective dynamics of ‘small-world’ networks,” *Nature*, vol. 393, no. 6684, pp. 440–442, 1998.
- [397] I. Weir and A. Pettitt, “Spatial modelling for binary data using a hidden conditional autoregressive Gaussian process: a multivariate extension of the probit model,” *Statistics and computing*, vol. 9, pp. 77–86, 1999.
- [398] ———, “Binary probability maps using a hidden conditional autoregressive Gaussian process with an application to finnish common toad data,” *Journal*

- of the Royal Statistical Society: Series C (Applied Statistics)*, vol. 49, no. 4, pp. 473–484, 2000.
- [399] S. Weisberg, *Applied Linear Regression, Third Edition*. New York: John Wiley & Sons, 2005.
- [400] M. West and J. Harrison, *Bayesian Forecasting and Dynamic Models, Second Edition*. New York: Springer, 1999.
- [401] H. White, S. Boorman, and R. Breiger, “Social structure from multiple networks. I. Blockmodels of roles and positions,” *American Journal of Sociology*, vol. 81, no. 4, pp. 730–780, 1976.
- [402] S. White and P. Smyth, “Algorithms for estimating relative importance in networks,” in *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge, Discovery, and Data Mining*, 2003.
- [403] J. Whittaker, *Graphical Models in Applied Multivariate Statistics*. Chichester: Wiley & Sons, 1990.
- [404] P. Whittle, “The outcome of a stochastic epidemic – a note on Bailey’s paper,” *Biometrika*, vol. 42, pp. 116–122, 1955.
- [405] A. Wille and P. Bühlmann, “Low-order conditional independence graphs for inferring genetic networks,” *Statistical Applications in Genetics and Molecular Biology*, vol. 5, no. 1, p. A1, 2006.
- [406] A. Wille, P. Zimmermann, E. Vránova, A. Fürholz, O. Laule, S. Bleuler, L. Hennig, A. Prelić, P. Rohr, L. Thiele *et al.*, “Sparse graphical Gaussian modeling of the isoprenoid gene network in arabidopsis thaliana,” *Genome Biology*, vol. 5, no. 11, p. R92, 2004.
- [407] C. Wiuf, M. Brämeier, O. Hagberg, and M. Stumpf, “A likelihood approach to analysis of network data,” *Proceedings of the National Academy of Sciences*, vol. 103, no. 20, pp. 7566–7570, 2006.
- [408] S. Wuchty and E. Almaas, “Peeling the yeast protein network,” *Proteomics*, vol. 5, no. 2, pp. 444–449, 2005.
- [409] Q. Yang and S. Lonardi, “A parallel edge-betweenness clustering tool for protein-protein interaction networks,” *International Journal of Data Mining and Bioinformatics*, vol. 1, no. 3, pp. 241–247, 2007.
- [410] M. Yuan and Y. Lin, “Model selection and estimation in the Gaussian graphical model,” *Biometrika*, vol. 94, no. 1, p. 19, 2007.
- [411] W. Zachary, “An information flow model for conflict and fission in small groups,” *Journal of Anthropological Research*, vol. 33, no. 4, pp. 452–473, 1977.
- [412] C.-H. Zhang, “Estimation of sums of random variables: examples and information bounds,” *Annals of Statistics*, vol. 33, no. 5, pp. 2022–2041, 2005.
- [413] H. Zhang and Y. Lin, “Component selection and smoothing for nonparametric regression in exponential families,” *Statistica Sinica*, vol. 16, pp. 1021–1042, 2006.
- [414] Y. Zhang, M. Roughan, C. Lund, and D. Donoho, “An information-theoretic approach to traffic matrix estimation,” in *Proceedings of SIGCOMM’03*, 2003.

- [415] Y. Zhang, Z. Ge, A. Greenberg, and M. Roughan, “Network anomography,” *Proceedings of ACM/USENIX Internet Measurement Conference (IMC)*, 2005.
- [416] J. Zhu and T. Hastie, “Kernel logistic regression and the import vector machine,” *Journal of Computational and Graphical Statistics*, vol. 14, no. 1, pp. 185–205, 2005.
- [417] X. Zhu, J. Kandola, Z. Ghahramani, and J. Lafferty, “Nonparametric transforms of graph kernels for semi-supervised learning,” *Advances in Neural Information Processing Systems (NIPS)*, vol. 17, pp. 1641–1648, 2005.
- [418] X. Zhu, J. Kandola, J. Lafferty, and Z. Ghahramani, “Graph kernels by spectral transforms,” in *Semi-Supervised Learning*, O. Chapelle, B. Schölkopf, and A. Zien, Eds. Cambridge, MA: MIT Press, 2006.

Author Index

- Abbeel, P. 200, 201
Achlioptas, D. 150
Adamic, L.A. 201
Adar, E. 201
Agarwal, D.K. 75, 118
Aiello, W. 158
Albert, R. 173–175
Alderson, D. 193
Almaas, E. 95
Alon, U. 151, 165, 166, 168
Altaf-Ul-Amin, M. 95
Alvarez-Hamelin, J.I. 69, 73
Amenta, N. 235
Anderson, R.M. 112, 272
Anderson, T.W. 213
Andersson, H. 275, 276, 278, 279
Argyriou, A. 266
- Bader, G.D. 95
Bader, J.S. 200
Ball, F. 276
Bandyopadhyay, S. 161
Barabási, A.L. 173–175
Barford, P. 328
Barrat, A. 69, 73, 96, 171, 271, 276, 280
Barthélemy, J.-P. 236
Barthélemy, M. 93, 271, 276, 280
Batagelj, V. 13, 73, 95, 104, 111, 159,
 160, 172, 175, 196
Baur, M. 73
- Becker, N.G. 275, 276
Belkin, M. 262, 282
Bell, M.G.H. 297, 301, 317
Bell, R. 75, 118
Bender-deMoll, S. 118
Benjamini, Y. 35, 211, 212
Berger, J.O. 232
Besag, J. 181, 187, 250, 251, 253, 254,
 340
Bickel, P. 329
Bindel, D. 322, 323, 326
Bing, N. 216, 241
Bishop, C.M. 343
Blum, A. 269
Boccaletti, S. 114, 119, 171, 175, 194
Bollback, J.P. 234
Bollobás, B. 45, 96, 97, 156, 169, 174,
 175, 193
Bonacich, P. 90
Boorman, S.A. 111
Borgatti, S.P. 88, 92, 93
Börner, K. 65
Bourque, M. 235
Boyack, K.W. 65, 67, 77
Brameier, M. 178
Brandes, U. 62, 73, 75, 76, 88–90, 93,
 95, 97, 99, 106, 108, 111, 119, 120,
 151, 159, 160, 172, 175, 196
Breiger, R.L. 111
Breiman, L. 218

- Brin, S. 92
 Britton, T. 275, 276, 278, 279
 Brockwell, P.J. 281, 313
 Broder, A. 100, 101
 Bryant, D. 224, 235
 Bühlmann, P. 216, 219
 Bunge, J. 151
 Burk, W.J. 281, 282
 Byers, J.W. 150
 Caceres, R. 230, 232, 320, 321
 Caldarelli, G. 97
 Campbell, D. 281
 Cao, J. 281, 307, 310
 Carey, H.C. 288
 Cascetta, E. 298, 299, 301, 305–307
 Casella, G. 45, 255
 Castro, R.M. 224, 225, 236, 237, 239,
 240, 318
 Chant, J. 200
 Chaudhuri, A. 200
 Chawla, S. 269
 Chen, G. 309
 Chen, T. 271
 Chen, Y. 160, 161, 164, 322, 323, 326
 Chor, B. 233
 Christensen, R. 295, 326
 Chua, D.B. 93, 323, 324, 326, 327
 Chui, C.K. 309
 Chung, F.R.K. 106, 158, 176–179, 193
 Clarke, F. 235
 Clauset, A. 106, 150
 Cleveland, W.S. 76
 Clifford, P.E. 339
 Coates, M.J. 224, 236–240, 318–320,
 327
 Conroy, J.M. 118, 119
 Cooper, C. 177
 Cormen, T.H. 24, 45, 98
 Cover, T.M. 305
 Cressie, N.A.C. 55, 250, 254, 256, 282,
 317
 Cristianini, N. 265, 266, 271
 Crovella, M. 4, 69, 72, 150, 298, 312,
 323, 324, 326, 327, 329
 Daley, D.J. 272, 274
 Dall'Asta, L. 69, 73, 136
 Darroch, J. N. 131
 Davidson, G.S. 67
 Davis, D. 307, 310
 Davis, G.A. 308, 310
 Davis, R.A. 281, 313
 de Haan, L. 84
 de la Fuente, A. 216, 241
 de Nooy, W. 13
 de Solla Price, D.J.D. 65, 173
 Deming, W.E. 294
 Demiris, N. 276
 Dempster, A.P. 217
 Deng, M. 271
 Dewey, T.G. 176–179
 di Battista, G. 58, 59, 76
 Diaconis, P. 160, 161, 164
 Diestel, R. 45
 Diot, C. 329
 Dobra, A. 220
 Dolev, S. 93
 Donoho, D.L. 282, 306, 311–313
 Doreian, P. 111, 118
 Dorogovtsev, S.N. 194
 Doyle, J.C. 86, 87, 193
 Drees, H. 84
 Drton, M. 217
 Duffield, N.G. 225, 229, 230, 232, 234,
 244, 319–321, 328
 Eades, P. 58, 59, 76
 Eames, K.T.D. 279, 282
 Edwards, D. 333
 Efron, B. 37, 212, 218, 219, 234
 Elisseeff, A. 265, 266
 Elliott, P. 256, 342
 Elovici, Y. 93
 Eppstein, D. 151
 Erdős, P. 156
 Erlebach, T. 88, 93, 95, 97, 99, 106,
 108, 111, 119
 Erramilli, V. 312
 Esty, W.W. 131
 Everett, M.G. 88, 92, 93

- Faith, J.J. 208, 220, 222, 223, 241
Faloutsos, C. 118
Faust, K. 76, 92, 111, 120, 156, 167
Felsenstein, J. 225, 228, 230, 233–236,
 241
Ferligoj, A. 111
Fiedler, M. 108
Figueiredo, M. 241
Fischer, M.M. 289
Fitzpatrick, M. 151
Fleischer, D. 93
Flynn, P.J. 103, 228
Ford, L.R. 100
Forster, J.J. 161
Fotheringham, A.S. 289
Foulds, L.R. 235
Frank, O. 135, 139, 140, 142, 145–147,
 150, 151, 162, 163, 182, 184, 341
Freeman, L.C. 49, 88, 89, 93
Friedman, J.H. 203, 204, 218, 220, 247,
 269, 282, 283, 293, 309
Friedman, N. 241, 282
Frieze, A. 177
Fronczak, A. 97
Fruchterman, T.M.J. 62
Fulkerson, D.R. 100

Gaertler, M. 73
Galas, D.J. 176–179
Gani, J. 272, 274
Gansner, E.R. 62
Garcia-Diaz, A. 285
Gardner, T.S. 208
Ge, Z. 329
Geman, D. 252, 256
Geman, S. 252, 256
Gentleman, R. 241
Getoor, L. 281, 282, 344
Geyer, C.J. 185
Ghahramani, Z. 266, 309
Gibson, D. 111
Gilbert, E.N. 156
Girvan, M. 102, 105, 110
Gkantsidis, C. 106–108
Godsil, C.D. 108

Goldberg, A.V. 100
Goldberg, D.S. 200
Goldstein, M.L. 83
Golightly, A. 281
Golub, G.H. 90, 110, 323
Gomory, R.E. 100
Good, I.J. 130
Gopal, S. 10, 289
Gordon, A.D. 228
Greenberg, A. 329
Grimes, C. 282
Gross, J.L. 45, 58
Gu, C. 262
Guare, J. 1, 98
Gupta, S. 112

Hagberg, O. 178
Hage, P. 92
Hammersley, J.M. 339
Han, J.D.J. 126, 150
Handcock, M.S. 183, 184, 186,
 188–190, 194, 204
Harary, H. 92
Harrison, J. 281
Hastie, T. 203, 204, 218–220, 247, 261,
 269, 282, 283, 293, 309
Heckerman, D. 333
Heesterbeek, J.A.P. 275
Held, L. 252
Herbster, M. 266
Hernández-Campos, F. 85
Hero, A.O. 224, 318
Hill, S.B. 75, 118
Hinton, G.E. 309
Hochberg, Y. 35, 211
Hoeschele, I. 216, 241
Hoff, P.D. 194, 200, 204, 205, 207
Hogue, C.W.V. 95
Holland, P.W. 184
Holmes, S.P. 160, 161, 164, 225, 236,
 242
Hołyst, J.A. 97
Hooker, G. 281
Horowitz, J. 225, 229, 230, 232, 234,
 244, 319–321

- Hotelling, H. 210, 214, 218
 Hu, T.C. 100
 Huelsenbeck, J.P. 234
 Hunter, D.R. 184, 186, 188–190
 Huson, D.H. 224
 Iida, Y. 317
 Ikeda, M. 187
 Ising, E. 249
 Itzkovitz, S. 151, 165, 166, 168
 Jackson, M.O. 193
 Jain, A.K. 103, 228
 Jana, R. 161
 Jedynak, M. 97
 Jeong, H. 126
 Jernigan, R.L. 102, 107
 Johnson, R.A. 219
 Johnstone, I. 219
 Jordan, M.I. 271, 333, 342–344
 Kahng, B. 177
 Kamada, T. 62
 Kandola, J. 265, 266
 Kashtan, N. 151, 165, 166, 168
 Kasif, S. 266, 271
 Kasturirangan, R. 172
 Katz, L. 90
 Katz, R.H. 322, 323, 326
 Kaufmann, M. 59, 75, 76
 Kaur, J. 329
 Kawai, S. 62
 Keeling, M. 279, 282
 Kempe, D. 118, 150
 Kermack, W.O. 272
 Kim, J. 177
 Kim, P.J. 126
 Kimeldorf, G. 259
 Kirsch, H.E. 114–116
 Klavans, R. 65, 77
 Kleinberg, J.M. 92, 111, 118, 172, 176,
 200–202
 Kloczkowski, A. 102, 107
 Kolaczyk, E.D. 93, 114–116, 271, 323,
 324, 326, 327
 Koller, D. 200, 201, 282
 Kondor, R.I. 263
 König, D. 2
 Koren, Y. 62
 Koskinen, J.H. 194
 Kossinets, G. 56, 118, 119
 Kramer, M.A. 114–116
 Krapivsky, PL 177
 Krishnamurthy, B. 4, 69, 72, 298, 329
 Kumar, A. 118
 Kumar, R. 118, 119, 176, 177
 Lafferty, J.D. 220, 263, 266
 Lakhina, A. 150, 311, 329
 Lanckriet, G.R.G. 266, 271
 Langford, J.C. 282
 Lauritzen, S.L. 217, 219, 333, 335, 339,
 341, 343
 Lawrence, E. 319, 321
 Lazega, E. 188
 Lee, S.H. 126
 Leinhardt, S. 184
 Leiserson, C.E. 24, 45, 98
 Leland, W.E. 328
 Leskovec, J. 118
 Leyvraz, F. 174
 Li, L. 193
 Li, M. 235
 Li, S.Z. 250, 252, 254
 Liang, F. 266
 Liang, G. 319, 321
 Liben-Nowell, D. 200–202
 Liljeros, F. 279
 Lin, Y. 220, 266
 Linial, M. 241
 Liseo, B. 232
 Little, R.J.A. 55, 200
 Liu, J.S. 45, 120, 160, 161, 164, 255
 Lo Presti, F. 225, 229, 230, 232, 234,
 244, 319–321
 Lonardi, S. 110
 Lu, L. 158, 176–179, 193
 Lund, C. 306, 311–313
 MacDonald, I.L. 337
 Maher, M.J. 301

- Marchette, D.J. 118, 119
Margush, T. 235
Marron, J.S. 85
Marsden, P.V. 76
Maslov, S. 86, 87
Matveeva, I. 262
May, R.M. 112, 150, 272
McCanne, S. 229
McCullagh, P. 257, 292, 293
McDonald, J.W. 161
McFarland, D. 118
McKay, B.D. 157
McKendrick, A.G. 272
McMorris, F.R. 235, 236
Meinshausen, N. 219
Mendes, J.F.F. 194
Mendes, P. 216, 241
Michailidis, G. 302, 319, 321
Michener, C.D. 240
Mihail, M. 106–108
Milgram, S. 1, 98, 169
Milo, R. 151, 160, 161, 165, 166, 168
Mitzenmacher, M. 85, 173
Mohar, B. 108
Mollison, D. 276
Mollory, M. 158, 160
Molloy, M. 158
Monasson, R. 172
Monro, S. 186
Moody, J. 118
Moore, C. 106, 150
Moors, T. 327
Morris, M. 180
Morris, S.A. 83
Motwani, R. 92
Mrvar, A. 13, 73, 95
Murali, T.M. 266
Murty, M.N. 103, 228
Nachman, I. 241
Nair, V.N. 319, 321
Nariai, N. 271
Nelder, J.A. 257, 292, 293
Newman, M.E.J. 3, 83, 87, 93, 97, 102, 105, 106, 110, 112, 113, 119, 159, 171, 172, 175, 194, 278
Nguyen, H.X. 322, 328
Nguyen, S. 301, 305, 306
Ni, J. 242
Nielsen, R. 234
Nihan, N.L. 308, 310
Niyogi, P. 262, 282
Nobel, W.S. 271
Nordvik, M.K. 279
North, S. 62
Novak, J. 118, 119
Nowak, R.D. 224, 236, 237, 239–241, 318–320
Nowicki, K. 194, 204
Nucci, A. 308, 313, 314
Ohno, S. 176
O’Kelly, M.E. 289
O’Neill, P.D. 276, 279
Ortúzar, J.D. 298, 317
Page, L. 92
Park, K. 329
Park, T. 307
Park, Y. 118, 119
Pastor-Satorras, R. 83, 87, 97
Pattison, P.E. 180, 183, 184, 188–190, 256, 342
Pearl, J. 343
Pe’er, D. 241
Perlman, M.D. 217
Pettitt, A.N. 256, 257
Phillips, D.T. 285
Pich, C. 62, 151
Pointurier, Y. 327
Pontil, M. 266
Popescul, A. 200
Priebe, C.E. 118, 119
Puzy, R. 93
Qazi, S. 327
Rabbat, M.G. 224, 241, 320, 327
Raftery, A.E. 194, 204

- Raghavan, P. 111, 118, 119
 Ramsay, J.O. 281
 Rao, A.R. 161
 Ratcliff, D. 131
 Ratnasamy, S. 229
 Ravikumar, P. 220
 Redner, S. 174, 177
 Reed, B.A. 158, 160
 Reingold, E.M. 62
 Rényi, A. 156
 Resnick, S. 84
 Riordan, O.M. 96, 97, 169, 174, 175
 Rivest, R.L 24, 45, 98
 Robbins, H. 186
 Robert, C.P. 45, 255
 Roberts, J.M. 161
 Robillard, P. 297, 299
 Robins, G.L. 180, 183, 184, 187–190,
 256, 342
 Robinson, D.F. 235
 Ron, A. 328
 Ronquist, F. 234
 Ross, S.M. 45, 273
 Roth, F.P. 200
 Rothberg, J.M. 200
 Roughan, M. 306, 311–313, 329
 Roweis, S.T. 282
 Royle, G. 108
 Rubin, D.B. 55, 200
 Rue, H. 252
 Sabidussi, G. 88
 Sachs, K. 241
 Salamatian, K. 308, 313, 314
 Salgado, H. 220
 Samorodnitsky, G. 85
 Samukhin, AN 174
 Särndal, C.E. 155
 Saul, L.K. 282
 Scalia-Tomba, G. 276
 Schäfer, J. 218
 Schölkopf, B. 261
 Scholtens, D. 241
 Schweinberger, M. 281, 282
 Scott, J. 76, 111, 119
 Sen, A.K. 288, 289, 292–294, 296, 297
 Sen, T.Z. 102, 107
 Shamir, R. 267, 281
 Sharan, R. 267, 281
 Shavitt, Y. 322
 Shawe-Taylor, J. 265, 266
 Sherali, H.D. 307
 Shipley, B. 241
 Shriram, A. 329
 Shumway, R.H. 309
 Sienkiewicz, J. 97
 Silva, V. 282
 Simon, H.A. 173
 Singhal, H. 302
 Smith, F.D. 85
 Smith, P.W.F. 161
 Smith, T.E. 288, 289, 292–294, 296,
 297
 Smith, V.A. 241
 Smola, A.J. 261, 263
 Smyth, P. 92, 93, 111
 Sneppen, K. 86, 87
 Snijders, T.A.B. 145–147, 161–164,
 183, 184, 186–190, 194, 204, 281,
 282
 Sokal, R.R. 240
 Sommers, J. 328
 Song, H. 322, 323, 326
 Soule, A. 307, 308, 311, 313, 314
 Spencer, J. 174
 Spiegelhalter, D.J. 343
 Spielman, D.A. 109
 St. John, K. 235
 Stark, C. 81
 Steglich, C.E.G. 281, 282
 Stein, C. 24, 45, 98
 Stephan, F.F. 294
 Stewart, J.Q. 288
 Stoer, M. 100
 Stoffer, D.S. 309
 Stokman, F.N. 118
 Storey, J.D. 35, 211, 218
 Strauss, D.J. 182, 184, 187, 252, 341
 Strimmer, K. 218
 Strogatz, S.H. 96, 98, 159, 169

- Stumpf, M.P.H. 150, 178
Sugiyama, K. 63
Sun, F. 271
Sun, X. 322
Swensson, B. 155

Taft, N. 308, 312–314
Tagawa, S. 63
Tamassia, R. 58, 59, 76
Tarjan, R.E. 100
Taskar, B. 200, 201, 281, 282, 344
Tatikonda, S. 242
Tebaldi, C. 303, 304
Tenenbaum, J.B. 282
Teng, S.H. 109
Thiran, P. 322, 328
Thomas, J.A. 305
Thompson, E.A. 185
Thompson, S. 151
Tibshirani, R.J. 37, 203, 204, 218–220,
 234, 247, 282, 283, 293, 309
Toda, M. 63
Tollis, I.G. 58, 59, 76
Tomkins, A. 118, 119
Towsley, D.F. 225, 229, 230, 232, 234,
 244, 319–321
Tsang, Y. 319
Tufte, E.R. 50, 76
Tukey, J.W. 76
Tuller, T. 233
Turán, P. 94
Tusnady, G. 174

Ulitsky, I. 267, 281
Ungar, L.H. 200

van Loan, C.F. 90, 110, 323
van Zuylen, H.J. 297, 305
Vardi, Y. 224, 298, 302, 303, 307, 311
Vázquez, A. 87
Vespignani, A. 69, 73, 83, 87, 97, 271,
 276, 280
Viger, F. 147, 148
Volinsky, C. 75, 118

Wagner, D. 59, 73, 75, 76
Wagner, F. 100
Wahba, G. 259, 262
Wainwright, M.J. 220
Wang, J. 151
Ward, J.H. 104
Wasserman, L. 12, 39, 45
Wasserman, S. 76, 92, 111, 120, 156,
 167, 180
Watts, D.J. 96, 98, 118, 119, 159, 169,
 172
Weigt, M. 96, 171
Weir, I.S. 256, 257
Weisberg, S. 297
West, M. 281, 303, 304
White, D.R. 93
White, H.C. 111
White, S. 92, 93, 111
Whittaker, J. 217, 333
Whittle, P. 275
Wichern, D.W. 219
Wiel, S.V. 307, 310
Wilkinson, D.J. 281
Wille, A. 215, 216
Willinger, W. 193, 329
Willumsen, L.G. 297, 298, 305, 317
Winograd, T. 92
Wiuf, C. 150, 178
Wolpert, R.L. 232
Wong, M.F. 200, 201
Wool, A. 322
Wormald, N.C. 157
Wretman, J. 155
Wu, C.J. 266
Wuchty, S. 95
Wylie, B.N. 67

Xie, H. 242
Xie, P. 150

Yang, Q. 110
Yang, Y.R. 242
Yekutieli, D. 212
Yellen, J. 45, 58
Yen, G.G. 83
Yener, B. 322

- Yu, B. 224, 307, 310, 318, 319, 321
Yuan, M. 220
Zachary, W.W. 6
Zaliznyak, A. 86, 87
Zaversnik, M. 73, 95
Zegura, E. 106–108
Zhang, C-H. 149
Zhang, H.H. 266
Zhang, L. 235
Zhang, Y. 306, 311–313, 329
Zhu, J. 261, 269
Zhu, X. 266
Zucchini, W. 337

Subject Index

A

acyclic 17
adjacency matrix 20
adjacent 16
alignment 265
anomaly
 in graph structure 119
 in network traffic 316, 327, 329
anomography 329
arc 16
 head 16
 mutual 16
 tail 16
association network 207
assortativity coefficient 112
auto-probit model 256
autonomous system 52, 70, 310

B

bagging *see* bootstrap aggregation
Barabási-Albert model 173, 277
Bayes rule 25
Bayesian network 344
bipartite *see* graph
bisection
 spectral 108–109
block modeling 111
Bonferroni correction 35
bootstrap 37, 234
 aggregation 218
bowtie 100–101
breadth first search 23

C

capture-recapture 129–130, 145

cascade process 229–233
centrality 88–93, 151, 187
 centralization index 93
 edge 93, 137, 324
 group centrality 93
 vertex 80
 betweenness 89–90, 137, 151
 closeness 88–89
 eigenvector 90
chain graph 344
circuit 17
clique 18
clustering coefficient 96–97, 137, 142, 157, 159, 164–165, 169–171
 higher order 97
 local 96
community detection 102
complete graph *see* graph
component 17
 giant 94, 98, 157–158
computational complexity 23–24
consensus tree 235
concentration graph *see* conditional
 independence graph
concentration matrix *see* precision matrix
conditional
 expectation 26
 probability 25
 probability density function 26
conditional independence graph 216
conditionally independent
 events 25
 random variables 26
connected 17
 strongly 17
 weakly 17
connected triple 96

- consensus tree 235
 copying 176–178
 core 95
 k-core 73, 95
 correlation 26, 42
 empirical 210
 partial 212–213
 Pearson product moment 209
 cost
 generalized 286
 covariance 26
 coverage estimator 131
 cross-validation 203, 205, 219, 267
 cumulative distribution function 25
 joint 26
 cut 100
 minimum 100
 cycle 17
- D**
- datasets
 traceroute 147–148
 Abilene 4
 delay 327
 routing 310–316, 324–325, 327
 topology 112
 traffic 310–316
 AIDS blog 10, 101, 166–167
 Austrian telephone call 289–290, 293–294
 circadian clock, in *Drosophila melanogaster* 8, 51–52, 81
 epileptic seizure 114–116
 karate club 6, 96, 104, 107, 108, 164–165
 lawyer collaboration 188–191, 205–206, 246–247, 264–266
 microarray, in *Escherichia coli* 210–211, 214–215, 220–223
 packet delay, sandwich probing 236–241
 protein function, in *Saccharomyces cerevisiae* 266–271
 protein interactions, in *Saccharomyces cerevisiae* 81–82, 86, 125, 140, 142, 266–271
 router-level Internet 69–74, 81–82, 86
 scientific citation 65–68
 decorated graph 17
 degree 16, 80
 average, estimation of 124–125, 137, 144–145, 148
 centrality *see* centrality
 correlation 86–88, 113
 distribution 81–85, 184, 191, 192
 fitting 83–85, 149–150, 178–179
 power law 82, 158–159, 163, 173–178, 195
 skewed 81, 160
 in-degree 16
 out-degree 16
 sequence 16, 133, 158, 160
 dendrogram 103
 density 95, 118
 depth first search 23
 diameter 17, 118, 157, 158, 169, 170, 175
 diffusion kernel *see* kernel
 digraph *see* graph, directed
 Dijkstra's algorithm 24, 89
 dissimilarity 104, 230
 distance
 between graphs 75
 between vertices 17
 average 98, 158, 159, 170
 geodesic 17, 89, 121, 191, 201
 drawing, graph 58–63
 aesthetics 59, 78
 conventions 59
 duplication *see* copying
 dyad 184
 dynamic process 246, 271–281
- E**
- eccentricity 92
 edge 16
 betweenness *see* centrality, edge
 directed 16
 proper 16
 weight 17
 ensemble 153
 entropy 305–306
 maximum 289, 306
 relative 305
 epidemic 272–279
 basic reproduction number 274, 277, 284
 general epidemic model 272
 inference for 275–276, 279
 network-based models of 276–279
 Erdős-Rényi model 156, 277
 ERGM *see* network graph model,
 exponential random graph model
 estimable 299, 330
 estimation 31, 154–155
 Bayesian 36, 204, 234, 279, 301, 303–306
 least squares 38, 83, 296, 299–301
 complexity penalized 41, 218–220, 259, 306, 307

- maximum likelihood 33, 37, 39, 41, 84, 120, 179, 185–187, 202, 231–234, 253, 275, 292, 302–303
 maximum pseudo-likelihood 187, 253, 321
 method of moments 33, 37, 162–163
 of group size 129–131
 profile maximum likelihood 231
 unbiased 34, 127–129, 137, 142
 expectation 25
 conditional *see* conditional expectation
 exponential family 180, 181, 186, 341
- F**
- factor graph 344
 factorization 335, 339, 341
 false discovery rate 35, 116, 211–212, 216, 223
 positive 35, 218
 Fiedler
 value 108
 vector 108, 121
 Fisher’s transformation 210, 214, 217
 flow 285
 and centrality based on 93
 max-flow problem 100
 forest 18
 Frobenius
 inner-product 266
 norm 324
- G**
- Gaussian distribution 28
 multivariate 39
 Gaussian graphical model 216–220, 340–341
 geodesic distance *see* distance, between vertices, geodesic
 Gibbs random field 250
 Gibbs sampling 186, 255, 282, 304
 global Markov property 339
 graph 16
 acyclic 17
 barbell 99
 bipartite 19
 complete 18
 directed 16
 directed acyclic 19
 dual 93
 planar 19
 regular 18
 simple 16
- graphical model
 directed 335
 Gaussian *see* Gaussian graphical model
 gravity model 287–297, 299, 306, 311
- H**
- Hammersley-Clifford theorem 181, 182, 194, 250, 339–342
 hidden Markov model 337
 hidden population
 estimating the size of 145–147, 162–163
 hierarchical clustering 103–106, 110, 228–237
 linkage, single- and complete- 104
 Ward’s method 104
 Hill estimator 84
 homophily 43, 190, 204
 hubs and authorities 92, 111
 hypothesis testing 32
p-value 35
 alternative hypothesis 32
 multiple testing problem 35
 null hypothesis 32
 power 35
 significance level 35
- I**
- incidence matrix 21
 incident 16
 incident subgraph sampling 132–133
 independent
 conditionally *see* conditionally independent
 dent
 events 25
 random variables 26
 induced subgraph sampling 57, 131–132, 168
 irreducible
 graph 178
 Markov chain *see* Markov chain
 Ising model 251
 isomorphic 16, 179
 isoperimetric number 108
 iterative proportional fitting 294
 iteratively reweighted least squares 202, 293
- J**
- Jaccard coefficient 66, 201
- K**
- k*-core *see* core

Kalman filtering 281, 308–310, 337
 kernel
 diffusion 263
 function 258
 Laplacian 262
 regression 257–266
 Königsberg bridge problem 2
 kriging 317
 network 326
 Kullback-Liebler divergence 305

L

Lanczos algorithm 110
 Laplacian
 normalized 46
 Laplacian, graph 21, 108, 111, 121, 262
 LARS 219
 Lasso 219
 layout, graph
 directed 63
 egocentric 62
 energy-placement 62
 Fruchterman-Reingold 62
 Kamada-Kawai 62
 large graphs 62
 planar 60
 spring-embedder 61
 trees 60
 length 17
 link 16
 link prediction 199–206
 link tracing 136
 loop 16

M

manifold learning 120, 281
 Markov chain 27
 ergodic 27, 161
 irreducible 27, 161
 stationary distribution 27
 Markov chain Monte Carlo 30, 159, 161–162, 186, 255
 Markov condition 335
 Markov random field 249–256
 auto-logistic 251, 253
 auto-models 251–252
 parameter inference 253–254
 prediction 255–256
 max-sum algorithm 343
 maximum entropy *see* entropy
 maximum likelihood *see* estimation
 mean-squared error 34

Menger's theorem 99, 118
 message-passing algorithm 343
 method of moments *see* estimation
 min-cut *see* cut
 minimum spanning tree 24
 missingness 200, 202, 203, 242, 256
 mixing 111–113
 modularity 105, 110, 112, 113
 moral graph 343
 motif, network 151, 165–168
 multi-edge 16
 multi-graph 16

N

nearest neighbor 246–249
 network 1
 network cohesion 94
 network graph 2
 network graph model 153
 agent-based modeling 193
 copying model 176–178
 exponential random graph model 180–191, 341–342
 degeneracy 188
 fitting 185–187
 preferential attachment model 173–175
 random graph model 154–168
 classical 156–157
 generalized 158–159
 simulation of 159–162
 small world 169–172
 network kriging *see* kriging
 network tomography *see* tomography,
 network
 node 16
 nonparametric 31
 n -plex *see* plex

O

order 16

P

PageRank algorithm 92
 pairwise Markov property 339
 parametric 31
 Pareto distribution 120
 partition 102
 partitioning, graph 102–111
 hierarchical *see* hierarchical clustering
 spectral 106–110
 path 17

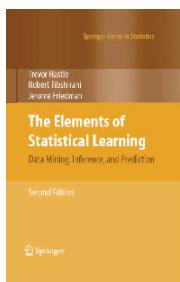
- percolation 276
 phylogenetic inference 225
 hierarchical clustering for 231
 likelihood-based methods for 233–234
 maximum parsimony 228
 planar *see* graph
 plex
 n-plex 95
 potential function 339
 precision 222
 precision matrix 217, 340
 preferential attachment 173–175
 Prim’s algorithm 24
 probability 25
 density (mass) function 25
 marginal 26
 probe, computer packet
 bicast 319
 flexicast 319
 multicast 226
 sandwich 236, 319
 unicast 319
 probit model 256
- Q**
- q*-value 35, 212
- R**
- random graph model *see* network graph model
 random variable 25
 reachable 17
 recall 222
 receiver operating characteristic (ROC) curve 206
 reducible *see* irreducible, graph
 regular *see* graph
 Representer theorem 259
 reproducing kernel Hilbert space 259
 rewiring 161, 169
- S**
- sampling
 network 131–137
 probability proportional to size 128
 unequal probability 127
 with replacement 126
 without replacement 128
 sampling distribution 34
 shortest-path distance *see* distance, between vertices, geodesic
- simple *see* graph
 simulation
 acceptance-rejection 30
 importance sampling 30
 inverse transform 29, 120
 Markov chain Monte Carlo *see* Markov chain Monte Carlo
 Metropolis-Hastings 30, 240
 singular value decomposition 323
 size 16
 small-world model *see* network graph model, small world
 snowball sampling 134–135, 145, 162
 species problem 130, 148–149
 under traceroute sampling 147
- spectral
 bisection *see* bisection
 partitioning *see* partitioning, graph
 standard error 34
 star sampling 133–134
 subgraph 16
 induced 16
 sum-product algorithm 343
 susceptible-infected-removed (SIR) model 272
- T**
- tomography 223
 network 224, 298, 311
 active versus passive 317
 tomogravity method 311
 traceroute sampling 70, 136–137, 147
 trail 17
 transitive triple 96
 transitivity *see* clustering coefficient
 tree 18
 ancestor 19
 children 19
 descendant 19
 directed 18
 leaf 19
 parent 19
 root 19
 triad 79
 census 167
 triadic closure 119
 triangle 96
- V**
- validation 50, 66, 76–77, 197–198, 242
 variance 26
 variational method 343

vertex 16
visualization, graph *see* drawing, graph
W
Wald statistic 34

walk 17
 directed 17
Watts-Strogatz model 169–171, 277
weighted network 194

Springer Series in Statistics

- Sen/Srivastava*: Regression Analysis: Theory, Methods, and Applications.
- Shao*: Mathematical Statistics, 2nd ed.
- Shorack*: Probability for Statisticians
- Shumway/Stoffer*: Time Series Analysis and Its Applications, 2nd ed.
- Simonoff*: Analyzing Categorical Data
- Terrell*: Mathematical Statistics: A Unified Introduction
- Timm*: Applied Multivariate Analysis
- Toutenberg*: Statistical Analysis of Designed Experiments, 2nd ed.
- Wasserman*: All of Nonparametric Statistics
- Wasserman*: All of Statistics: A Concise Course in Statistical Inference
- Weiss*: Modeling Longitudinal Data
- Whittle*: Probability via Expectation, 4th ed.



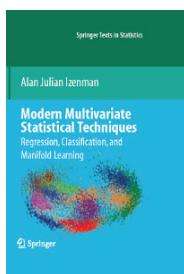
The Elements of Statistical Learning, Second Edition Data Mining, Inference, and Prediction

Trevor Hastie, Robert Tibshirani, Jerome Friedman

This book should be a valuable resource for statisticians and anyone interested in data mining in science or industry. The book's coverage is broad, from supervised learning (prediction) to unsupervised learning. The many topics include neural networks, support vector machines, classification trees and boosting--the first comprehensive treatment of this topic in any book. This major new edition features many topics not covered in the original, including graphical models, random forests, ensemble methods, least angle regression & path algorithms for the lasso, non-negative matrix factorization, and spectral clustering.

2009. 768 pp. (Springer Series in Statistics) Hardcover

ISBN 978-0-84857-0



Modern Multivariate Statistical Techniques Regression, Classification, and Manifold Learning

Alan Julian Izenman

This book is appropriate for advanced undergraduate and graduate students, researchers in statistics, computer science, artificial intelligence, psychology, cognitive sciences, business, medicine, bioinformatics, and engineering. Familiarity with multivariable calculus, linear algebra, and probability and statistics is required. The book presents a carefully-integrated mixture of theory and applications, and of classical and modern multivariate statistical techniques, including Bayesian methods. There are over 60 interesting data sets used as examples in the book, over 200 exercises, and many color illustrations and photographs.

2008. XXVI, 734 pp. (Springer Texts in Statistics) Hardcover

ISBN 978-0-387-78188-4

Statistical Learning from a Regression Perspective

Richard A. Berk

This book considers statistical learning applications when interest centers on the conditional distribution of the response variable, given a set of predictors, and when it is important to characterize how the predictors are related to the response. Among the statistical learning procedures examined are bagging, random forests, boosting, and support vector machines. Intuitive explanations and visual representations are prominent. All of the analyses included are done in R.

2008. 360 pp. (Springer Series in Statistics) Hardcover

ISBN 978-0-387-77500-5

Easy Ways to Order ►

Call: Toll-Free 1-800-SPRINGER • E-mail: orders-ny@springer.com • Write: Springer, Dept. S8113, PO Box 2485, Secaucus, NJ 07096-2485 • Visit: Your local scientific bookstore or urge your librarian to order.