

Springer
Handbooks of
Computational
Statistics

C. Chen
W. Härdle
A. Unwin
(Editors)

Handbook of Data Visualization



Springer

Handbook of Data Visualization

Chun-houh Chen
Wolfgang Härdle
Antony Unwin

Editors

Handbook of Data Visualization

With 569 Figures and 50 Tables



Editors

Dr. Chun-houh Chen

Institute of Statistical Science
Academia Sinica
128 Academia Road, Section 2
Taipei 115
Taiwan
cchen@stat.sinica.edu.tw

Professor Wolfgang Härdle

CASE – Center for Applied Statistics
and Economics
School of Business and Economics
Humboldt-Universität zu Berlin
Spandauer Straße 1
10178 Berlin
Germany
haerdle@wiwi.hu-berlin.de

Professor Antony Unwin

Mathematics Institute
University of Augsburg
86135 Augsburg
Germany
unwin@math.uni-augsburg.de

ISBN 978-3-540-33036-3

e-ISBN 978-3-540-33037-0

DOI 10.1007/978-3-540-33037-0

Library of Congress Control Number: 2007931825

© 2008 Springer-Verlag Berlin Heidelberg

This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilm or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the German Copyright Law of September 9, 1965, in its current version, and permission for use must always be obtained from Springer. Violations are liable for prosecution under the German Copyright Law.

The use of general descriptive names, registered names, trademarks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

Typesetting and Production: LE-T_EX Jelonek, Schmidt & Vöckler GbR, Leipzig, Germany

Cover: deblik, Berlin, Germany

Printed on acid-free paper

9 8 7 6 5 4 3 2 1

springer.com

Table of Contents

I. Data Visualization

I.1 Introduction

Antony Unwin, Chun-houh Chen, Wolfgang K. Härdle 3

II. Principles

II.1 A Brief History of Data Visualization

Michael Friendly 15

II.2 Good Graphics?

Antony Unwin 57

II.3 Static Graphics

Paul Murrell 79

II.4 Data Visualization Through Their Graph Representations

George Michailidis 103

II.5 Graph-theoretic Graphics

Leland Wilkinson 121

II.6 High-dimensional Data Visualization

Martin Theus 151

II.7 Multivariate Data Glyphs: Principles and Practice

Matthew O. Ward 179

II.8 Linked Views for Visual Exploration

Adalbert Wilhelm 199

II.9 Linked Data Views

Graham Wills 217

II.10 Visualizing Trees and Forests

Simon Urbanek 243

III. Methodologies

III.1 Interactive Linked Micromap Plots for the Display of Geographically Referenced Statistical Data	
<i>Jürgen Symanzik, Daniel B. Carr</i>	267
III.2 Grand Tours, Projection Pursuit Guided Tours, and Manual Controls	
<i>Dianne Cook, Andreas Buja, Eun-Kyung Lee, Hadley Wickham</i>	295
III.3 Multidimensional Scaling	
<i>Michael A.A. Cox, Trevor F. Cox</i>	315
III.4 Huge Multidimensional Data Visualization: Back to the Virtue of Principal Coordinates and Dendrograms in the New Computer Age	
<i>Francesco Palumbo, Domenico Vistocco, Alain Morineau</i>	349
III.5 Multivariate Visualization by Density Estimation	
<i>Michael C. Minnotte, Stephan R. Sain, David W. Scott</i>	389
III.6 Structured Sets of Graphs	
<i>Richard M. Heiberger, Burt Holland</i>	415
III.7 Regression by Parts: Fitting Visually Interpretable Models with GUIDE	
<i>Wei-Yin Loh</i>	447
III.8 Structural Adaptive Smoothing by Propagation–Separation Methods	
<i>Jörg Polzehl, Vladimir Spokoiny</i>	471
III.9 Smoothing Techniques for Visualisation	
<i>Adrian W. Bowman</i>	493
III.10 Data Visualization via Kernel Machines	
<i>Yuan-chin Ivan Chang, Yuh-Jye Lee, Hsing-Kuo Pao, Mei-Hsien Lee, Su-Yun Huang</i>	539
III.11 Visualizing Cluster Analysis and Finite Mixture Models	
<i>Friedrich Leisch</i>	561
III.12 Visualizing Contingency Tables	
<i>David Meyer, Achim Zeileis, Kurt Hornik</i>	589
III.13 Mosaic Plots and Their Variants	
<i>Heike Hofmann</i>	617
III.14 Parallel Coordinates: Visualization, Exploration and Classification of High-Dimensional Data	
<i>Alfred Inselberg</i>	643
III.15 Matrix Visualization	
<i>Han-Ming Wu, ShengLi Tzeng, Chun-Houh Chen</i>	681
III.16 Visualization in Bayesian Data Analysis	
<i>Jouni Kerman, Andrew Gelman, Tian Zheng, Yuejing Ding</i>	709
III.17 Programming Statistical Data Visualization in the Java Language	
<i>Junji Nakano, Yoshikazu Yamamoto, Keisuke Honda</i>	725
III.18 Web-Based Statistical Graphics using XML Technologies	
<i>Yoshiro Yamamoto, Masaya Iizuka, Tomokazu Fujino</i>	757

IV. Selected Applications

IV.1 Visualization for Genetic Network Reconstruction

Grace S. Shieh, Chin-Yuan Guo 793

IV.2 Reconstruction, Visualization and Analysis of Medical Images

Henry Horng-Shing Lu 813

IV.3 Exploratory Graphics of a Financial Dataset

Antony Unwin, Martin Theus, Wolfgang K. Härdle 831

IV.4 Graphical Data Representation in Bankruptcy Analysis

Wolfgang K. Härdle, Rouslan A. Moro, Dorothea Schäfer 853

IV.5 Visualizing Functional Data with an Application

to eBay's Online Auctions

Wolfgang Jank, Galit Shmueli, Catherine Plaisant, Ben Shneiderman 873

IV.6 Visualization Tools for Insurance Risk Processes

Krzysztof Burnecki, Rafał Weron 899

List of Contributors

Adrian W. Bowman

University of Glasgow
Department of Statistics
UK
adrian@stats.gla.ac.uk

Chun-houh Chen

Academia Sinica
Institute of Statistical Science
Taiwan
cchen@stat.sinica.edu.tw

Andreas Buja

University of Pennsylvania
Statistics Department
USA
buja@wharton.upenn.edu

Dianne Cook

Iowa State University
Department of Statistics
USA
dicook@iastate.edu

Krzysztof Burnecki

Wroclaw University of Technology
Institute of Mathematics
and Computer Science
Poland
krzysztof.burnecki@gmail.com

Michael A. A. Cox

University of Newcastle Upon Tyne
Division of Psychology
School of Biology and Psychology
UK
mike.cox@ncl.ac.uk

Daniel B. Carr

George Mason University
Center for Computational Statistics
USA
dcarr@gmu.edu

Trevor F. Cox

Data Sciences Unit
Unilever R&D Port Sunlight
UK
trevor.cox@unilever.com

Yuan-chin Ivan Chang

Academia Sinica
Institute of Statistical Science
Taiwan
ycchang@stat.sinica.edu.tw

Yuejing Ding

Columbia University
Department of Statistics
USA
yding@stat.columbia.edu

Michael Friendly

York University
Psychology Department
Canada
friendly@yorku.ca

Burt Holland

Temple University
Department of Statistics
USA
bholland@temple.edu

Tomokazu Fujino

Fukuoka Women's University
Department of Environmental Science
Japan
fujino@fwu.ac.jp

Keisuke Honda

Graduate University
for Advanced Studies
Japan
khonda@ism.ac.jp

Andrew Gelman

Columbia University
Department of Statistics
USA
gelman@stat.columbia.edu

Kurt Hornik

Wirtschaftsuniversität Wien
Department of Statistics
and Mathematics
Austria
Kurt.Hornik@wu-wien.ac.at

Chin-Yuan Guo

Academia Sinica
Institute of Statistical Science
Taiwan

Su-Yun Huang

Academia Sinica
Institute of Statistical Science
Taiwan
syhuang@stat.sinica.edu.tw

Wolfgang K. Härdle

Humboldt-Universität zu Berlin
CASE – Center for Applied Statistics
and Economics
Germany
haerdle@wiwi.hu-berlin.de

Masaya Iizuka

Okayama University
Graduate School of Natural Science
and Technology
Japan
iizuka@ems.okayama-u.ac.jp

Richard M. Heiberger

Temple University
Department of Statistics
USA
rmh@temple.edu

Alfred Inselberg

Tel Aviv University
School of Mathematical Sciences
Israel
aiisreal@post.tau.ac.il

Heike Hofmann

Iowa State University
Department of Statistics
USA
hofmann@iastate.edu

Wolfgang Jank

University of Maryland
Department of Decision
and Information Technologies
USA
wjank@rhsmith.umd.edu

Jouni Kerman

Novartis Pharma AG
USA
jouni.kerman@novartis.com

David Meyer

Wirtschaftsuniversität Wien
Department of Information Systems
and Operations
Austria
David.Meyer@wu-wien.ac.at

Eun-Kyung Lee

Seoul National University
Department of Statistics
Korea
gracesle@snu.ac.kr

George Michailidis

University of Michigan
Department of Statistics
USA
gmichail@umich.edu

Yuh-Jye Lee

National Taiwan University
of Science and Technology
Department of Computer Science
and Information Engineering
Taiwan
yuh-jye@mail.ntust.edu.tw

Michael C. Minnotte

Utah State University
Department of Mathematics
and Statistics
USA
mike.minnotte@usu.edu

Mei-Hsien Lee

National Taiwan University
Institute of Epidemiology
Taiwan

Alain Morineau

La Revue MODULAD
France
alain.morineau@modulad.fr

Friedrich Leisch

Ludwig-Maximilians-Universität
Institut für Statistik
Germany
Friedrich.Leisch@stat.uni-muenchen.de

Rouslan A. Moro

Humboldt-Universität zu Berlin
Institut für Statistik und Ökonometrie
Germany
rmore@diw.de

Wei-Yin Loh

University of Wisconsin-Madison
Department of Statistics
USA
loh@stat.wisc.edu

Paul Murrell

University of Auckland
Department of Statistics
New Zealand
paul@stat.auckland.ac.nz

Henry Horng-Shing Lu

National Chiao Tung University
Institute of Statistics
Taiwan
hslu@stat.nctu.edu.tw

Junji Nakano

The Institute of Statistical Mathematics
and the Graduate University
for Advanced Studies
Japan
nakanoj@ism.ac.jp

Francesco Palumbo

University of Macerata
Dipartimento di Istituzioni
Economiche e Finanziarie
Italy
palumbo@unimc.it

Hsing-Kuo Pao

National Taiwan University
of Science and Technology
Department of Computer Science
and Information Engineering
Taiwan
pao@mail.ntust.edu.tw

Catherine Plaisant

University of Maryland
Department of Computer Science
USA
plaisant@cs.umd.edu

Jörg Polzehl

Weierstrass Institute
for Applied Analysis and Stochastics
Germany
polzehl@wias-berlin.de

Stephan R. Sain

University of Colorado at Denver
Department of Mathematics
USA
ssain@math.cudenver.edu

Dorothea Schäfer

Wirtschaftsforschung (DIW) Berlin
German Institute for Economic Research
Germany
dschaefer@diw.de

David W. Scott

Rice University
Division Statistics
USA
scottdw@rice.edu

Grace Shwu-Rong Shieh

Academia Sinica
Institute of Statistical Science
Taiwan
gshieh@stat.sinica.edu.tw

Galit Shmueli

University of Maryland
Department of Decision
and Information Technologies
USA
gshmueli@rhsmith.umd.edu

Ben Schneiderman

University of Maryland
Department of Computer Science
USA
ben@cs.umd.edu

Vladimir Spokoiny

Weierstrass Institute
for Applied Analysis and Stochastics
Germany
spokoiny@wias-berlin.de

Jürgen Symanzik

Utah State University
Department of Mathematics
and Statistics
USA
symanzik@math.usu.edu

Martin Theus

University of Augsburg
Department of Computational Statistics
and Data Analysis
Germany
martin.theus@math.uni-augsburg.de

ShengLi Tzeng

Academia Sinica
Institute of Statistical Science
Taiwan
h0hl@stat.sinica.edu.tw

Antony Unwin

Mathematics Institute
University of Augsburg
Germany
unwin@math.uni-augsburg.de

Leland Wilkinson

SYSTAT Software Inc. Chicago
USA
leland.wilkinson@systat.com

Simon Urbanek

AT&T Labs – Research
USA
urbanek@research.att.com

Graham Wills

SPSS Inc. Chicago
USA
gwills@spss.com

Domenico Vistocco

University of Cassino
Dipartimento di Economia e Territorio
Italy
vistocco@unicas.it

Han-Ming Wu

Academia Sinica
Institute of Statistical Science
Taiwan
hmwu@stat.sinica.edu.tw

Matthew O. Ward

Worcester Polytechnic Institute
Computer Science Department
USA
matt@cs.wpi.edu

Yoshikazu Yamamoto

Tokushima Bunri University
Department of Engineering
Japan
yamamoto@es.bunri-u.ac.jp

Rafał Weron

Wrocław University of Technology
Institute of Mathematics
and Computer Science
Poland
rafal.weron@im.pwr.wroc.pl

Yoshiro Yamamoto

Tokai University
Department of Mathematics
Japan
yamamoto@sm.u-tokai.ac.jp

Hadley Wickham

Iowa State University
Department of Statistics
USA
hadley@iastate.edu

Achim Zeileis

Wirtschaftsuniversität Wien
Department of Statistics
and Mathematics
Austria
Achim.Zeileis@wu-wien.ac.at

Adalbert Wilhelm

International University Bremen
Germany
a.wilhelm@iu-bremen.de

Tian Zheng

Columbia University
Department of Statistics
USA
tzheng@stat.columbia.edu

Part I

Data Visualization

Introduction

I.1

Antony Unwin, Chun-houh Chen, Wolfgang K. Härdle

1.1	<i>Computational Statistics and Data Visualization</i>	4
	Data Visualization and Theory	4
	Presentation and Exploratory Graphics	4
	Graphics and Computing	5
1.2	<i>The Chapters</i>	6
	Summary and Overview; Part II.....	7
	Summary and Overview; Part III.....	9
	Summary and Overview; Part IV	10
	The Authors.....	11
1.3	<i>Outlook</i>	12

1.1

Computational Statistics and Data Visualization

This book is the third volume of the Handbook of Computational Statistics and covers the field of data visualization. In line with the companion volumes, it contains a collection of chapters by experts in the field to present readers with an up-to-date and comprehensive overview of the state of the art. Data visualization is an active area of application and research, and this is a good time to gather together a summary of current knowledge.

Graphic displays are often very effective at communicating information. They are also very often not effective at communicating information. Two important reasons for this state of affairs are that graphics can be produced with a few clicks of the mouse without any thought and the design of graphics is not taken seriously in many scientific textbooks. Some people seem to think that preparing good graphics is just a matter of common sense (in which case their common sense cannot be in good shape), while others believe that preparing graphics is a low-level task, not appropriate for scientific attention. This volume of the Handbook of Computational Statistics takes graphics for data visualization seriously.

1.1.1 Data Visualization and Theory

Graphics provide an excellent approach for exploring data and are essential for presenting results. Although graphics have been used extensively in statistics for a long time, there is not a substantive body of theory about the topic. Quite a lot of attention has been paid to graphics for presentation, particularly since the superb books of Edward Tufte. However, this knowledge is expressed in principles to be followed and not in formal theories. Bertin's work from the 1960s is often cited but has not been developed further. This is a curious state of affairs. Graphics are used a great deal in many different fields, and one might expect more progress to have been made along theoretical lines.

Sometimes in science the theoretical literature for a subject is considerable while there is little applied literature to be found. The literature on data visualization is very much the opposite. Examples abound in almost every issue of every scientific journal concerned with quantitative analysis. There are occasionally articles published in a more theoretical vein about specific graphical forms, but little else. Although there is a respected statistics journal called the Journal of Computational and Graphical Statistics, most of the papers submitted there are in computational statistics. Perhaps this is because it is easier to publish a study of a technical computational problem than it is to publish work on improving a graphic display.

1.1.2 Presentation and Exploratory Graphics

The differences between graphics for presentation and graphics for exploration lie in both form and practice. Presentation graphics are generally static, and a single

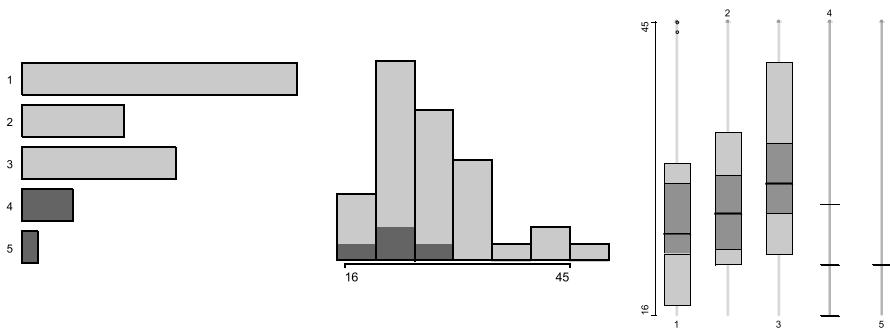


Figure 1.1. A barchart of the number of authors per paper, a histogram of the number of pages per paper, and parallel boxplots of length by number of authors. Papers with more than three authors have been selected

graphic is drawn to summarize the information to be presented. These displays should be of high quality and include complete definitions and explanations of the variables shown and of the form of the graphic. Presentation graphics are like proofs of mathematical theorems; they may give no hint as to how a result was reached, but they should offer convincing support for its conclusion. Exploratory graphics, on the other hand, are used for looking for results. Very many of them may be used, and they should be fast and informative rather than slow and precise. They are not intended for presentation, so that detailed legends and captions are unnecessary. One presentation graphic will be drawn for viewing by potentially thousands of readers while thousands of exploratory graphics may be drawn to support the data investigations of one analyst.

Books on visualization should make use of graphics. Figure 1.1 shows some simple summaries of data about the chapters in this volume, revealing that over half the chapters had more than one author and that more authors does not always mean longer papers.

Graphics and Computing

1.1.3

Developments in computing power have been of great benefit to graphics in recent years. It has become possible to draw precise, complex displays with great ease and to print them with impressive quality at high resolution. That was not always the case, and initially computers were more a disadvantage for graphics. Computing screens and printers could at best produce clumsy line-driven displays of low resolution without colour. These offered no competition to careful, hand-drawn displays. Furthermore, even early computers made many calculations much easier than before and allowed fitting of more complicated models. This directed attention away from graphics, and it is only in the last 20 years that graphics have come into their own again.

These comments relate to presentation graphics, that is, graphics drawn for the purpose of illustrating and explaining results. Computing advances have benefitted exploratory graphics, that is, graphics drawn to support exploring data, far more. Not just the quality of graphic representation has improved but also the quantity. It is now trivial to draw many different displays of the same data or to riffle through many different versions interactively to look for information in data. These capabilities are only gradually becoming appreciated and capitalized on.

The importance of software availability and popularity in determining what analyses are carried out and how they are presented will be an interesting research topic for future historians of science. In the business world, no one seems to be able to do without the spreadsheet Excel. If Excel does not offer a particular graphic form, then that form will not be used. (In fact Excel offers many graphic forms, though not all that a statistician would want.) Many scientists, who only rarely need access to computational power, also rely on Excel and its options. In the world of statistics itself, the packages SAS and SPSS were long dominant. In the last 15 years, first S and S-plus and now R have emerged as important competitors. None of these packages currently provide effective interactive tools for exploratory graphics, though they are all moving slowly in that direction as well as extending the range and flexibility of the presentation graphics they offer.

Data visualization is a new term. It expresses the idea that it involves more than just representing data in a graphical form (instead of using a table). The information behind the data should also be revealed in a good display; the graphic should aid readers or viewers in seeing the structure in the data. The term data visualization is related to the new field of information visualization. This includes visualization of all kinds of information, not just of data, and is closely associated with research by computer scientists. Up till now the work in this area has tended to concentrate just on presenting information, rather than on what may be deduced from it. Statisticians tend to be concerned more with variability and to emphasize the statistical properties of results. The closer linking of graphics with statistical modelling can make this more explicit and is a promising research direction that is facilitated by the flexible nature of current computing software. Statisticians have an important role to play here.

1.2

The Chapters

Needless to say, each Handbook chapter uses a lot of graphic displays. Figure 1.2 is a scatterplot of the number of figures against the number of pages. There is an approximate linear relationship with a couple of papers having somewhat more figures per page and one somewhat less. The scales have been chosen to maximize the data-ink ratio. An alternative version with equal scales makes clearer that the number of figures per page is almost always less than one.

The Handbook has been divided into three sections: Principles, Methodology, and Applications. Needless to say, the sections overlap. Figure 1.3 is a binary matrix visualization using Jaccard coefficients for both chapters (rows) and index entries

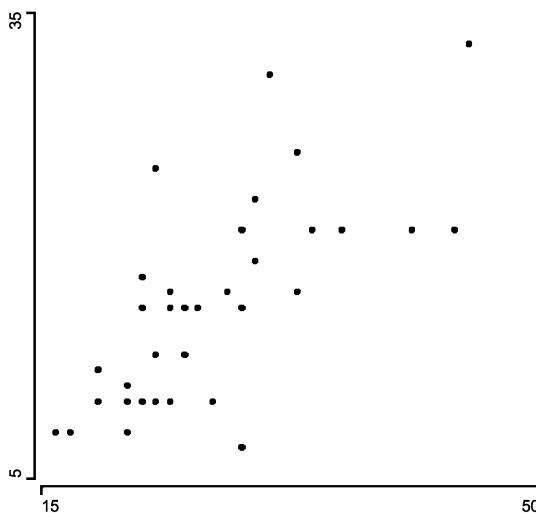


Figure 1.2. A scatterplot of the number of figures against the number of pages for the Handbook's chapters

(columns) to explore links between chapters. In the raw data map (lower-left portion of Fig. 1.3) there is a banding of black dots from the lower-left to upper-right corners indicating a possible transition of chapter/index combinations. In the proximity map of indices (upper portion of Fig. 1.3), index groups A, B, C, D, and E are overlapped with each other and are dominated by chapters of Good Graphics, History, Functional Data, Matrix Visualization, and Regression by Parts respectively.

Summary and Overview; Part II

1.2.1

The ten chapters in Part II are concerned with principles of data visualization. First there is an historical overview by Michael Friendly, the custodian of the Internet Gallery of Data Visualization, outlining the developments in graphical displays over the last few hundred years and including many fine examples.

In the next chapter Antony Unwin discusses some of the guidelines for the preparation of sound and attractive data graphics. The question mark in the chapter title sums it up well: whatever principles or recommendations are followed, the success of a graphic is a matter of taste; there are no fixed rules.

The importance of software for producing graphics is incontrovertible. Paul Murrell in his chapter summarizes the requirements for producing accurate and exact static graphics. He emphasizes both the need for flexibility in customizing standard plots and the need for tools that permit the drawing of new plot types.

Structure in data may be represented by mathematical graphs. George Michailidis pursues this idea in his chapter and shows how this leads to another class of graphic displays associated with multivariate analysis methods.

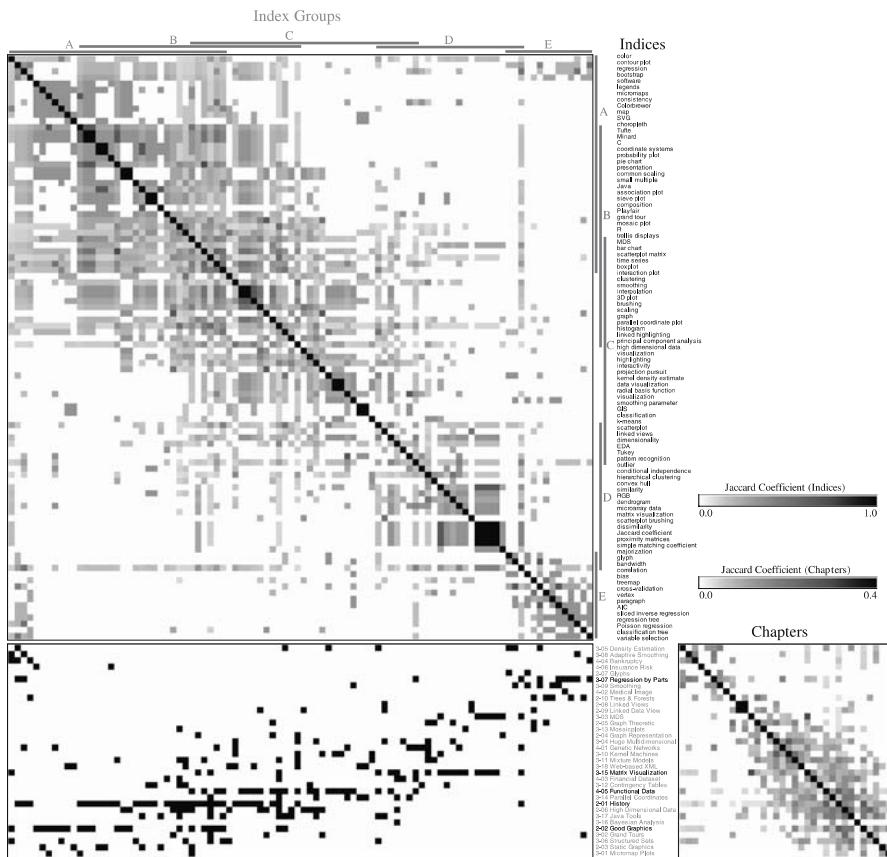


Figure 1.3. Matrix visualizations of the Handbook with chapters in the rows and index entries in the columns

Lee Wilkinson approaches graph-theoretic visualizations from another point of view, and his displays are concerned predominantly, though by no means exclusively, with trees, directed graphs and geometric graphs. He also covers the layout of graphs, a tricky problem for large numbers of vertices, and raises the intriguing issue of graph matching.

Most data displays concentrate on one or two dimensions. This is frequently sufficient to reveal striking information about a dataset. To gain insight into multivariate structure, higher-dimensional representations are required. Martin Theus discusses the main statistical graphics of this kind that do not involve dimension reduction and compares their possible range of application.

Everyone knows about Chernoff faces, though not many ever use them. The potential of data glyphs for representing cases in informative and productive ways has not been fully realized. Matt Ward gives an overview of the wide variety of possible forms and of the different ways they can be utilized.

There are two chapters on linking. Adalbert Wilhelm describes a formal model for linked graphics and the conceptual structure underlying it. He is able to encompass different types of linking and different representations. Graham Wills looks at linking in a more applied context and stresses the importance of distinguishing between views of individual cases and aggregated views. He also highlights the variety of selection possibilities there are in interactive graphics. Both chapters point out the value of linking simple data views over linking complicated ones.

The final chapter in this section is by Simon Urbanek. He describes the graphics that have been introduced to support tree models in statistics. The close association between graphics and the models (and collections of models in forests) is particularly interesting and has relevance for building closer links between graphics and models in other fields.

Summary and Overview; Part III

1.2.2

The middle and largest section of the Handbook concentrates on individual area of graphics research.

Geographical data can obviously benefit from visualization. Much of Bertin's work was directed at this kind of data. Juergen Symanzik and Daniel Carr write about micromaps (multiple small images of the same area displaying different parts of the data) and their interactive extension.

Projection pursuit and the grand tour are well known but not easy to use. Despite the availability of attractive free software, it is still a difficult task to analyse datasets in depth with this approach. Dianne Cook, Andreas Buja, Eun-Kyung Lee and Hadley Wickham describe the issues involved and outline some of the progress that has been made.

Multidimensional scaling has been around for a long time. Michael Cox and Trevor Cox (no relation, but an MDS would doubtless place them close together) review the current state of research.

Advances in high-throughput techniques in industrial projects, academic studies and biomedical experiments and the increasing power of computers for data collection have inevitably changed the practice of modern data analysis. Real-life datasets become larger and larger in both sample size and numbers of variables. Francesco Palumbo, Alain Morineau and Domenico Vistocco illustrate principles of visualization for such situations.

Some areas of statistics benefit more directly from visualization than others. Density estimation is hard to imagine without visualization. Michael Minnotte, Steve Sain and David Scott examine estimation methods in up to three dimensions. Interestingly there has not been much progress with density estimation in even three dimensions.

Sets of graphs can be particularly useful for revealing the structure in datasets and complement modelling efforts. Richard Heiberger and Burt Holland describe an approach primarily making use of Cartesian products and the Trellis paradigm. Wei-Yin Loh describes the use of visualization to support the use of regression models, in particular with the use of regression trees.

Instead of visualizing the structure of samples or variables in a given dataset, researchers may be interested in visualizing images collected with certain formats. Usually the target images are collected with various types of noise pattern and it is necessary to apply statistical or mathematical modelling to remove or diminish the noise structure before the possible genuine images can be visualized. Jörg Polzehl and Vladimir Spokoiny present one such novel adaptive smoothing procedure in reconstructing noisy images for better visualization.

The continuing increase in computer power has had many different impacts on statistics. Computationally intensive smoothing methods are now commonplace, although they were impossible only a few years ago. Adrian Bowman gives an overview of the relations between smoothing and visualization. Yuan-chin Chang, Yuh-Jye Lee, Hsing-Kuo Pao, Mei-Hsien Lee and Su-Yun Huang investigate the impact of kernel machine methods on a number of classical techniques: principal components, canonical correlation and cluster analysis. They use visualizations to compare their results with those from the original methods.

Cluster analyses have often been a bit suspect to statisticians. The lack of formal models in the past and the difficulty of judging the success of the clusterings were both negative factors. Fritz Leisch considers the graphical evaluation of clusterings and some of the possibilities for a sounder methodological approach.

Multivariate categorical data were difficult to visualize in the past. The chapter by David Meyer, Achim Zeileis and Kurt Hornik describes fairly classical approaches for low dimensions and emphasizes the link to model building. Heike Hofmann describes the powerful tools of interactive mosaicplots that have become available in recent years, not least through her own efforts, and discusses how different variations of the plot form can be used for gaining insight into multivariate data features.

Alfred Inselberg, the original proposer of parallel coordinate plots, offers an overview of this approach to multivariate data in his usual distinctive style. Here he considers in particular classification problems and how parallel coordinate views can be adapted and amended to support this kind of analysis.

Most analyses using graphics make use of a standard set of graphical tools, for example, scatterplots, barcharts, and histograms. Han-Ming Wu, ShengLi Tzeng and Chun-hou Chen describe a different approach, built around using colour approximations for individual values in a data matrix and applying cluster analyses to order the matrix rows and columns in informative ways.

For many years Bayesians were primarily theoreticians. Thanks to MCMC methods they are now able to also apply their ideas to great effect. This has led to new demands in assessing model fit and the quality of the results. Jouni Kerman, Andrew Gelman, Tian Zheng and Yuejing Ding discuss graphical approaches for tackling these issues in a Bayesian framework.

Without software to draw the displays, graphic analysis is almost impossible nowadays. Junji Nakano, Yamamoto Yoshikazu and Keisuke Honda are working on Java-based software to provide support for new developments, and they outline their approach here. Many researchers are interested in providing tools via the Web. Yoshiro Yamamoto, Masaya Iizuka and Tomokazu Fujino discuss using XML for interactive statistical graphics and explain the issues involved.

Summary and Overview; Part IV

1.2.3

The final section contains seven chapters on specific applications of data visualization. There are, of course, individual applications discussed in earlier chapters, but here the emphasis is on the application rather than principles or methodology.

Genetic networks are obviously a promising area for informative graphic displays. Grace Shieh and Chin-Yuan Guo describe some of the progress made so far and make clear the potential for further research.

Modern medical imaging systems have made significant contributions to diagnoses and treatments. Henry Lu discusses the visualization of data from positron emission tomography, ultrasound and magnetic resonance.

Two chapters examine company bankruptcy datasets. In the first one, Antony Unwin, Martin Theus and Wolfgang Härdle use a broad range of visualization tools to carry out an extensive exploratory data analysis. No large dataset can be analysed cold, and this chapter shows how effective data visualization can be in assessing data quality and revealing features of a dataset. The other bankruptcy chapter employs graphics to visualize SVM modelling. Wolfgang Härdle, Rouslan Moro and Dorothea Schäfer use graphics to display results that cannot be presented in a closed analytic form.

The astonishing growth of eBay has been one of the big success stories of recent years. Wolfgang Jank, Galit Shmueli, Catherine Plaisant and Ben Shneiderman have studied data from eBay auctions and describe the role graphics played in their analyses.

Krzysztof Burnecki and Rafal Weron consider the application of visualization in insurance. This is another example of how the value of graphics lies in providing insight into the output of complex models.

The Authors

1.2.4

The editors would like to thank the authors of the chapters for their contributions. It is important for a collective work of this kind to cover a broad range and to gather many experts with different interests together. We have been fortunate in receiving the assistance of so many excellent contributors.

The mixture at the end remains, of course, a mixture. Different authors take different approaches and have different styles. It early became apparent that even the term data visualization means different things to different people! We hope that the Handbook gains rather than loses by this eclecticism.

Figures 1.1 and 1.2 earlier in the chapter showed that the chapter form varied between authors in various ways. Figure 1.4 reveals another aspect. The scatterplot shows an outlier with a very large number of references (the historical survey of Michael Friendly) and that some papers referenced the work of their own authors more than others. The histogram is for the rate of self-referencing.

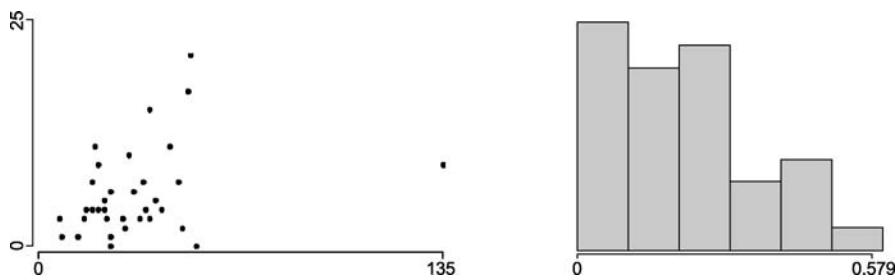


Figure 1.4. A scatterplot of the number of references to papers by a chapter's authors against the total number of references and a histogram of the rate of self-referencing

1.3

Outlook

There are many open issues in data visualization and many challenging research problems. The datasets to be analysed tend to be more complex and are certainly becoming larger all the time. The potential of graphical tools for exploratory data analysis has not been fully realized, and the complementary interplay between statistical modelling and graphics has not yet been fully exploited. Advances in computer software and hardware have made producing graphics easier, but they have also contributed to raising the standards expected.

Future developments will undoubtedly include more flexible and powerful software and better integration of modelling and graphics. There will probably be individual new and innovative graphics and some improvements in the general design of displays. Gradual gains in knowledge about the perception of graphics and the psychological aspects of visualization will lead to improved effectiveness of graphic displays. Ideally there should be progress in the formal theory of data visualization, but that is perhaps the biggest challenge of all.

Part II

Principles

A Brief History of Data Visualization

Michael Friendly

1.1	<i>Introduction</i>	16
1.2	<i>Milestones Tour</i>	17
	Pre-17th Century: Early Maps and Diagrams.....	17
	1600–1699: Measurement and Theory	19
	1700–1799: New Graphic Forms	22
	1800–1850: Beginnings of Modern Graphics	25
	1850–1900: The Golden Age of Statistical Graphics.....	28
	1900–1950: The Modern Dark Ages	37
	1950–1975: Rebirth of Data Visualization	39
	1975–present: High-D, Interactive and Dynamic Data Visualization	40
1.3	<i>Statistical Historiography</i>	42
	History as ‘Data’	42
	Analysing Milestones Data	43
	What Was He Thinking? – Understanding Through Reproduction.....	45
1.4	<i>Final Thoughts</i>	48

It is common to think of statistical graphics and data visualization as relatively modern developments in statistics. In fact, the graphic representation of quantitative information has deep roots. These roots reach into the histories of the earliest map making and visual depiction, and later into thematic cartography, statistics and statistical graphics, medicine and other fields. Along the way, developments in technologies (printing, reproduction), mathematical theory and practice, and empirical observation and recording enabled the wider use of graphics and new advances in form and content.

This chapter provides an overview of the intellectual history of data visualization from medieval to modern times, describing and illustrating some significant advances along the way. It is based on a project, called the *Milestones Project*, to collect, catalogue and document in one place the important developments in a wide range of areas and fields that led to modern data visualization. This effort has suggested some questions concerning the use of present-day methods of analysing and understanding this history, which I discuss under the rubric of ‘statistical historiography’.

1.1

Introduction

The only new thing in the world is the history you don't know. – Harry S Truman

It is common to think of statistical graphics and data visualization as relatively modern developments in statistics. In fact, the graphic portrayal of quantitative information has deep roots. These roots reach into the histories of the earliest map-making and visual depiction, and later into thematic cartography, statistics and statistical graphics, with applications and innovations in many fields of medicine and science which are often intertwined with each other. They also connect with the rise of statistical thinking and widespread data collection for planning and commerce up through the 19th century. Along the way, a variety of advancements contributed to the widespread use of data visualization today. These include technologies for drawing and reproducing images, advances in mathematics and statistics, and new developments in data collection, empirical observation and recording.

From above ground, we can see the current fruit and anticipate future growth; we must look below to understand their germination. Yet the great variety of roots and nutrients across these domains, which gave rise to the many branches we see today, are often not well known and have never been assembled in a single garden to be studied or admired.

This chapter provides an overview of the intellectual history of data visualization from medieval to modern times, describing and illustrating some significant advances along the way. It is based on what I call the Milestones Project, an attempt to provide a broadly comprehensive and representative catalogue of important developments in *all* fields related to the history of data visualization.

There are many historical accounts of developments within the fields of probability (Hald, 1990), statistics (Pearson, 1978; Porter, 1986; Stigler, 1986), astronomy (Riddell, 1980) and cartography (Wallis and Robinson, 1987), which relate to, *inter alia*, some of the important developments contributing to modern data visualization. There are other, more specialized, accounts which focus on the early history of graphic recording (Hoff and Geddes, 1959, 1962), statistical graphs (Funkhouser, 1936, 1937; Royston, 1970; Tilling, 1975), fitting equations to empirical data (Farebrother, 1999), economics and time-series graphs (Klein, 1997), cartography (Friis, 1974; Kruskal, 1977) and thematic mapping (Robinson, 1982; Palsky, 1996) and so forth; Robinson (Robinson, 1982, Chap. 2) presents an excellent overview of some of the important scientific, intellectual and technical developments of the 15th–18th centuries leading to thematic cartography and statistical thinking. Wainer and Velleman (2001) provide a recent account of some of the history of statistical graphics.

But there are no accounts which span the entire development of visual thinking and the visual representation of data and which collate the contributions of disparate disciplines. Inasmuch as their histories are intertwined, so too should be any telling of the development of data visualization. Another reason for interweaving these accounts is that practitioners in these fields today tend to be highly specialized and unaware of related developments in areas outside their domain, much less of their history.

Milestones Tour

1.2

Every picture tells a story.

– Rod Stewart, 1971

In organizing this history, it proved useful to divide history into epochs, each of which turned out to be describable by coherent themes and labels. This division is, of course, somewhat artificial, but it provides the opportunity to characterize the accomplishments in each period in a general way before describing some of them in more detail. Figure 1.1, discussed in Sect. 1.3.2, provides a graphic overview of the epochs I describe in the subsections below, showing the frequency of events considered milestones in the periods of this history. For now, it suffices to note the labels attached to these epochs, a steady rise from the early 18th century to the late 19th century, with a curious wiggle thereafter.

In the larger picture – recounting the history of data visualization – it turns out that many of the milestone items have a story to be told: What motivated this development? What was the communication goal? How does it relate to other developments – What were the precursors? How has this idea been used or re-invented today? Each section below tries to illustrate the general themes with a few exemplars. In particular, this account attempts to tell a few representative stories of these periods, rather than to try to be comprehensive.

For reasons of economy, only a limited number of images could be printed here, and these only in black and white. Others are referred to by Web links, mostly from

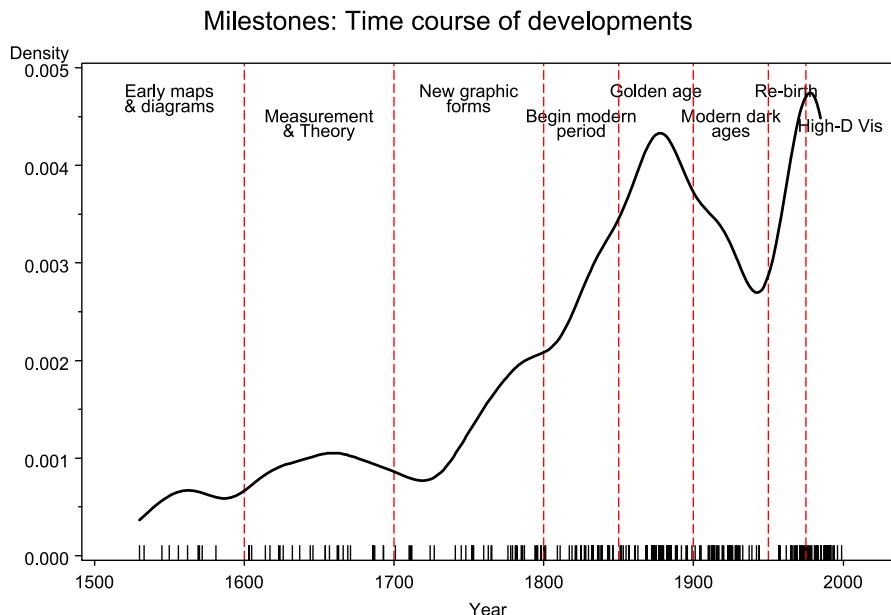


Figure 1.1. Time distribution of events considered milestones in the history of data visualization, shown by a rug plot and density estimate

the Milestones Project, <http://www.math.yorku.ca/SCS/Gallery/milestone/>, where a colour version of this chapter will also be found.

1.2.1 Pre-17th Century: Early Maps and Diagrams

The earliest seeds of visualization arose in geometric diagrams, in tables of the positions of stars and other celestial bodies, and in the making of maps to aid in navigation and exploration. The idea of coordinates was used by ancient Egyptian surveyors in laying out towns, earthly and heavenly positions were located by something akin to latitude and longitude by at least 200 B.C., and the map projection of a spherical earth into latitude and longitude by Claudius Ptolemy [c. 85–c. 165] in Alexandria would serve as reference standards until the 14th century.

Among the earliest graphical depictions of quantitative information is an anonymous 10th-century multiple time-series graph of the changing position of the seven most prominent heavenly bodies over space and time (Fig. 1.2), described by Funkhouser (1936) and reproduced in Tufte (1983, p. 28). The vertical axis represents the inclination of the planetary orbits; the horizontal axis shows time, divided into 30 intervals. The sinusoidal variation with different periods is notable, as is the use of a grid, suggesting both an implicit notion of a coordinate system and something akin to graph paper, ideas that would not be fully developed until the 1600–1700s.

In the 14th century, the idea of plotting a theoretical function (as a proto bar graph) and the logical relation between tabulating values and plotting them appeared in

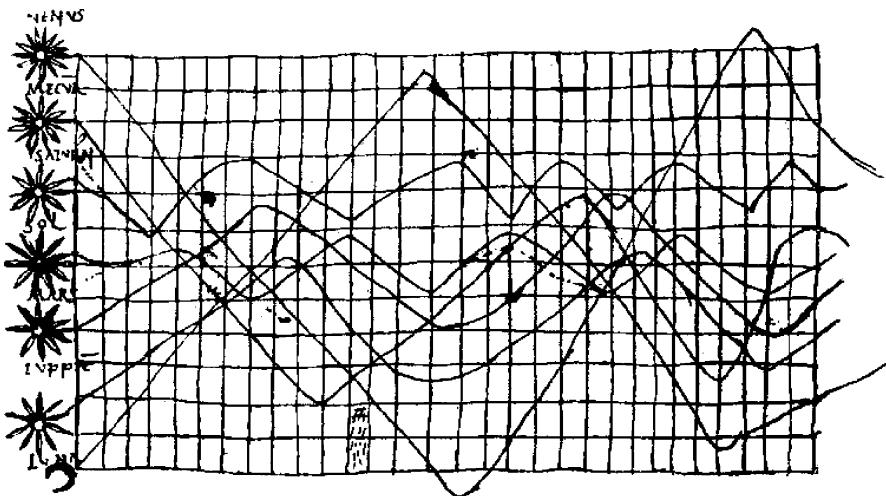


Figure 1.2. Planetary movements shown as cyclic inclinations over time, by an unknown astronomer, appearing in a 10th-century appendix to commentaries by A.T. Macrobius on Cicero's *In Somnium Scipionis*. Source: Funkhouser (1936, p. 261)

a work by Nicole Oresme [1323–1382] Bishop of Liseus¹ (Oresme, 1482, 1968), followed somewhat later by the idea of a theoretical graph of distance vs. speed by Nicolas of Cusa.

By the 16th century, techniques and instruments for precise observation and measurement of physical quantities and geographic and celestial position were well developed (for example, a ‘wall quadrant’ constructed by Tycho Brahe [1546–1601], covering an entire wall in his observatory). Particularly important were the development of triangulation and other methods to determine mapping locations accurately (Frisius, 1533; Tartaglia, 1556). As well, we see initial ideas for capturing images directly (the camera obscura, used by Reginer Gemma-Frisius in 1545 to record an eclipse of the sun), the recording of mathematical functions in tables (trigonometric tables by Georg Rheticus, 1550) and the first modern cartographic atlas (*Theatrum Orbis Terrarum* by Abraham Ortelius, 1570). These early steps comprise the beginnings of data visualization.

1600–1699: Measurement and Theory

1.2.2

Amongst the most important problems of the 17th century were those concerned with physical measurement – of time, distance and space – for astronomy, survey-

¹ Funkhouser (1936, p. 277) was sufficiently impressed with Oresme’s grasp of the relation between functions and graphs that he remarked, ‘If a pioneering contemporary had collected some data and presented Oresme with actual figures to work upon, we might have had statistical graphs four hundred years before Playfair.’

ing, map making, navigation and territorial expansion. This century also saw great new growth in theory and the dawn of practical application – the rise of analytic geometry and coordinate systems (Descartes and Fermat), theories of errors of measurement and estimation (initial steps by Galileo in the analysis of observations on Tycho Brahe's star of 1572 (Hald, 1990, §10.3)), the birth of probability theory (Pascal and Fermat) and the beginnings of demographic statistics (John Graunt) and 'political arithmetic' (William Petty) – the study of population, land, taxes, value of goods, etc. for the purpose of understanding the wealth of the state.

Early in this century, Christopher Scheiner (1626–1630, recordings from 1611) introduced an idea Tufte (1983) would later call the principle of 'small multiples' to show the changing configurations of sunspots over time, shown in Fig. 1.3. The multiple images depict the recordings of sunspots from 23 October 1611 until 19 December of that year. The large key in the upper left identifies seven groups of sunspots by the letters A–G. These groups are similarly identified in the 37 smaller images, arrayed left to right and top to bottom below.

Another noteworthy example (Fig. 1.4) shows a 1644 graphic by Michael Florent van Langren [1600–1675], a Flemish astronomer to the court of Spain, believed to be the first visual representation of statistical data (Tufte, 1997, p. 15). At that time, lack of

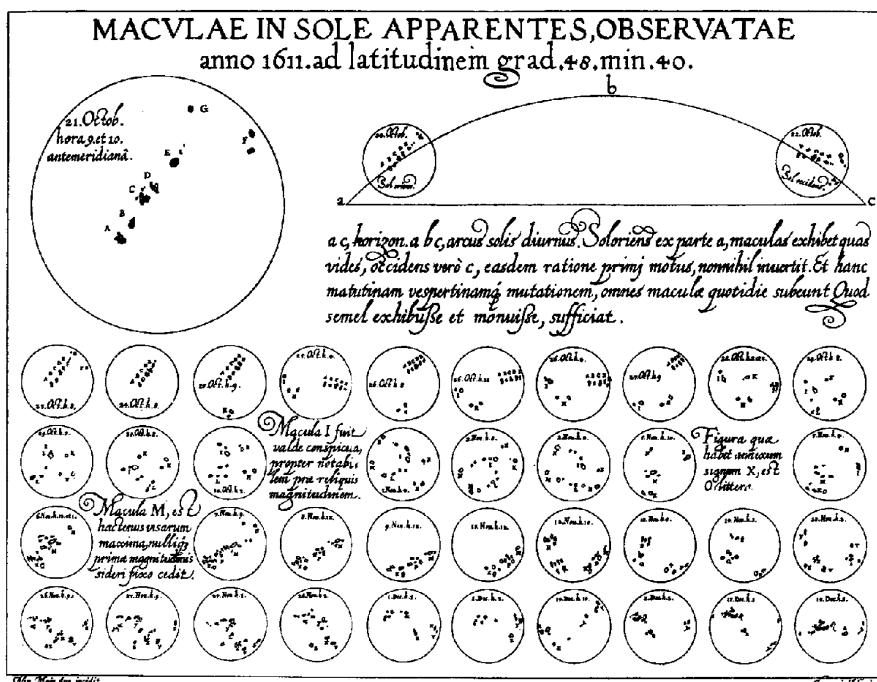


Figure 1.3. Scheiner's 1626 representation of the changes in sunspots over time. Source: Scheiner (1626–1630)

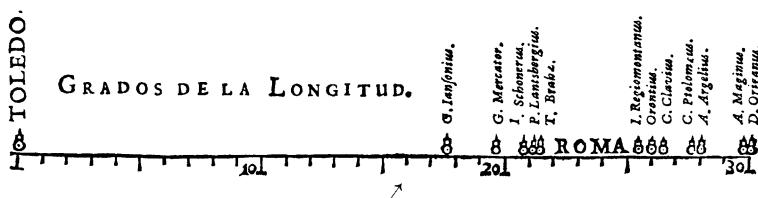


Figure 1.4. Langren's 1644 graph of determinations of the distance, in longitude, from Toledo to Rome. The correct distance is $16^{\circ}30'$. Source: Tufte (1997, p. 15)

a reliable means to determine longitude at sea hindered navigation and exploration.² This 1-D line graph shows all 12 known estimates of the difference in longitude between Toledo and Rome and the name of the astronomer (Mercator, Tycho Brahe, Ptolemy, etc.) who provided each observation.

What is notable is that van Langren could have presented this information in various tables – ordered by author to show provenance, by date to show priority, or by distance. However, only a graph shows the wide variation in the estimates; note that the range of values covers nearly half the length of the scale. Van Langren took as his overall summary the centre of the range, where there happened to be a large enough gap for him to inscribe 'ROMA.' Unfortunately, all of the estimates were biased upwards; the true distance ($16^{\circ}30'$) is shown by the arrow. Van Langren's graph is also a milestone as the earliest known exemplar of the principle of 'effect ordering for data display' (Friendly and Kwan, 2003).

In the 1660s, the systematic collection and study of social data began in various European countries, under the rubric of 'political arithmetic' (John Graunt, 1662 and William Petty, 1665), with the goals of informing the state about matters related to wealth, population, agricultural land, taxes and so forth,³ as well as for commercial purposes such as insurance and annuities based on life tables (Jan de Witt, 1671). At approximately the same time, the initial statements of probability theory around 1654 (see Ball, 1908) together with the idea of coordinate systems were applied by Christiaan Huygens in 1669 to give the first graph of a continuous distribution function⁴ (from Graunt's based on the bills of mortality). The mid-1680s saw the first bivariate plot derived from empirical data, a theoretical curve relating barometric pressure to altitude, and the first known weather map,⁵ showing prevailing winds on a map of the earth (Halley, 1686).

By the end of this century, the necessary elements for the development of graphical methods were at hand – some real data of significant interest, some theory to make

² For navigation, latitude could be fixed from star inclinations, but longitude required accurate measurement of time at sea, an unsolved problem until 1765 with the invention of a marine chronometer by John Harrison. See Sobel (1996) for a popular account.

³ For example, Graunt (1662) used his tabulations of London births and deaths from parish records and the bills of mortality to estimate the number of men the king would find available in the event of war (Klein, 1997, pp. 43–47).

⁴ Image: <http://math.yorku.ca/SCS/Gallery/images/huygens-graph.gif>

⁵ Image: <http://math.yorku.ca/SCS/Gallery/images/halleyweathermap-1686.jpg>

sense of them, and a few ideas for their visual representation. Perhaps more importantly, one can see this century as giving rise to the beginnings of visual thinking, as illustrated by the examples of Scheiner and van Langren.

1.2.3

1700–1799: New Graphic Forms

With some rudiments of statistical theory, data of interest and importance, and the idea of graphic representation at least somewhat established, the 18th century witnessed the expansion of these aspects to new domains and new graphic forms. In cartography, mapmakers began to try to show more than just geographical position on a map. As a result, new data representations (isolines and contours) were invented, and thematic mapping of physical quantities took root. Towards the end of this century, we see the first attempts at the thematic mapping of geologic, economic and medical data.

Abstract graphs, and graphs of functions became more widespread, along with the early stirrings of statistical theory (measurement error) and systematic collection of empirical data. As other (economic and political) data began to be collected, some novel visual forms were invented to portray them, so the data could ‘speak to the eyes.’

For example, the use of isolines to show contours of equal value on a coordinate grid (maps and charts) was developed by Edmund Halley (1701). Figure 1.5, showing isogons – lines of equal magnetic declination – is among the first examples of thematic cartography, overlaying data on a map. Contour maps and topographic maps were introduced somewhat later by Philippe Buache (1752) and Marcellin du Carla-Boniface (1782).

Timelines, or ‘cartes chronologiques,’ were first introduced by Jacques Barbeu-Dubourg in the form of an annotated chart of all of history (from Creation) on a 54-foot scroll (Ferguson, 1991). Joseph Priestley, presumably independently, used a more convenient form to show first a timeline chart of biography (lifespans of 2000 famous people, 1200 B.C. to A.D. 1750, Priestley, 1765), and then a detailed chart of history (Priestley, 1769).

The use of geometric figures (squares or rectangles) and cartograms to compare areas or demographic quantities by Charles de Fourcroy⁶ (1782) and August F.W. Crome (1785) provided another novel visual encoding for quantitative data using superimposed squares to compare the areas of European states.

As well, several technological innovations provided necessary ingredients for the production and dissemination of graphic works. Some of these facilitated the reproduction of data images, such as three-colour printing, invented by Jacob le Blon in 1710, and lithography, invented by Aloys Senefelder in 1798. Of the latter, Robinson (1982, p. 57) says “the effect was as great as the introduction [of the Xerox machine].” Yet, likely due to expense, most of these new graphic forms appeared in publications with limited circulation, unlikely to attract wide attention.

⁶ Image: <http://math.yorku.ca/SCS/Gallery/images/palsky/defourcroy.jpg>

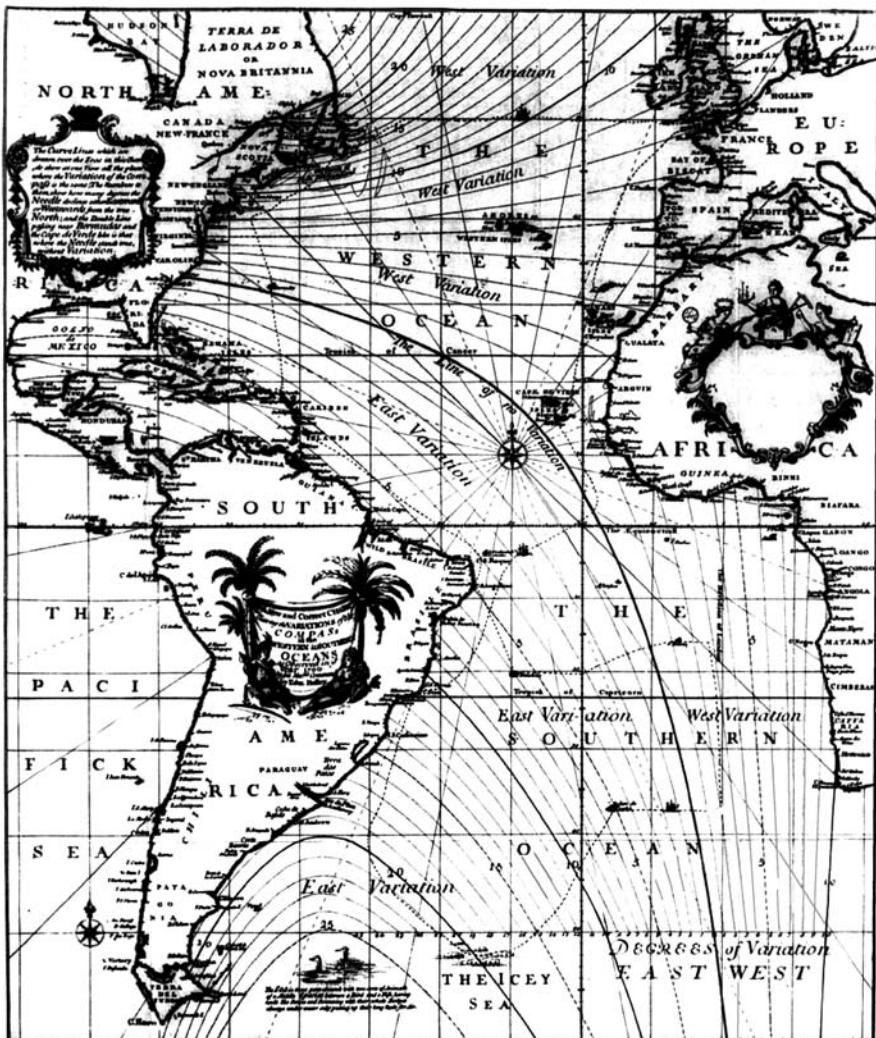


Figure 1.5. A portion of Edmund Halley's *New and Correct Sea Chart Shewing the Variations in the Compass in the Western and Southern Ocean*, 1701. Source: Halley (1701), image from Palsky (1996, p. 41)

A prodigious contributor to the use of the new graphical methods, Johann Lambert [1728–1777] introduced the ideas of curve fitting and interpolation from empirical data points. He used various sorts of line graphs and graphical tables to show periodic variation in, for example, air and soil temperature.⁷

William Playfair [1759–1823] is widely considered the inventor of most of the graphical forms used today – first the line graph and barchart (Playfair, 1786), later the

⁷ Image: <http://www.journals.uchicago.edu/Isis/journal/demo/v000n000/000000/fg7.gif>

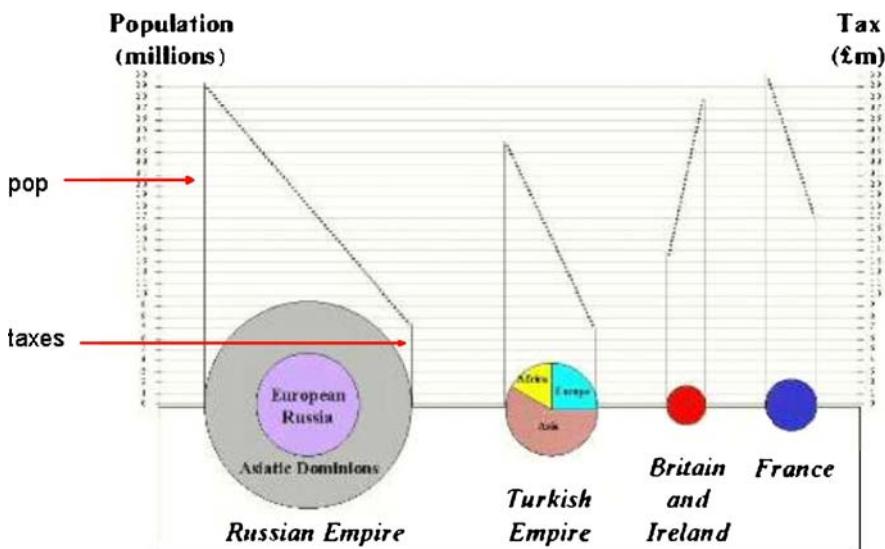


Figure 1.6. Redrawn version of a portion of Playfair's 1801 pie-circle-line chart, comparing population and taxes in several nations

piechart and circle graph (Playfair, 1801). Figure 1.6 shows a creative combination of different visual forms: circles, pies and lines, redrawn from Playfair (1801, Plate 2).

The use of two separate vertical scales for different quantities (population and taxes) is today considered a sin in statistical graphics (you can easily jiggle either scale to show different things). But Playfair used this device to good effect here to try to show taxes per capita in various nations and argue that the British were over-taxed, compared with others. But, alas, showing simple numbers by a graph was hard enough for Playfair – he devoted several pages of text in Playfair (1786) describing how to read and understand a line graph. The idea of calculating and graphing rates and other indirect measurements was still to come.

In this figure, the left axis and line on each circle/pie graph shows population, while the right axis and line shows taxes. Playfair intended that the *slope* of the line connecting the two would depict the rate of taxation directly to the eye; but, of course, the slope also depends on the diameters of the circles. Playfair's graphic sins can perhaps be forgiven here, because the graph clearly shows the slope of the line for Britain to be in the opposite direction of those for the other nations.

A somewhat later graph (Playfair, 1821), shown in Fig. 1.7, exemplifies the best that Playfair had to offer with these graphic forms. Playfair used three parallel time series to show the price of wheat, weekly wages and reigning ruler over a 250-year span from 1565 to 1820 and used this graph to argue that workers had become better off in the most recent years.

By the end of this century (1794), the utility of graphing in scientific applications prompted a Dr Buxton in London to patent and market printed coordinate paper; curiously, a patent for lined notepaper was not issued until 1815. The first known

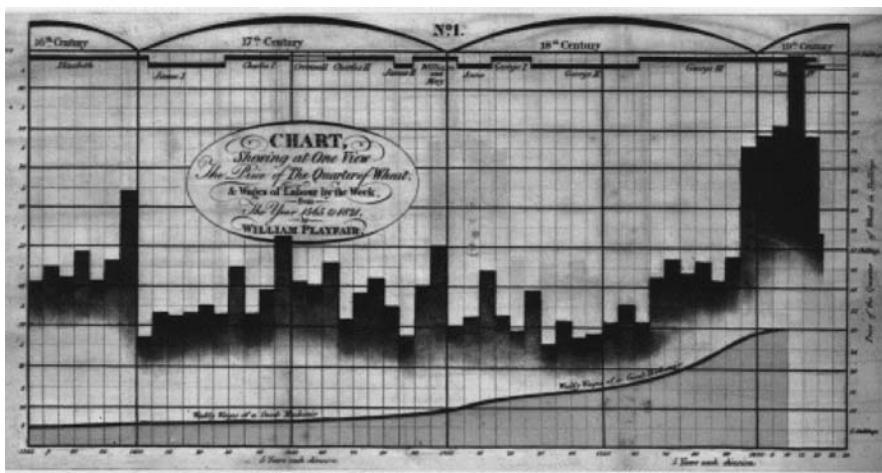


Figure 1.7. William Playfair's 1821 time-series graph of prices, wages and reigning ruler over a 250-year period. Source: Playfair (1821), image from Tufte (1983, p. 34)

published graph using coordinate paper is one of periodic variation in barometric pressure (Howard, 1800). Nevertheless, graphing of data would remain rare for another 30 or so years,⁸ perhaps largely because there wasn't much quantitative information (apart from widespread astronomical, geodetic and physical measurement) of sufficient complexity to require new methods and applications. Official statistics, regarding population and mortality, and economic data were generally fragmentary and often not publicly available. This would soon change.

1800–1850: Beginnings of Modern Graphics

1.2.4

With the fertilization provided by the previous innovations of design and technique, the first half of the 19th century witnessed explosive growth in statistical graphics and thematic mapping, at a rate which would not be equalled until modern times.

In statistical graphics, all of the modern forms of data display were invented: bar- and piecharts, histograms, line graphs and time-series plots, contour plots, scatter-plots and so forth. In thematic cartography, mapping progressed from single maps to comprehensive atlases, depicting data on a wide variety of topics (economic, social, moral, medical, physical, etc.), and introduced a wide range of novel forms of symbolism. During this period graphical analysis of natural and physical phenomena (lines of magnetism, weather, tides, etc.) began to appear regularly in scientific publications as well.

In 1801, the first geological maps were introduced in England by William Smith [1769–1839], setting the pattern for geological cartography or 'stratigraphic geology'

⁸ William Herschel (1833), in a paper that describes the first instance of a modern scatterplot, devoted three pages to a description of plotting points on a grid.

(Smith, 1815). These and other thematic maps soon led to new ways of showing quantitative information on maps and, equally importantly, to new domains for graphically based inquiry.

In the 1820s, Baron Charles Dupin [1784–1873] invented the use of continuous shadings (from white to black) to show the distribution and degree of illiteracy in France (Dupin, 1826) – the first unclassed choropleth map,⁹ and perhaps the first modern-style thematic statistical map (Palsky, 1996, p. 59). Later given the lovely title ‘*Carte de la France obscure et de la France éclairée*,’ it attracted wide attention, and was also perhaps the first application of graphics in the social realm.

More significantly, in 1825, the ministry of justice in France instituted the first centralized national system of crime reporting, collected quarterly from all departments and recording the details of every charge laid before the French courts. In 1833, André-Michel Guerry, a lawyer with a penchant for numbers, used these data (along with other data on literacy, suicides, donations to the poor and other ‘moral’ variables) to produce a seminal work on the moral statistics of France (Guerry, 1833) – a work that (along with Quetelet, 1831, 1835) can be regarded as the foundation of modern social science.¹⁰

Guerry used maps in a style similar to Dupin to compare the ranking of departments on pairs of variables, notably crime vs. literacy, but other pairwise variable comparisons were made.¹¹ He used these to argue that the lack of an apparent (negative) relation between crime and literacy contradicted the armchair theories of some social reformers who had argued that the way to reduce crime was to increase education.¹² Guerry’s maps and charts made somewhat of an academic sensation both in France and the rest of Europe; he later exhibited several of these at the 1851 London Exhibition and carried out a comparative study of crime in England and France (Guerry, 1864) for which he was awarded the Moynton Prize in statistics by the French Academy of Sciences.¹³ But Guerry’s systematic and careful work was unable

⁹ Image: http://math.yorku.ca/SCS/Gallery/images/dupin1826-map_200.jpg

¹⁰ Guerry showed that rates of crime, when broken down by department, type of crime, age and gender of the accused and other variables, remained remarkably consistent from year to year, yet varied widely across departments. He used this to argue that such regularity implied the possibility of establishing social laws, much as the regularity of natural phenomena implied physical ones. Guerry also pioneered the study of suicide, with tabulations of suicides in Paris, 1827–1830, by sex, age, education, profession, etc., and a content analysis of suicide notes as to presumed motives.

¹¹ Today, one would use a scatterplot, but that graphic form had only just been invented (Herschel, 1833) and would not enter common usage for another 50 years; see Friendly and Denis (2005).

¹² Guerry seemed reluctant to take sides. He also contradicted the social conservatives who argued for the need to build more prisons or impose more severe criminal sentences. See Whitt (2002).

¹³ Among the 17 plates in this last work, seven pairs of maps for England and France each included sets of small line graphs to show trends over time, decompositions by subtype of crime and sex, distributions over months of the year, and so forth. The final plate, on general causes of crime, is an incredibly detailed and complex multivariate semi-graphic display attempting to relate various types of crimes to each other, to various social and moral aspects (instruction, religion, population) as well as to their geographic distribution.

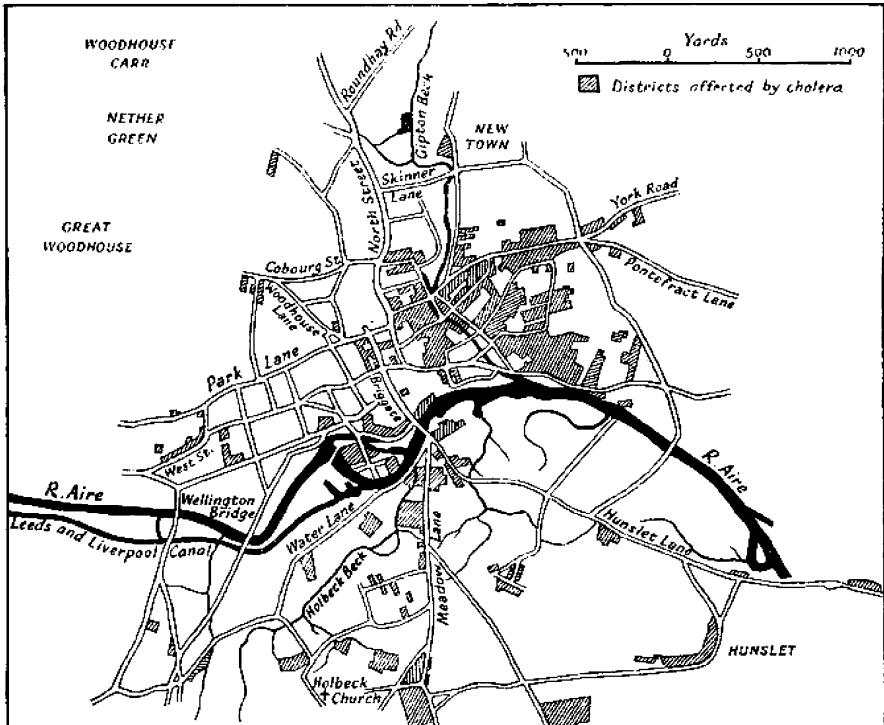


Figure 1.8. A portion of Dr Robert Baker's cholera map of Leeds, 1833, showing the districts affected by cholera. Source: Gilbert (1958, Fig. 2)

to shine in the shadows cast by Adolphe Quetelet, who regarded moral and social statistics as his own domain.

In October 1831, the first case of Asiatic cholera occurred in Great Britain, and over 52 000 people died in the epidemic that ensued over the next 18 months or so (Gilbert, 1958). Subsequent cholera epidemics in 1848–1849 and 1853–1854 produced similarly large death tolls, but the water-borne cause of the disease was unknown until 1855 when Dr John Snow produced his famous dot map¹⁴ (Snow, 1855) showing deaths due to cholera clustered around the Broad Street pump in London. This was indeed a landmark graphic discovery, but it occurred at the end of the period, roughly 1835–1855, which marks a high point in the application of thematic cartography to human (social, medical, ethnic) topics. The first known disease map of cholera (Fig. 1.8), due to Dr Robert Baker (1833), shows the districts of Leeds 'affected by cholera' in the particularly severe 1832 outbreak.

I show this figure to make another point – why Baker's map did not lead to a 'eureka' experience, while John Snow's did. Baker used a town plan of Leeds that had been divided into districts. Of a population of 76 000 in all of Leeds, Baker mapped

¹⁴ Image: <http://www.math.yorku.ca/SCS/Gallery/images/snow4.jpg>

the 1800 cholera cases by hatching in red ‘the districts in which the cholera had prevailed.’ In his report, he noted an association between the disease and living conditions: ‘how exceedingly the disease has prevailed in those parts of the town where there is a deficiency, often an entire want of sewage, drainage and paving’ (Baker, 1833, p. 10). Baker did not indicate the incidence of disease on his map, nor was he equipped to display *rates* of disease (in relation to population density),¹⁵ and his knowledge of possible causes, while definitely on the right track, was both weak and implicit (not analysed graphically or by other means). It is likely that some, perhaps tenuous, causal indicants or evidence were available to Baker, but he was unable to connect the dots or see a geographically distributed outcome in relation to geographic factors in even the simple ways that Guerry had tried.

At about the same time, 1830–1850, the use of graphs began to become recognized in some official circles for economic and state planning – where to build railroads and canals? What is the distribution of imports and exports? This use of graphical methods is no better illustrated than in the works of Charles Joseph Minard [1781–1870], whose prodigious graphical inventions led Funkhouser (1937) to call him the Playfair of France. To illustrate, we choose (with some difficulty) an 1844 ‘tableau-graphique’ (Fig. 1.9) by Minard, an early progenitor of the modern mosaicplot (Friendly, 1994). On the surface, mosaicplots descend from bar charts, but Minard introduced two simultaneous innovations: the use of divided and proportional-width bars so that area had a concrete visual interpretation. The graph shows the transportation of commercial goods along one canal route in France by variable-width, divided bars (Minard, 1844). In this display the width of each vertical bar shows distance along this route; the divided-bar segments have height proportional to amount of goods of various types (shown by shading), so the area of each rectangular segment is proportional to the cost of transport. Minard, a true visual engineer (Friendly, 2000), developed such diagrams to argue visually for setting differential price rates for partial vs. complete runs. Playfair had tried to make data ‘speak to the eyes,’ but Minard wished to make them ‘calculer par l’œil’ as well.

It is no accident that, in England, outside the numerous applications of graphical methods in the sciences, there was little interest in or use of graphs amongst statisticians (or ‘statists’ as they called themselves). If there is a continuum ranging from ‘graph people’ to ‘table people,’ British statisticians and economists were philosophically more table-inclined and looked upon graphs with suspicion up to the time of William Stanley Jevons around 1870 (Maas and Morgan, 2005). Statistics should be concerned with the recording of ‘facts relating to communities of men which are capable of being expressed by numbers’ (Mouat, 1885, p. 15), leaving the generalization to laws and theories to others. Indeed, this view was made abundantly clear in the logo of the Statistical Society of London (now the Royal Statistical Society): a banded

¹⁵ The German geographer Augustus Petermann produced a ‘Cholera map of the British Isles’ in 1852 using national data from the 1831–1832 epidemic (image: <http://images.rgs.org/webimages/0/0/10000/1000/800/S0011888.jpg>) shaded in proportion to the relative rate of mortality using class intervals ($< 1/35, 1/35 : 1/100, 1/100 : 1/200, \dots$). No previous disease map had allowed determination of the range of mortality in any given area.

C^otableau figuratif du mouvement commercial du Canal du Centre en 1844
dressé par M^{me} Moineau sur les renseignements de M^{me} Comte. x^o 1845 Ch^ef Otteman^d
Le mouvement total équivaut à 111.000 tonnes passant le longeur du Canal en 11. kilomètres
Le tonnage y est réparti pour 10.000 tonnes.

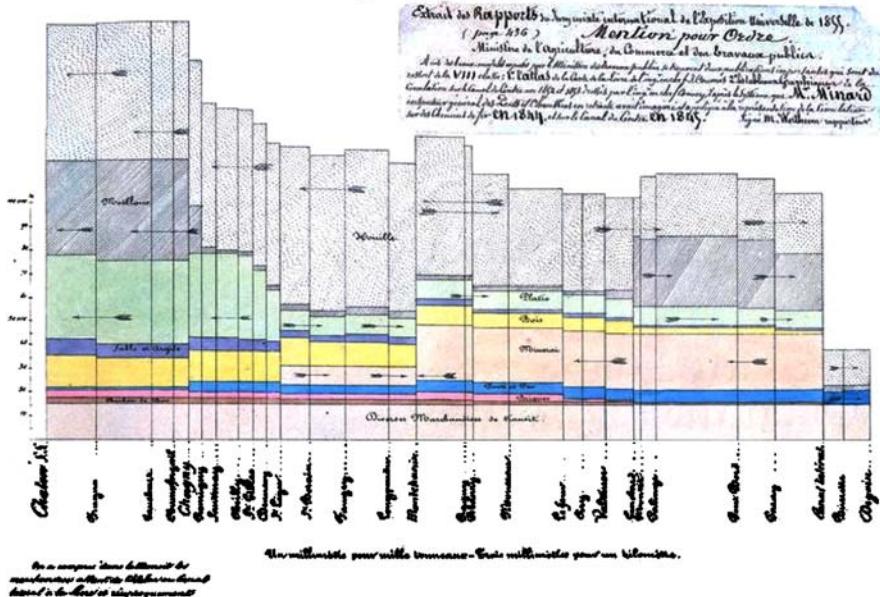


Figure 1.9. Minard's Tableau Graphique, showing the transportation of commercial goods along the Canal du Centre (Chalon–Dijon). Intermediate stops are spaced by distance, and each *bar* is divided by type of goods, so the area of each tile represents the cost of transport. Arrows show the direction of transport. Source: ENPC:5860/C351 (Col. et cliché ENPC; used by permission)

sheaf of wheat, with the motto *Aliis Exterendum* – to others to flail the wheat. Making graphs, it seemed, was too much like breadmaking.

1850–1900: The Golden Age of Statistical Graphics

125

By the mid-1800s, all the conditions for the rapid growth of visualization had been established – a ‘perfect storm’ for data graphics. Official state statistical offices were established throughout Europe, in recognition of the growing importance of numerical information for social planning, industrialization, commerce and transportation. Statistical theory, initiated by Gauss and Laplace and extended to the social realm by Guerry and Quetelet, provided the means to make sense of large bodies of data.

What started as the *Age of Enthusiasm* (Funkhouser, 1937; Palsky, 1996) for graphics ended with what can be called the *Golden Age*, with unparalleled beauty and many innovations in graphics and thematic cartography. So varied were these developments that it is difficult to be comprehensive, but a few themes stand out.

Escaping Flatland

Although some attempts to display more than two variables simultaneously had occurred earlier in multiple time series (Playfair, 1801; Minard, 1826), contour graphs (Vauthier, 1874) and a variety of thematic maps, (e.g. Berghaus (1838)) a number of significant developments extended graphics beyond the confines of a flat piece of paper. Gustav Zeuner [1828–1907] in Germany (Zeuner, 1869), and later Luigi Perozzo [1750–1875] in Italy (Perozzo, 1880) constructed 3-D surface plots of population data.¹⁶ The former was an axonometric projection showing various slices, while the latter (a 3-D graph of population in Sweden from 1750–1875 by year and age group) was printed in red and black and designed as a stereogram.¹⁷

Contour diagrams, showing isolevel curves of 3-D surfaces, had also been used earlier in mapping contexts (Nautonier, 1602–1604; Halley, 1701; von Humboldt, 1817), but the range of problems and data to which they were applied expanded considerably over this time in attempts to understand relations among more than two data-based variables, or where the relationships are statistical, rather than functional or measured with little error. It is more convenient to describe these under Galton, below. By 1884, the idea of visual and imaginary worlds of varying numbers of dimensions found popular expression in Edwin Abbott's (1884) *Flatland*, implicitly suggesting possible views in four and more dimensions.

Graphical Innovations

With the usefulness of graphical displays for understanding complex data and phenomena established, many new graphical forms were invented and extended to new areas of inquiry, particularly in the social realm.

Minard (1861) developed the use of divided circle diagrams on maps (showing both a total, by area, and subtotals, by sectors, with circles for each geographic region on the map). Later he developed to an art form the use of flow lines on maps of width proportional to quantities (people, goods, imports, exports) to show movement and transport geographically. Near the end of his life, the flow map would be taken to its highest level in his famous depiction of the fate of the armies of Napoleon and Hannibal, in what Tufte (1983) would call the 'best graphic ever produced.' See Friendly (2002) for a wider appreciation of Minard's work.

The social and political uses of graphics is also evidenced in the polar area charts (called 'rose diagrams' or 'coxcombs') invented by Florence Nightingale [1820–1910] to wage a campaign for improved sanitary conditions in battlefield treatment of soldiers (Nightingale, 1857). They left no doubt that many more soldiers died from disease and the consequences of wounds than at the hands of the enemy. From around the same time, Dr John Snow [1813–1858] is remembered for his use of a dot map of deaths from cholera in an 1854 outbreak in London. Plotting the residence of each

¹⁶ Image: <http://math.yorku.ca/SCS/Gallery/images/stereo2.jpg>

¹⁷ Zeuner used one axis to show year of birth and another to show present age, with number of surviving persons on the third, vertical, axis giving a 3-D surface. One set of curves thus showed the distribution of population for a given generation; the orthogonal set of curves showed the distributions across generations at a given point in time, e.g. at a census.

deceased provided the insight for his conclusion that the source of the outbreak could be localized to contaminated water from a pump on Broad Street, the founding innovation for modern epidemiological mapping.

Scales and shapes for graphs and maps were also transformed for a variety of purposes, leading to semi-logarithmic graphs (Jevons, 1863, 1879) to show percentage change in commodities over time, log-log plots to show multiplicative relations, anamorphic maps by Émile Cheysson (Palsky, 1996, Figs. 63–64) using deformations of spatial size to show a quantitative variable (e.g. the decrease in time to travel from Paris to various places in France over 200 years) and alignment diagrams or nomograms using sets of parallel axes. We illustrate this slice of the Golden Age with Fig. 1.10, a tour-de-force graphic for determination of magnetic deviation at sea in relation to latitude and longitude without calculation ('L'Abaque Triomphé') by Charles Lallemand (1885), director general of the geodetic measurement of altitudes throughout France, which combines many variables into a multifunction nomogram, using 3-D juxtaposition of anamorphic maps, parallel coordinates and hexagonal grids.

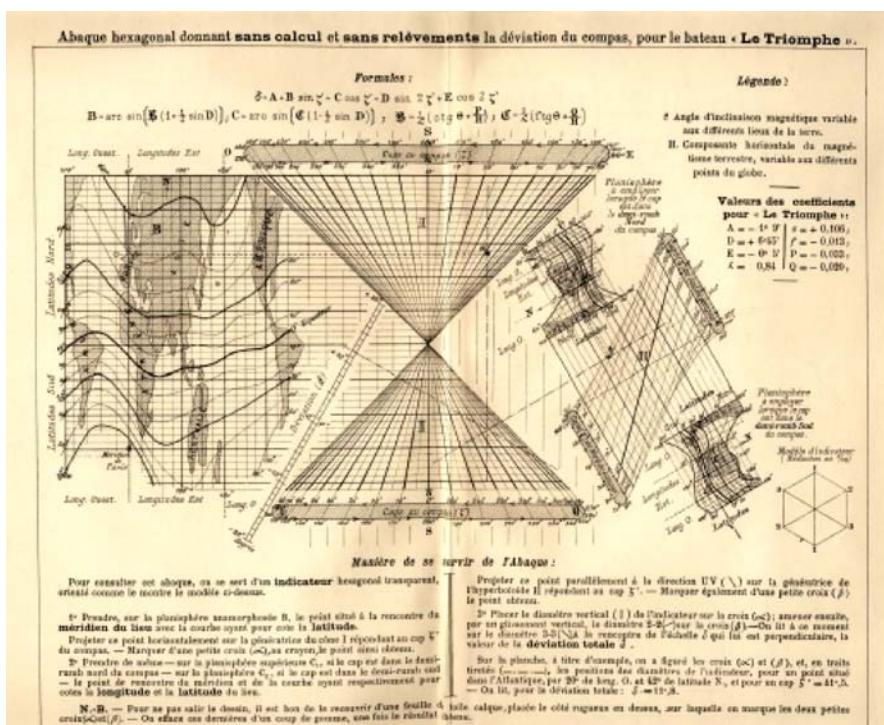


Figure 1.10. Lallemand's *L'abaque du bateau "Le Triomphé"*, allowing determination of magnetic deviation at sea without calculation. Source: courtesy Mme Marie-Noëlle Maisonneuve, Les fonds anciens de la bibliothèque de l'École des Mines de Paris

Galton's Contributions

Special note should be made of the varied contributions of Francis Galton [1822–1911] to data visualization and statistical graphics. Galton's role in the development of the ideas of correlation and regression are well known. Less well known is the role that visualization and graphing played in his contributions and discoveries.

Galton's statistical insight (Galton, 1886) – that, in a bivariate (normal) distribution, (say, height of a child against height of parents), (a) The isolines of equal frequency would appear as concentric ellipses and (b) The locus of the (regression) lines of means of $y|x$ and of $x|y$ were the conjugate diameters of these ellipses – was based largely on visual analysis from the application of smoothing to his data. Karl Pearson would later say 'that Galton should have evolved all this from his observations is to my mind one of the most noteworthy scientific discoveries arising from pure analysis of observations' (Pearson, 1920, p. 37). This was only one of Galton's discoveries based on graphical methods.

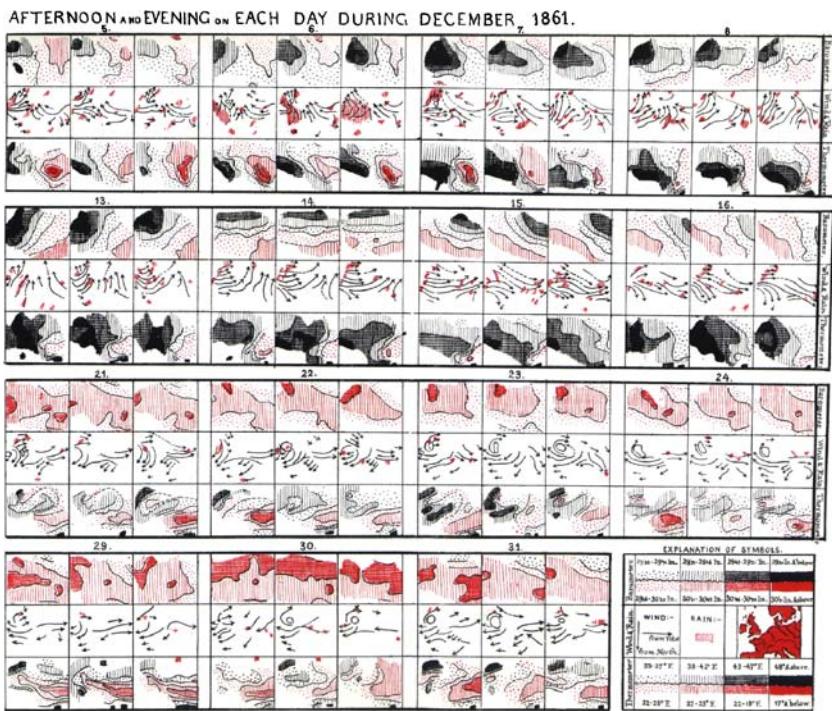
In earlier work, Galton had made wide use of isolines, contour diagrams and smoothing in a variety of areas. An 1872 paper showed the use of 'isodic curves' to portray the joint effects of wind and current on the distance ships at sea could travel in any direction. An 1881 'isochronic chart' (Galton, 1881) showed the time it took to reach any destination in the world from London by means of coloured regions on a world map. Still later, he analysed rates of fertility in marriages in relation to the ages of father and mother using 'isogens', curves of equal percentage of families having a child (Galton, 1894).

But perhaps the most notable non-statistical graphical discovery was that of the "anti-cyclonic" (anticlockwise) pattern of winds around low-pressure regions, combined with clockwise rotations around high-pressure zones. Galton's work on weather patterns began in 1861 and was summarized in *Meteorographica* (1863). It contained a variety of ingenious graphs and maps (over 600 illustrations in total), one of which is shown in Fig. 1.11. This remarkable chart, one of a two-page Trellis-style display, shows observations on barometric pressure, wind direction, rain and temperature from 15 days in December 1861.¹⁸ For each day, the 3×3 grid shows schematic maps of Europe, mapping pressure (row 1), wind and rain (row 2) and temperature (row 3), in the morning, afternoon and evening (columns). One can clearly see the series of black areas (low pressure) on the barometric charts for about the first half of the month, corresponding to the anticlockwise arrows in the wind charts, followed by a shift to red areas (high pressure) and more clockwise arrows. Wainer (2005, p. 56) remarks, 'Galton did for the collectors of weather data what Kepler did for Tycho Brahe. This is no small accomplishment.'

Statistical Atlases

The collection, organization and dissemination of official government statistics on population, trade and commerce, social, moral and political issues became wide-

¹⁸ In July 1861, Galton distributed a circular to meteorologists throughout Europe, asking them to record these data synchronously, three times a day for the entire month of December 1861. About 50 weather stations supplied the data; see Pearson (1914–1930, pp. 37–39).



published as large-format books (about 11 × 17 in.), and many of the plates folded out to four or six times that size, all printed in colour and with great attention to layout and composition. We concur with Funkhouser (1937, p. 336) that “the *Albums* present the finest specimens of French graphic work in the century and considerable pride was taken in them by the French people, statisticians and laymen alike.”

The subject matter of the albums largely concerned economic and financial data related to the planning, development and administration of public works – transport of passengers and freight, by rail, on inland waterways and through seaports, but also included such topics as revenues in the major theaters of Paris, attendance at the universal expositions of 1867, 1878 and 1889, changes in populations of French departments over time and so forth.

More significantly for this account the *Albums* can also be viewed as an exquisite sampler of all the graphical methods known at the time, with significant adaptations to the problem at hand. The majority of these graphs used and extended the flow map pioneered by Minard. Others used polar forms – variants of pie and circle diagrams, star plots and rose diagrams, often overlaid on a map and extended to show additional variables of interest. Still others used subdivided squares in the manner of modern mosaic displays (Friendly, 1994) to show the breakdown of a total (passengers, freight) by several variables. It should be noted that in almost all cases the graphical representation of the data was accompanied by numerical annotations or tables, providing precise numerical values.

The *Albums* are discussed extensively by Palsky (1996), who includes seven representative illustrations. It is hard to choose a single image here, but my favourites are surely the recursive, multimosaic of rail transportation for the 1884–1886 volumes, the first of which is shown in Fig. 1.12. This cartogram uses one large mosaic (in the lower left) to show the numbers of passengers and tons of freight shipped from Paris from the four principal train stations. Of the total leaving Paris, the amounts going to each main city are shown by smaller mosaics, coloured according to railway lines; of those amounts, the distribution to smaller cities is similarly shown, connected by lines along the rail routes.

Among the many other national statistical albums and atlases, those from the US Census bureau also deserve special mention. The *Statistical Atlas of the Ninth Census*, produced in 1872–1874 under the direction of Francis A. Walker [1840–1897], contained 60 plates, including several novel graphic forms. The ambitious goal was to present a graphic portrait of the nation, and it covered a wide range of physical and human topics: geology, minerals and weather; population by ethnic origin, wealth, illiteracy, school attendance and religious affiliation; death rates by age, sex, race and cause; prevalence of blindness, deaf mutism and insanity; and so forth. ‘Age pyramids’ (back-to-back, bilateral frequency histograms and polygons) were used effectively to compare age distributions of the population for two classes (gender, married/single, etc.). Subdivided squares and area-proportional pies of various forms were also used to provide comparisons among the states on multiple dimensions simultaneously

as an engineer at the ENPC and later became a professor of political economy at the École des Mines.

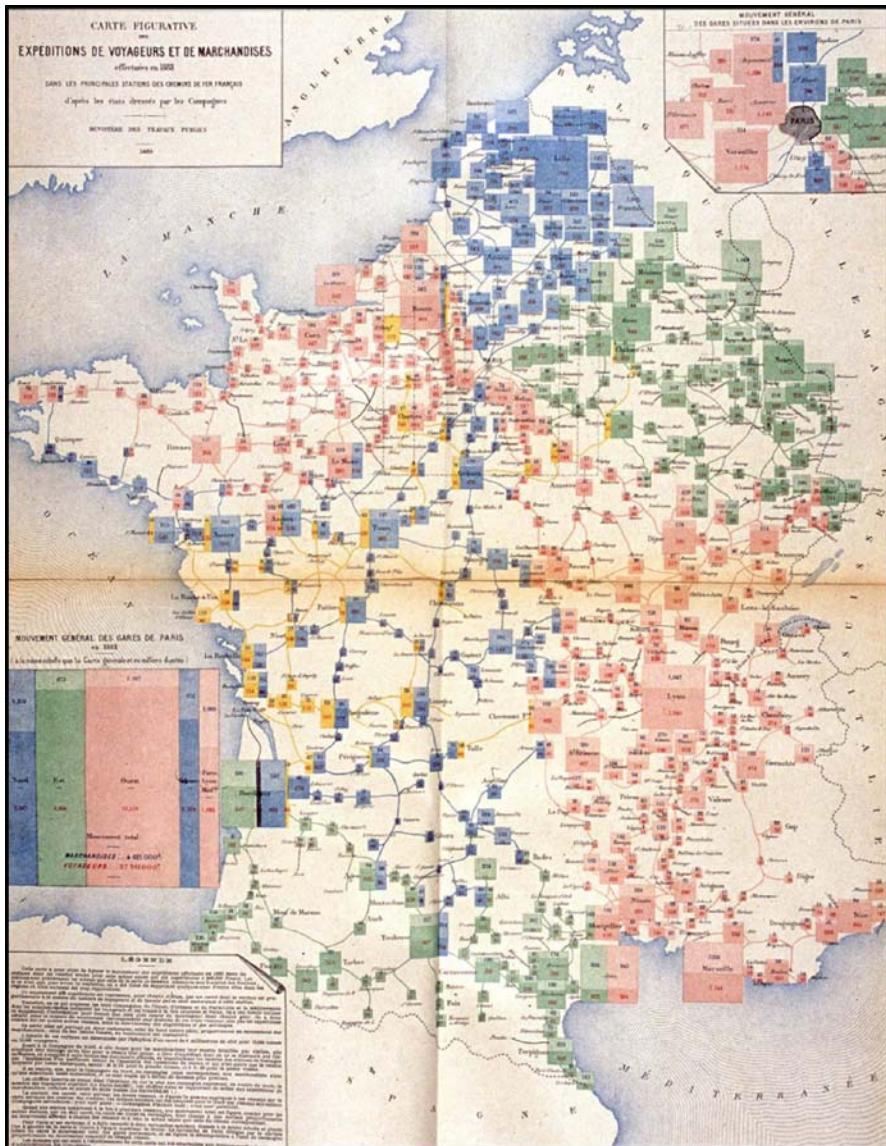


Figure 1.12. [This figure also appears in the color insert.] *Mouvement des voyageurs et des marchandises dans les principales stations de chemins de fer en 1882*. Scale: $2 \text{ mm}^2 = 10\,000$ passagers ou tons de fret. Source: Album, 1884, Plate 11 (author's collection)

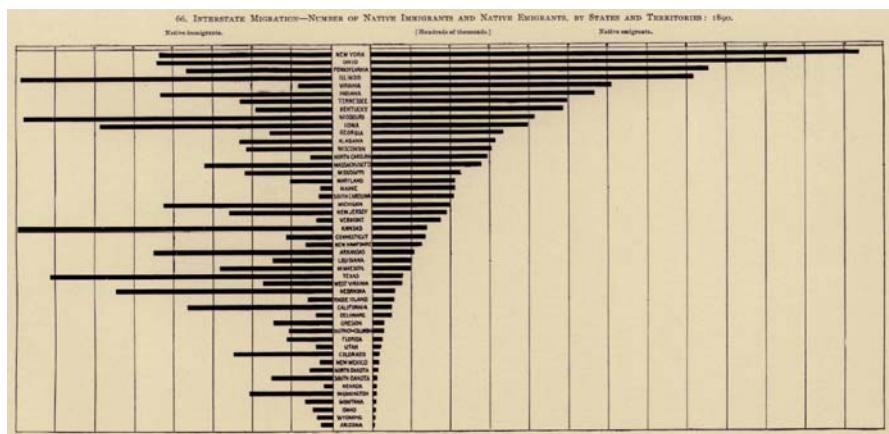
(employed/unemployed, sex, schooling, occupational categories). The desire to provide for easy comparisons among states and other categorizations was expressed by arranging multiple subfigures as 'small multiples' in many plates.

Following each subsequent decennial census for 1880 to 1900, reports and statistical atlases were produced with more numerous and varied graphic illustrations. The 1898 volume from the Eleventh Census (1890), under the direction of Henry Gannett [1846–1914], contained over 400 graphs, cartograms and statistical diagrams. There were several ranked parallel coordinate plots comparing states and cities over all censuses from 1790–1890. Trellis-like collections of shaded maps showed interstate migration, distributions of religious membership, deaths by known causes and so forth.

The 1880 and 1890 volumes produced under Gannett's direction are also notable for (a) the multimodal combination of different graphic forms (maps, tables, bar-charts, bilateral polygons) in numerous plates and (b) the consistent use of effect-order sorting (Friendly and Kwan, 2003) to arrange states or other categories in relation to what was to be shown, rather than for lookup (e.g. Alabama–Wyoming).

For example, Fig. 1.13 shows interstate immigration in relation to emigration for the 49 states and territories in 1890. The right side shows population loss sorted by emigration, ranging from New York, Ohio, Pennsylvania and Illinois at the top to Idaho, Wyoming and Arizona at the bottom. The left side shows where the emigrants went: Illinois, Missouri, Kansas and Texas had the biggest gains, Virginia the biggest net loss. It is clear that people were leaving the eastern states and were attracted to those of the Midwest Mississippi valley. Other plates showed this data in map-based formats.

However, the Age of Enthusiasm and the Golden Age were drawing to a close. The French *Albums de statistique graphique* were discontinued in 1897 due to the high cost of production; statistical atlases appeared in Switzerland in 1897 and 1914, but never again. The final two US Census atlases, issued after the 1910 and 1920 censuses, ‘were both routinized productions, largely devoid of colour and graphic imagination’ (Dahmann, 2001).



1900–1950: The Modern Dark Ages

1.2.6

If the late 1800s were the ‘golden age’ of statistical graphics and thematic cartography, the early 1900s can be called the ‘modern dark ages’ of visualization (Friendly and Denis, 2000).

There were few graphical innovations, and by the mid-1930s the enthusiasm for visualization which characterized the late 1800s had been supplanted by the rise of quantification and formal, often statistical, models in the social sciences. Numbers, parameter estimates and, especially, those with standard errors were precise. Pictures were – well, just pictures: pretty or evocative, perhaps, but incapable of stating a ‘fact’ to three or more decimals. Or so it seemed to many statisticians.

But it is equally fair to view this as a time of necessary dormancy, application and popularization rather than one of innovation. In this period statistical graphics became mainstream. Graphical methods entered English²⁰ textbooks (Bowley, 1901; Peddle, 1910; Haskell, 1919; Karsten, 1925), the curriculum (Costelloe, 1915; Warne, 1916) and standard use in government (Ayres, 1919), commerce (Gantt charts and Shewart’s control charts) and science.

These textbooks contained rather detailed descriptions of the graphic method, with an appreciative and often modern flavour. For example, Sir Arthur Bowley’s (1901) *Elements of Statistics* devoted two chapters to graphs and diagrams and discussed frequency and cumulative frequency curves (with graphical methods for finding the median and quartiles), effects of choice of scales and baselines on visual estimation of differences and ratios, smoothing of time-series graphs, rectangle diagrams in which three variables could be shown by height, width and area of bars, and ‘historical diagrams’ in which two or more time series could be shown on a single chart for comparative views of their histories.

Bowley’s (1901, pp. 151–154) example of smoothing (Fig. 1.14) illustrates the character of his approach. Here he plotted the total value of exports from Britain and Ireland over the period 1855–1899. At issue was whether exports had become stationary in the most recent years, and the conclusion by Sir Robert Giffen (1899), based solely on tables of averages for successive 5-year periods,²¹ that ‘the only sign of stationarity is an increase at a less rate in the last periods than in the earlier periods’ (p. 152). To answer this, he graphed the raw data, together with curves of the moving average over 3-, 5- and 10-year periods. The 3- and 5-year moving averages show strong evidence of an approximately 10-year cycle, and he noted, ‘no argument can stand which does not take account of the cycle of trade, which is not eliminated until we take decennial averages’ (p. 153). To this end, he took averages of successive 10-year periods starting 1859 and drew a freehand curve ‘keeping as close [to the points] as possible,

²⁰ The first systematic attempt to survey, describe and illustrate available graphic methods for experimental data was that of Étienne Jules Marey’s (1878) *La Méthode Graphique*. Marey [1830–1904] also invented several devices for visual recording, including the spymograph and chronophotography to record the motion of birds in flight, people running and so forth.

²¹ Giffen, an early editor of *The Statist*, also wrote a statistical text published posthumously in 1913; it contained an entire chapter on constructing tables, but not a single graph (Klein, 1997, p. 17).

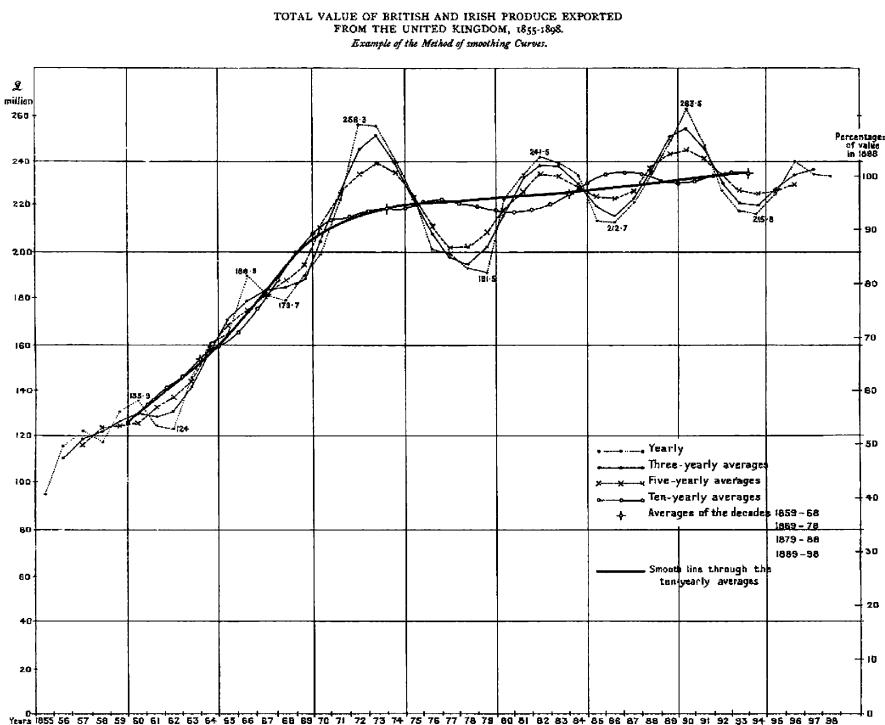


Figure 1.14. Arthur Bowley's demonstration of methods of smoothing a time-series graph. Moving averages of 3, 5 and 10 years are compared with a freehand curve drawn through four points representing the averages of successive 10-year periods. *Source:* Bowley (1901, opposite p. 151)

without making sudden changes in curvature,' giving the thick curve in Fig. 1.14.²² Support for Sir Robert's conclusion and the evidence for a 10-year cycle owe much to this graphical treatment.

Moreover, perhaps for the first time, graphical methods proved crucial in a number of new insights, discoveries and theories in astronomy, physics, biology and other sciences. Among these, one may refer to (a) E.W. Maunder's (1904) 'butterfly diagram' to study the variation of sunspots over time, leading to the discovery that they were markedly reduced in frequency from 1645–1715; (b) the Hertzsprung–Russell diagram (Hertzsprung, 1911; Spence and Garrison, 1993), a log-log plot of luminosity as a function of temperature for stars, used to explain the changes as a star evolves and laying the groundwork for modern stellar physics; (c) the discovery of the concept of atomic number by Henry Moseley (1913) based largely on graphical analysis. See Friendly and Denis (2005) for more detailed discussion of these uses.

²² A reanalysis of the data using a loess smoother shows that this is in fact oversmoothed and corresponds closely to a loess window width of $f = 0.50$. The optimal smoothing parameter, minimizing AIC_C is $f = 0.16$, giving a smooth more like Bowley's 3- and 5-year moving averages.

As well, experimental comparisons of the efficacy of various graphics forms were begun (Eells, 1926; von Huhn, 1927; Washburne, 1927), a set of standards and rules for graphic presentation was finally adopted by a joint committee (Joint Committee on Standards for Graphic Presentation, 1914) and a number of practical aids to graphing were developed. In the latter part of this period, new ideas and methods for multidimensional data in statistics and psychology would provide the impetus to look beyond the 2-D plane.

Graphic innovation was also awaiting new ideas and technology: the development of the machinery of modern statistical methodology, and the advent of the computational power and display devices which would support the next wave of developments in data visualization.

1950–1975: Rebirth of Data Visualization

1.2.7

Still under the influence of the formal and numerical zeitgeist from the mid-1930s on, data visualization began to rise from dormancy in the mid-1960s. This was spurred largely by three significant developments:

- In the USA, John W. Tukey [1915–2000], in a landmark paper, *The Future of Data Analysis* (Tukey, 1962), issued a call for the recognition of data analysis as a legitimate branch of statistics distinct from mathematical statistics; shortly later, he began the invention of a wide variety of new, simple and effective graphic displays, under the rubric of ‘exploratory data analysis’ (EDA) – stem-leaf plots, box-plots, hanging rootograms, two-way table displays and so forth, many of which entered the statistical vocabulary and software implementation. Tukey’s stature as a statistician and the scope of his informal, robust and graphical approach to data analysis were as influential as his graphical innovations. Although not published until 1977, chapters from Tukey’s EDA book (Tukey, 1977) were widely circulated as they began to appear in 1970–1972 and began to make graphical data analysis both interesting and respectable again.
- In France, Jacques Bertin [1918–] published the monumental *Sémiologie graphique* (Bertin, 1967). To some, this appeared to do for graphics what Mendeleev had done for the organization of the chemical elements, that is, to organize the visual and perceptual elements of graphics according to the features and relations in data. In a parallel but separate stream, an exploratory and graphical approach to multidimensional data (‘l’analyse des données’) begun by Jean-Paul Benzécri [1932–] provided French and other European statisticians with an alternative, visually based view of what statistics was about. Other graphically minded schools of data-thought would later arise in the Netherlands (Gifi), Germany and elsewhere in Europe.
- But the skills of hand-drawn maps and graphics had withered during the dormant ‘modern dark ages’ of graphics (though nearly every figure in Tukey’s *EDA* (Tukey, 1977) was, by intention, hand-drawn). Computer processing of statistical data began in 1957 with the creation of FORTRAN, the first high-level language for computing. By the late 1960s, widespread mainframe university computers offered the possibility to construct old and new graphic forms by computer

programs. Interactive statistical applications, e.g. Fowlkes (1969); Fishkeller et al. (1974), and true high-resolution graphics were developed but would take a while to enter common use.

By the end of this period significant intersections and collaborations would begin: (a) Computer science research (software tools, C language, UNIX, etc.) at Bell Laboratories (Becker, 1994) and elsewhere would combine forces with (b) Developments in data analysis (EDA, psychometrics, etc.) and (c) Display and input technology (pen plotters, graphic terminals, digitizer tablets, the mouse, etc.). These developments would provide new paradigms, languages and software packages for expressing statistical ideas and implementing data graphics. In turn, they would lead to an explosive growth in new visualization methods and techniques.

Other themes began to emerge, mostly as initial suggestions: (a) Various novel visual representations of multivariate data (Andrews' (1972) Fourier function plots, Chernoff (1973) faces, star plots, clustering and tree representations); (b) The development of various dimension-reduction techniques (biplot (Gabriel, 1971), multidimensional scaling, correspondence analysis), providing visualization of multidimensional data in a 2-D approximation; (c) Animations of a statistical process; and (d) Perceptually based theory and experiments related to how graphic attributes and relations might be rendered to better convey data visually.

By the close of this period, the first exemplars of modern GIS and interactive systems for 2-D and 3-D statistical graphics would appear. These would set goals for future development and extension.

1975–present: High-D, Interactive and Dynamic Data Visualization

1.2.8

During the last quarter of the 20th century data visualization blossomed into a mature, vibrant and multidisciplinary research area, as may be seen in this Handbook, and software tools for a wide range of visualization methods and data types are available for every desktop computer. Yet it is hard to provide a succinct overview of the most recent developments in data visualization because they are so varied and have occurred at an accelerated pace and across a wider range of disciplines. It is also more difficult to highlight the most significant developments which may be seen as such in a subsequent history focusing on this recent period.

With this disclaimer, a few major themes stand out.

- The development of highly interactive statistical computing systems. Initially, this meant largely command-driven, directly programmable systems (APL, S), as opposed to compiled, batch processing;
- New paradigms of direct manipulation for visual data analysis (linking, brushing (Becker and Cleveland, 1987), selection, focusing, etc.);
- New methods for visualizing high-dimensional data (the grand tour (Asimov, 1985), scatterplot matrix (Tukey and Tukey, 1981), parallel coordinates plot (Inselberg, 1985; Wegman, 1990), spreadplots (Young, 1994a), etc.);

-
- The invention (or re-invention) of graphical techniques for discrete and categorical data;
 - The application of visualization methods to an ever-expanding array of substantive problems and data structures; and
 - Substantially increased attention to the cognitive and perceptual aspects of data display.

These developments in visualization methods and techniques arguably depended on advances in theoretical and technological infrastructure, perhaps more so than in previous periods. Some of these are:

- Large-scale statistical and graphics software engineering, both commercial (e.g. SAS) and non-commercial (e.g. Lisp-Stat, the R project). These have often been significantly leveraged by open-source standards for information presentation and interaction (e.g. Java, Tcl/Tk);
- Extensions of classical linear statistical modelling to ever-wider domains (generalized linear models, mixed models, models for spatial/geographical data and so forth);
- Vastly increased computer processing speed and capacity, allowing computationally intensive methods (bootstrap methods, Bayesian MCMC analysis, etc.), access to massive data problems (measured in terabytes) and real-time streaming data. Advances in this area continue to press for new visualization methods.

From the early 1970s to mid-1980s, many of the advances in statistical graphics concerned static graphs for multidimensional quantitative data, designed to allow the analyst to see relations in progressively higher dimensions. Older ideas of dimension-reduction techniques (principal component analysis, multidimensional scaling, discriminant analysis, etc.) led to generalizations of projecting a high-dimensional dataset to ‘interesting’ low-dimensional views, as expressed by various numerical indices that could be optimized (projection pursuit) or explored interactively (grand tour).

The development of general methods for multidimensional contingency tables began in the early 1970s, with Leo Goodman (1970), Shelly Haberman (1973) and others (Bishop et al., 1975) laying out the fundamentals of log-linear models. By the mid-1980s, some initial, specialized techniques for visualizing such data were developed (four-fold display (Fienberg, 1975), association plot (Cohen, 1980), mosaicplot (Hartigan and Kleiner, 1981) and sieve diagram (Riedwyl and Schüpbach, 1983)), based on the idea of displaying frequencies by area (Friendly, 1995). Of these, extensions of the mosaicplot (Friendly, 1994, 1999) have proved most generally useful and are now widely implemented in a variety of statistical software, most completely in the vcd package (Meyer et al., 2005) in R and interactive software from the Augsburg group (MANET, Mondrian).

It may be argued that the greatest potential for recent growth in data visualization came from the development of interactive and dynamic graphic methods, allowing instantaneous and direct manipulation of graphical objects and related statistical properties. One early instance was a system for interacting with probability plots (Fowlkes, 1969) in real time, choosing a shape parameter of a reference distribution

and power transformations by adjusting a control. The first general system for manipulating high-dimensional data was PRIM-9, developed by Fishkeller, Friedman and Tukey (1974), and providing dynamic tools for projecting, rotating (in 3-D), isolating (identifying subsets) and masking data in up to 9 dimensions. These were quite influential, but remained one-of-a-kind, ‘proof-of-concept’ systems. By the mid-1980s, as workstations and display technology became cheaper and more powerful, desktop software for interactive graphics became more widely available (e.g. MacSpin, Xgobi). Many of these developments to that point are detailed in the chapters of *Dynamic Graphics for Statistics* (Cleveland and McGill, 1988).

In the 1990s, a number of these ideas were brought together to provide more general systems for dynamic, interactive graphics, combined with data manipulation and analysis in coherent and extensible computing environments. The combination of all these factors was more powerful and influential than the sum of their parts. Lisp-Stat (Tierney, 1990) and its progeny (Arc, Cook and Weisberg, 1999; ViSta, Young, 1994b), for example, provided an easily extensible object-oriented environment for statistical computing. In these systems, widgets (sliders, selection boxes, pick lists, etc.), graphs, tables, statistical models and the user all communicated through messages, acted upon by whoever was a designated ‘listener,’ and had a method to respond. Most of the ideas and methods behind present-day interactive graphics are described and illustrated in Young et al. (2006). Other chapters in this Handbook provide current perspectives on other aspects of interactive graphics.

1.3 Statistical Historiography

As mentioned at the outset, this review is based on the information collected for the Milestones Project, which I regard (subject to some caveats) as a relatively comprehensive corpus of the significant developments in the history of data visualization. As such, it is of interest to consider what light modern methods of statistics and graphics can shed on this history, a self-referential question we call ‘statistical historiography’ (Friendly, 2005). In return, this offers other ways to view this history.

1.3.1 History as ‘Data’

Historical events, by their nature, are typically discrete, but marked with dates or ranges of dates, and some description – numeric, textual, or classified by descriptors (who, what, where, how much and so forth). Amongst the first to recognize that history could be treated as data and portrayed visually, Joseph Priestley (1765; 1769) developed the idea of depicting the lifespans of famous people by horizontal lines along a time scale. His enormous (2×3 ft., or $.75 \times 1$ m) and detailed *Chart of Biography* showed two thousand names from 1200 B.C. to A.D. 1750 by horizontal lines from birth to death, using dots at either end to indicate ranges of uncertainty. Along the vertical dimension, Priestly classified these individuals, e.g., as statesmen or men of learning. A small fragment of this chart is shown in Fig. 1.15.

A Specimen of a Chart of Biography.

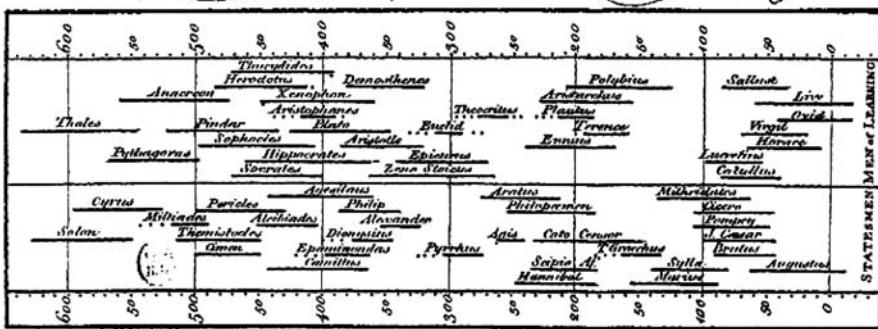


Figure 1.15. A specimen version of Priestley's *Chart of Biography*. Source: Priestley (1765)

Priestley's graphical representations of time and duration apparently influenced Playfair's introduction of time-series charts and barcharts (Funkhouser, 1937, p. 280). But these inventions did not inspire the British statisticians of his day, as noted earlier; historical events and statistical facts were seen as separate, rather than as data arrayed along a time dimension. In 1885, at the Jubilee meeting of the Royal Statistical Society, Alfred Marshall (1885) argued that the causes of historical events could be understood by the use of statistics displayed by 'historical curves' (time-series graphs): 'I wish to argue that the graphic method may be applied as to enable history to do this work better than it has hitherto' (p. 252). Maas and Morgan (2005) discuss these issues in more detail.

Analysing Milestones Data

1.3.2

The information collected in the Milestones Project is rendered in print and Web forms as a chronological list but is maintained as a relational database (historical items, references, images) in order to be able to work with it as 'data.' The simplest analyses examine trends over time. Figure 1.1 shows a density estimate for the distribution of 248 milestone items from 1500 to the present, keyed to the labels for the periods in history. The bumps, peaks and troughs all seem interpretable: note particularly the steady rise up to about 1880, followed by a decline through the 'modern dark ages' to 1945, then the steep rise up to the present. In fact, it is slightly surprising to see that the peak in the Golden Age is nearly as high as that at present, but this probably just reflects underrepresentation of the most recent events.²³

²³ Technical note: In this figure an optimal bandwidth for the kernel density estimate was selected (using the Sheather-Jones plug-in estimate) for each series separately. The smaller range and sample size of the entries for Europe vs. North America gives a smaller bandwidth for the former, by a factor of about 3. Using a common bandwidth fixed to that determined for the whole series (Fig. 1.1) undersmoothes the more extensive data on European develop-

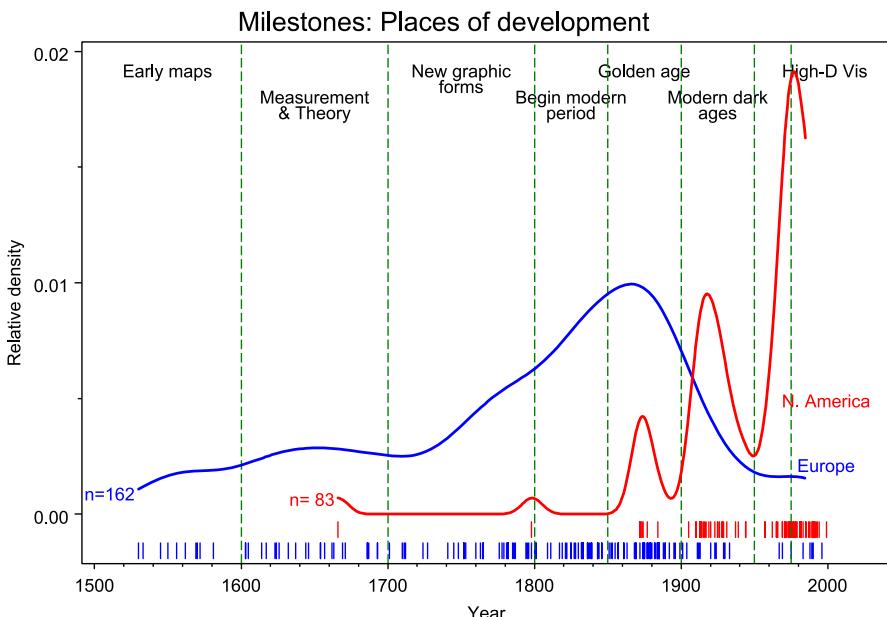


Figure 1.16. The distribution of milestone items over time, comparing trends in Europe and North America

Other historical patterns can be examined by classifying the items along various dimensions (place, form, content and so forth). If we classify the items by place of development (Europe vs. North America, ignoring Other), interesting trends appear (Fig. 1.16). The greatest peak in Europe around 1875–1880 coincided with a smaller peak in North America. The decline in Europe following the Golden Age was accompanied by an initial rise in North America, largely due to popularization (e.g. textbooks) and significant applications of graphical methods, then a steep decline as mathematical statistics held sway.

Finally, Fig. 1.17 shows two mosaicplots for the milestone items classified by Epoch, Subject matter and Aspect. Subject was classed as having to do with human (e.g. mortality, disease), physical or mathematical characteristics of what was represented in the innovation. Aspect classed each item according to whether it was primarily map-based, a diagram or statistical innovation or a technological one. The left mosaic shows the shifts in Subject over time: most of the early innovations concerned physical subjects, while the later periods shift heavily to mathematical ones. Human topics are not prevalent overall but were dominant in the 19th century. The right mosaic, for Subject \times Aspect, indicates that, unsurprisingly, map-based innovations were mainly about physical and human subjects, while diagrams and statistical ones were largely about mathematical subjects. Historical classifications clearly rely on more

ments and oversmoothes the North American ones. The details differ, but most of the points made in the discussion about what was happening when and where hold.

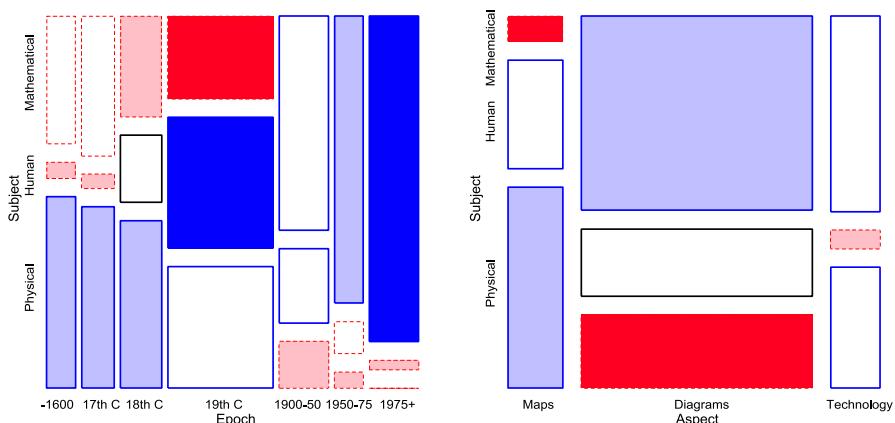


Figure 1.17. [This figure also appears in the color insert.] Mosaic plots for milestones items, classified by Subject, Aspect and Epoch. Cells with greater (less) frequency than expected under independence are coloured blue (red), with intensity proportional to the deviation from independence

detailed definitions than described here; however, it seems reasonable to suggest that such analyses of history as ‘data’ are a promising direction for future work.

What Was He Thinking? – Understanding Through Reproduction

1.3.3

Historical graphs were created using available data, methods, technology and understanding current at the time. We can often come to a better understanding of intellectual, scientific and graphical questions by attempting a re-analysis from a modern perspective.

Earlier, we showed Playfair’s time-series graph (Fig. 1.7) of wages and prices and noted that Playfair wished to show that workers were better off at the end of the period shown than at any earlier time. Presumably he wished to draw the reader’s eye to the narrowing of the gap between the bars for prices and the line graph for wages. Is this what you see?

What this graph shows directly is quite different from Playfair’s intention. It appears that wages remained relatively stable while the price of wheat varied greatly. The inference that wages increased relative to prices is indirect and not visually compelling.

We cannot resist the temptation to give Playfair a helping hand here – by graphing the ratio of wages to prices (labour cost of wheat), as shown in Fig. 1.18. But this would not have occurred to Playfair because the idea of relating one time series to another by ratios (index numbers) would not occur for another half-century (due to Jevons). See Friendly and Denis (2005) for further discussion of Playfair’s thinking.

As another example, we give a brief account of an attempt to explore Galton’s discovery of regression and the elliptical contours of the bivariate normal surface,

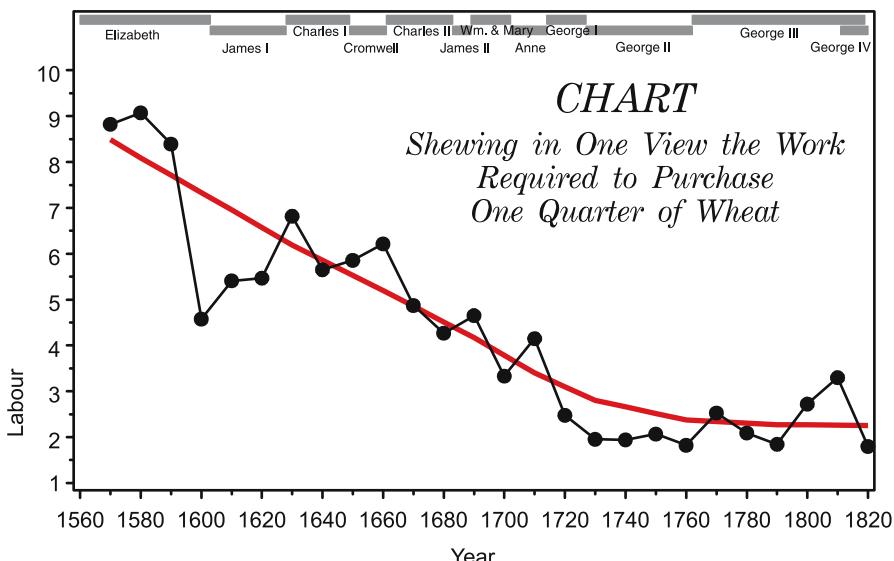


Figure 1.18. Redrawn version of Playfair's time-series graph showing the ratio of price of wheat to wages, together with a loess smoothed curve

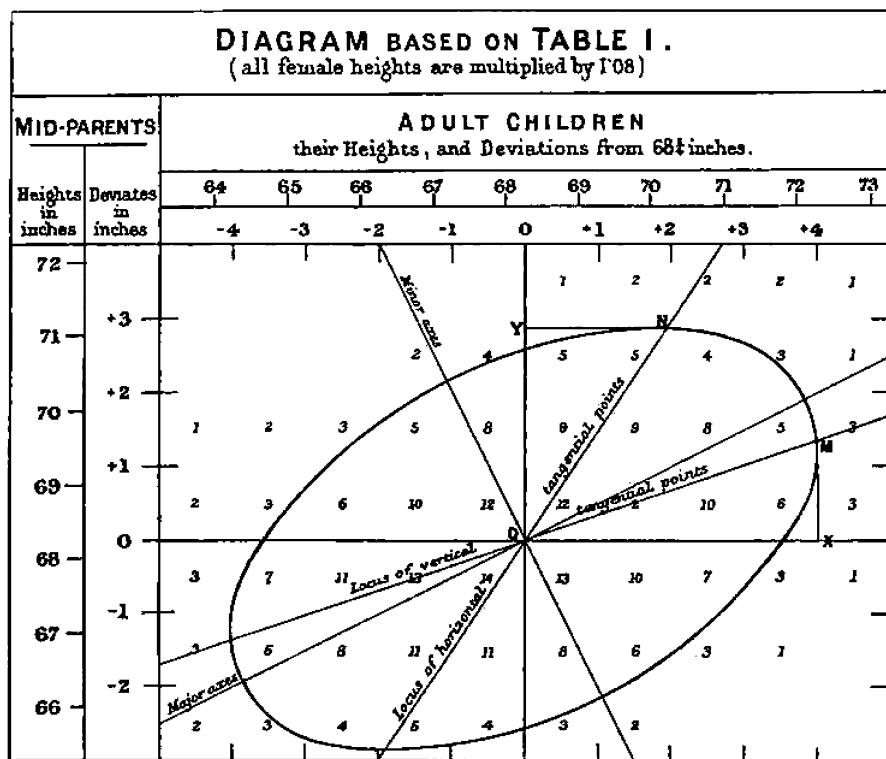


Figure 1.19. Galton's smoothed correlation diagram for the data on heights of parents and children, showing one ellipse of equal frequency. Source: (Galton, 1886, Plate X)

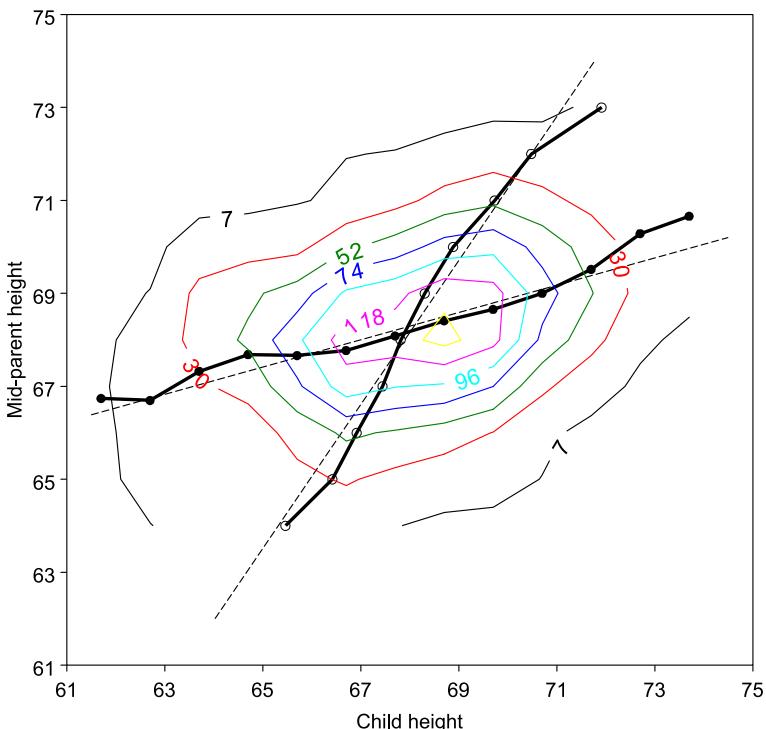


Figure 1.20. Contour plot of Galton's smoothed data, showing the curves of $\bar{y}|x$ (filled circles, solid line), $\bar{x}|y$ (open circles, solid line) and the corresponding regression lines (dashed)

treated in more detail in Friendly and Denis (2005). Galton's famous graph showing these relations (Fig. 1.19) portrays the joint frequency distribution of the height of children and the average height of their parents. It was produced from a 'semi-graphic table' in which Galton averaged the frequencies in each set of four adjacent cells, drew isocurves of equal smoothed value and noted that these formed 'concentric and similar ellipses.'

A literal transcription of Galton's method, using contour curves of constant average frequency and showing the curves of the means of $y|x$ and $x|y$, is shown in Fig. 1.20. It is not immediately clear that the contours are concentric ellipses, nor that the curves of means are essentially linear and have horizontal and vertical tangents to the contours.

A modern data analyst following the spirit of Galton's method might substitute a smoothed bivariate kernel density estimate for Galton's simple average of adjacent cells. The result, using jittered points to depict the cell frequencies, and a smoothed loess curve to show $E(y|x)$ is shown in Fig. 1.21. The contours now *do* emphatically suggest concentric similar ellipses, and the regression line is near the points of vertical tangency. A reasonable conclusion from these figures is that Galton did not slavishly interpolate isofrequency values as is done in the contour plot shown in

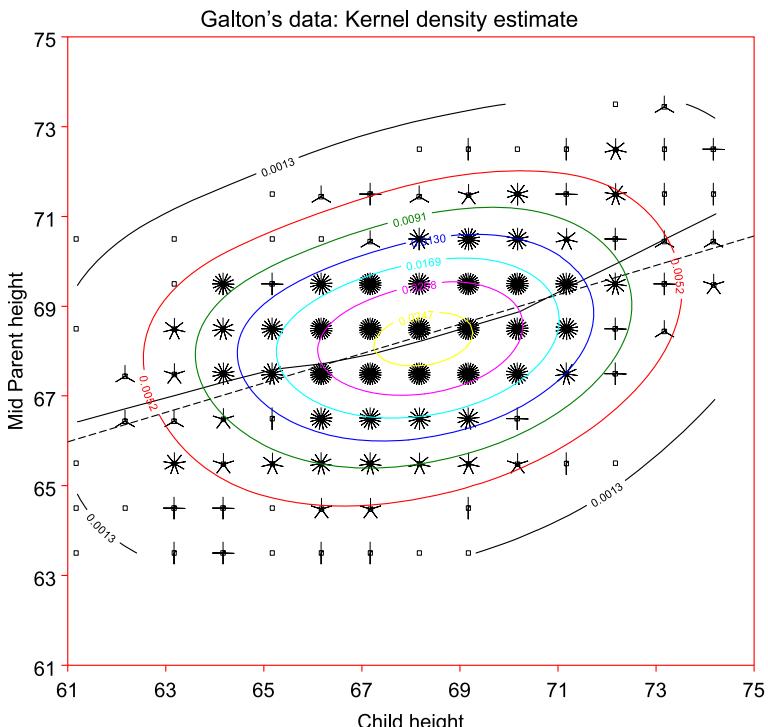


Figure 1.21. Bivariate kernel density estimate of Galton's data, using jittered points for the data, and a smoothed loess curve for $\mathcal{E}(y|x)$ (solid) and regression line (dashed)

Fig. 1.20. Rather, he drew his contours to the smoothed data by eye and brain (as he had done earlier with maps of weather patterns), with knowledge that he could, as one might say today, trade some increase in bias for a possible decrease in variance, and so achieve a greater smoothing.

1.4 Final Thoughts

This chapter is titled ‘A brief history...’ out of recognition that it is impossible to do full justice to the history of data visualization in such a short account. This is doubly so because I have attempted to present a broad view spanning the many areas of application in which data visualization took root and developed. That being said, it is hoped that this overview will lead modern readers and developers of graphical methods to appreciate the rich history behind the latest hot new methods. As we have seen, almost all current methods have a much longer history than is commonly thought. Moreover, as I have surveyed this work and travelled to many libraries to view original works and read historical sources, I have been struck by the exquisite

beauty and attention to graphic detail seen in many of these images, particularly those from the 19th century. We would be hard-pressed to recreate many of these today.

From this history one may also see that most of the innovations in data visualization arose from concrete, often practical, goals: the need or desire to see phenomena and relationships in new or different ways. It is also clear that the development of graphic methods depended fundamentally on parallel advances in technology, data collection and statistical theory. Finally, I believe that the application of modern methods of data visualization to its own history, in this self-referential way I call ‘statistical historiography,’ offers some interesting views of the past and challenges for the future.

Acknowledgement. This work is supported by Grant 8150 from the National Sciences and Engineering Research Council of Canada. I am grateful to the archivists of many libraries and to *les Chevaliers des Albums de Statistique Graphique*: Antoine de Falguerolles, Ruddy Ostermann, Gilles Palsky, Ian Spence, Antony Unwin, and Howard Wainer for historical information, images, and helpful suggestions.

References

- Abbott, E.A. (1884). *Flatland: A Romance of Many Dimensions*, Buccaneer Books, Cutchogue, NY. (1976 reprint of the 1884 edition).
- Andrews, D.F. (1972). Plots of high dimensional data, *Biometrics*, 28:125–136.
- Asimov, D. (1985). Grand tour, *SIAM Journal of Scientific and Statistical Computing*, 6(1):128–143.
- Ayres, L.P. (1919). *The War with Germany, A Statistical Summary*, U.S. Government Printing Office, Washington, D.C. Commonly known as the Ayres report; reprinted: Arno Press, NY, 1979.
- Baker, R. (1833). Report of the Leeds Board of Health. British Library, London: 10347.ee.17(8).
- Ball, W.W.R. (1908). *A Short Account of the History of Mathematics*, 4edn, Macmillan & Co., London. (re-published in 1960, N.Y.: Dover).
- Becker, R.A. (1994). A brief history of S, in P. Dirschedl and R. Ostermann (eds), *Computational Statistics*, Physica Verlag, Heidleberg, pp. 81–110.
- Becker, R.A. and Cleveland, W.S. (1987). Brushing scatterplots, *Technometrics*, 29:127–142.
- Berghaus, H. (1838). *Physikalischer Atlas*, Justus Perthes, Gotha. 2 vols., published 1845–48.
- Bertin, J. (1967). *Sémiologie Graphique: Les diagrammes, les réseaux, les cartes*, Gauthier-Villars, Paris.
- Bishop, Y.M.M., Fienberg, S.E. and Holland, P.W. (1975). *Discrete Multivariate Analysis: Theory and Practice*, MIT Press, Cambridge, MA.
- Bowley, A.L. (1901). *Elements of Statistics*, P.S. King and Son, London.
- Buache, P. (1752). Essai de géographie physique, *Mémoires de l'Académie Royale des Sciences* pp. 399–416. Bibliothèque Nationale de France, Paris (Tolbiac): Ge.FF-8816–8822.

- Chernoff, H. (1973). The use of faces to represent points in k -dimensional space graphically, *Journal of the American Statistical Association*, 68:361–368.
- Cleveland, W.S. and McGill, M.E. (eds) (1988). *Dynamic Graphics for Statistics*, CRC Press, Boca Raton, FL.
- Cohen, A. (1980). On the graphical display of the significant components in a two-way contingency table, *Communications in Statistics – Theory and Methods*, A9:1025–1041.
- Cook, R.D. and Weisberg, S. (1999). *Applied Regression Including Computing and Graphics*, Wiley, New York.
- Costelloe, M.F.P. (1915). Graphic methods and the presentation of this subject to first year college students, *Nebraska Blue Print*.
- Crome, A.F.W. (1785). *Über die Grösse und Bevölkerung der Sämtlichen Europäischen Staaten*, Weygand, Leipzig.
- Dahmann, D.C. (2001). Presenting the nation's cultural geography [in census atlases], At American Memory: Historical Collections for the National Digital Library Internet. <http://memory.loc.gov/ammem/gmdhtml/census.html>.
- de Fourcroy, C. (1782). *Essai d'une table poléométrique, ou amusement d'un amateur de plans sur la grandeur de quelques villes*, Dupain-Triel, Paris.
- de Witt, J. (1671). *The Worth of Life Annuities Compared to Redemption Bonds*, n.p., Leiden.
- du Carla-Boniface, M. (1782). Expression des nivelllements; ou, méthode nouvelle pour marquer sur les cartes terrestres et marines les hauteurs et les configurations du terrain. In François de Dainville, “From the Depths to the Heights,” translated by Arthur H. Robinson, *Surveying and Mapping*, 1970, 30:389–403, on page 396.
- de Nautonier, G. (1602–1604) *Mecometrie de l'eymant, c'est à dire la maniere de mesurer les longitudes par le moyen de l'eymant*, n.p., Paris. British Library, London: 533.k.9.
- Dupin, C. (1826). *Carte figurative de l'instruction populaire de la France*, Jobard. Bibliothèque Nationale de France, Paris (Tolbiac): Ge C 6588 (Funkhouser (1937, p. 300) incorrectly dates this as 1819).
- Eells, W.C. (1926). The relative merits of circles and bars for representing component parts, *Journal of the American Statistical Association*, 21:119–132.
- Farebrother, R.W. (1999). *Fitting Linear Relationships: A History of the Calculus of Observations 1750–1900*, Springer, New York.
- Ferguson, S. (1991). The 1753 carte chronographique of Jacques Barbeu-Dubourg, *Princeton University Library Chronicle*, 52:190–230.
- Fienberg, S.E. (1975). Perspective Canada as a social report, *Technical report*, Department of Applied Statistics, University of Minnesota. unpublished paper.
- Fishkeller, M.A., Friedman, J.H. and Tukey, J.W. (1974). PRIM-9: An interactive multidimensional data display and analysis system, *Technical Report SLAC-PUB-1408*, Stanford Linear Accelerator Center, Stanford, CA.
- Fowlkes, E.B. (1969). User's manual for a system for interactive probability plotting on Graphic-2, *Technical report*, Bell Laboratories.
- Friendly, M. (1994). Mosaic displays for multi-way contingency tables, *Journal of the American Statistical Association*, 89:190–200.

- Friendly, M. (1995). Conceptual and visual models for categorical data, *The American Statistician*, 49:153–160.
- Friendly, M. (1999). Extending mosaic displays: Marginal, conditional, and partial views of categorical data, *Journal of Computational and Graphical Statistics*, 8(3):373–395.
- Friendly, M. (2000). Re-Visions of Minard, *Statistical Computing & Statistical Graphics Newsletter*, 11(1):1, 13–19.
- Friendly, M. (2002). Visions and re-visions of Charles Joseph Minard, *Journal of Educational and Behavioral Statistics*, 27(1):31–52.
- Friendly, M. (2005). Milestones in the history of data visualization: A case study in statistical historiography, in C. Weihs and W. Gaul (eds), *Classification: The Ubiquitous Challenge*, Springer, New York, pp. 34–52.
- Friendly, M. and Denis, D. (2000). The roots and branches of statistical graphics, *Journal de la Société Française de Statistique*, 141(4):51–60. (published in 2001).
- Friendly, M. and Denis, D. (2005). The early origins and development of the scatterplot, *Journal of the History of the Behavioral Sciences*, 41(2):103–130.
- Friendly, M. and Kwan, E. (2003). Effect ordering for data displays, *Computational Statistics and Data Analysis*, 43(4):509–539.
- Friis, H.R. (1974). Statistical cartography in the United States prior to 1870 and the role of Joseph C.G. Kennedy and the U.S. Census Office, *American Cartographer*, 1:131–157.
- Frisius, R.G. (1533). *Libellus de locorum describendorum ratione*, Antwerp.
- Funkhouser, H.G. (1936). A note on a tenth century graph, *Osiris*, 1:260–262.
- Funkhouser, H.G. (1937). Historical development of the graphical representation of statistical data, *Osiris*, 3(1):269–405. reprinted Brugge, Belgium: St. Catherine Press, 1937.
- Gabriel, K.R. (1971). The biplot graphic display of matrices with application to principal components analysis, *Biometrics*, 58(3):453–467.
- Galton, F. (1863). *Meteorographica, or, Methods of Mapping the Weather*, Macmillan, London. British Library, London: Maps.53.b.32.
- Galton, F. (1881). On the construction of isochronic passage charts, *Proceedings of the Geographical Section of the British Association*, n.v.(XI):657, 704. Read Sept. 1, 1881; also published: *Roy. Geog. Soc. Proc.*, 1881, 657–658.
- Galton, F. (1886). Regression towards mediocrity in hereditary stature, *Journal of the Anthropological Institute*, 15:246–263.
- Galton, F. (1894). Results derived from the natality table of Körösi by employing the method of contours or isogens, *Journal of the Royal Statistical Society*, 57(4):702–708.
- Giffen, R. (1899). The excess of imports, *Journal of the Royal Statistical Society*, 62(1):1–82.
- Gilbert, E.W. (1958). Pioneer maps of health and disease in England, *Geographical Journal*, 124:172–183.
- Goodman, L.A. (1970). The multivariate analysis of qualitative data: Interactions among multiple classifications, *Journal of the American Statistical Association*, 65:226–256.

-
- Graunt, J. (1662). *Natural and Political Observations Mentioned in a Following Index and Made Upon the Bills of Mortality*, Martin, Allestry, and Dicas, London.
- Guerry, A.-M. (1833). *Essai sur la statistique morale de la France*, Crochard, Paris. English translation: Hugh P. Whitt and Victor W. Reinking, Lewiston, N.Y.: Edwin Mellen Press, 2002.
- Guerry, A.-M. (1864). *Statistique morale de l'Angleterre comparée avec la statistique morale de la France, d'après les comptes de l'administration de la justice criminelle en Angleterre et en France, etc.*, J.-B. Baillièvre et fils, Paris. Bibliothèque Nationale de France, Paris (Tolbiac): GR FOL-N-319; SG D/4330; British Library, London: Maps 32.e.34; Staatsbibliothek zu Berlin: Fe 8586; Library of Congress: 11005911.
- Haberman, S.J. (1973). The analysis of residuals in cross-classified tables, *Biometrics*, 29:205–220.
- Hald, A. (1990). *A History of Probability and Statistics and their Application before 1750*, John Wiley and Sons, New York.
- Halley, E. (1686). On the height of the mercury in the barometer at different elevations above the surface of the earth, and on the rising and falling of the mercury on the change of weather, *Philosophical Transactions* pp. 104–115.
- Halley, E. (1701). The description and uses of a new, and correct sea-chart of the whole world, shewing variations of the compass, London.
- Hartigan, J.A. and Kleiner, B. (1981). Mosaics for contingency tables, in W.F. Eddy (ed), *Computer Science and Statistics: Proceedings of the 13th Symposium on the Interface*, Springer-Verlag, New York, NY, pp. 268–273.
- Haskell, A.C. (1919). *How to Make and Use Graphic Charts*, Codex, New York.
- Herschel, J.F.W. (1833). On the investigation of the orbits of revolving double stars, *Memoirs of the Royal Astronomical Society*, 5:171–222.
- Hertzsprung, E. (1911). Publikationen des Astrophysikalischen Observatoriums zu Potsdam. Num. 63.
- Hoff, H.E. and Geddes, L.A. (1959). Graphic recording before Carl Ludwig: An historical summary, *Archives Internationales d'Histoire des Sciences*, 12:3–25.
- Hoff, H.E. and Geddes, L.A. (1962). The beginnings of graphic recording, *Isis*, 53:287–324. Pt. 3.
- Howard, L. (1800). On a periodical variation of the barometer, apparently due to the influence of the sun and moon on the atmosphere, *Philosophical Magazine*, 7:355–363.
- Inselberg, A. (1985). The plane with parallel coordinates, *The Visual Computer*, 1:69–91.
- Jevons, W.S. (1863). A serious fall in the value of gold ascertained, and its social effects set fourth, London.
- Jevons, W.S. (1879). Graphical method, *Principles of Science: A Treatise on Logic and Scientific Method*, 3rd edn, Dover, New York, pp. 492–496. First ed.: 1874; page numbers from 3rd Ed. Dover reprint (1958).
- Joint Committee on Standards for Graphic Presentation (1914). Preliminary report published for the purpose of inviting suggestions for the benefit of the committee, *Publications of the American Statistical Association*, 14(112):790–797.

- Karsten, K.G. (1925). *Charts and Graphs. An Introduction to Graphic Methods in the Control and Analysis of Statistics*, Prentice Hall, New York.
- Klein, J.L. (1997). *Statistical Visions in Time: A History of Time Series Analysis, 1662–1938*, Cambridge University Press, Cambridge, UK.
- Kruskal, W. (1977). Visions of maps and graphs, *Proceedings of the International Symposium on Computer-Assisted Cartography, Auto-Carto II*, pp. 27–36. 1975.
- Lallemand, C. (1885). *Les abaques hexagonaux: Nouvelle méthode générale de calcul graphique, avec de nombreux exemples d'application*, Ministère des travaux publics, Comité du nivellation général de la France, Paris.
- Maas, H. and Morgan, M.S. (2005). Timing history: The introduction of graphical analysis in 19th century british economics, *Revue d'Histoire des Sciences Humaines*, 7:97–127.
- Marey, E.-J. (1878). *La méthode graphique dans les sciences expérimentales et principalement en physiologie et en médecine*, G. Masson, Paris.
- Marshall, A. (1885). On the graphic method of statistics, *Journal of the Royal Statistical Society, Jubille volume*, 251–260.
- Maunder, E.W. (1904). Note on the distribution of sun-spots in heliographic latitude, 1874 to 1902, *Royal Astronomical Society Monthly Notices*, 64:747–761.
- Meyer, D., Zeileis, A. and Hornik, K. (2005). *vcd: Visualizing Categorical Data*. R package version 0.9–5.
- Minard, C.J. (1826). *Projet de canal et de chemins de fer pour le transport de pavés à Paris, précédé d'un tableau des progrès de la dépense du pavé de Paris pendant les deux derniers siècles*, A. Pihan Delaforest, Paris. École Nationale des Ponts et Chaussées, Paris: 4.5065/C289.
- Minard, C.J. (1844). Tableaux figuratifs de la circulation de quelques chemins de fer, lith. (n.s.). École Nationale des Ponts et Chaussées, Paris: 5860/C351, 5299/C307.
- Minard, C.J. (1861). *Des tableaux graphiques et des cartes figuratives*, E. Thunot et Cie, Paris. École Nationale des Ponts et Chaussées, Paris: 3386/C161; Bibliothèque Nationale de France, Paris (Tolbiac): V-16168.
- Moseley, H. (1913). The high frequency spectra of the elements, *Philosophical Magazine* pp. 1024. (continued, 1914, pp. 703).
- Mouat, F.J. (1885). History of the Statistical Society of London, *Journal of the Royal Statistical Society, Jubille volume*, 14–62.
- Nightingale, F. (1857). *Mortality of the British Army*, Harrison and Sons, London.
- Oresme, N. (1482). *Tractatus de latitudinibus formarum*, Padova. British Library, London: IA 3Q024.
- Oresme, N. (1968). *Nicole Oresme and the Medieval Geometry of Qualities and Motions: A Treatise on the Uniformity and Difformity Known as Tractatus de configurationibus qualitatum et motuum*, University of Wisconsin Press, Madison WI. tr.: M. Clagget.
- Palsky, G. (1996). *Des chiffres et des cartes: naissance et développement de la cartographie quantitative française au XIX^e siècle*, Comité des Travaux Historiques et Scientifiques (CTHS), Paris.

- Pearson, E.S. (ed) (1978). *The History of Statistics in the 17th and 18th Centuries Against the Changing Background of Intellectual, Scientific and Religious Thought*, Griffin & Co. Ltd., London. Lectures by Karl Pearson given at University College London during the academic sessions 1921–1933.
- Pearson, K. (1914–1930). *The life, letters and labours of Francis Galton*, Cambridge: University Press.
- Pearson, K. (1920). Notes on the history of correlation, *Biometrika*, 13(1):25–45.
- Peddle, J.B. (1910). *The Construction of Graphical Charts*, McGraw-Hill, New York.
- Perozzo, L. (1880). Della rappresentazione grafica di una collettività di individui nella successione del tempo, *Annali di Statistica*, 12:1–16. British Library, London: S.22.
- Petty, W. (1665). *The economic writings of Sir William Petty: together with the observations upon the bills of mortality*, The University Press, Cambridge. C.H. Hall (ed) (more probably by Captain John Graunt).
- Playfair, W. (1786). *Commercial and Political Atlas: Representing, by Copper-Plate Charts, the Progress of the Commerce, Revenues, Expenditure, and Debts of England, during the Whole of the Eighteenth Century*, Corry, London. Re-published in Wainer, H. and Spence, I. (eds), *The Commercial and Political Atlas and Statistical Breviary*, 2005, Cambridge University Press, ISBN 0-521-85554-3.
- Playfair, W. (1801). *Statistical Breviary; Shewing, on a Principle Entirely New, the Resources of Every State and Kingdom in Europe*, Wallis, London. Re-published in Wainer, H. and Spence, I. (eds), *The Commercial and Political Atlas and Statistical Breviary*, 2005, Cambridge University Press, ISBN 0-521-85554-3.
- Playfair, W. (1821). Letter on our agricultural distresses, their causes and remedies; accompanied with tables and copperplate charts shewing and comparing the prices of wheat, bread and labour, from 1565 to 1821. British Library, London: 8275.c.64.
- Porter, T.M. (1986). *The Rise of Statistical Thinking 1820–1900*, Princeton University Press, Princeton, NJ.
- Priestley, J. (1765). A chart of biography, London. British Library, London: 611.I.19.
- Priestley, J. (1769). *A New Chart of History*, Thomas Jeffreys, London. British Library, London: Cup.1250.e.18 (1753).
- Quetelet, A. (1831). *Recherches sur le penchant au crime aux différens âges*, Hayez, Brussels. English translation by Sawyer F. Sylvester. Cincinnati, OH: Anderson Publishing Company, 1984.
- Quetelet, A. (1835). *Sur l'homme et le développement de ses facultés, ou Essai de physique sociale*, Bachelier, Paris. English translation, by R. Knox and T. Smilbert, *A Treatise on Man and the Development of his Faculties*. Edinburgh, 1842; New York: Burt Franklin, 1968.
- Riddell, R.C. (1980). Parameter disposition in pre-Newtonian planetary theories, *Archives Hist. Exact Sci.*, 23:87–157.
- Riedwyl, H. and Schüpbach, M. (1983). Siebdiagramme: Graphische Darstellung von Kontingenztafeln, *Technical Report 12*, Institute for Mathematical Statistics, University of Bern, Bern, Switzerland.
- Robinson, A.H. (1982). *Early Thematic Mapping in the History of Cartography*, University of Chicago Press, Chicago.

- Royston, E. (1970). Studies in the history of probability and statistics, III. a note on the history of the graphical presentation of data, *Biometrika* pp. 241–247. 43, Pts. 3 and 4 (December 1956); reprinted In *Studies in the History Of Statistics and Probability Theory*, eds. E.S. Pearson and M.G. Kendall, London: Griffin.
- Scheiner, C. (1626–1630) *Rosa ursina sive sol ex admirando facularum & macularum suarum phoenomeno varius*, Andream Phaeum, Bracciano, Italy. British Library, London: 532.l.6.
- Smith, W. (1815). *A delineation of the strata of England and Wales, with part of Scotland; exhibiting the collieries and mines, the marshes and fenlands originally overflowed by the sea, and the varieties of soil according to the substrata, illustrated by the most descriptive names*, John Cary, London. British Library, London: Maps 1180.(19).
- Snow, J. (1855). *On the Mode of Communication of Cholera*, 2 edn, (n.p.), London.
- Sobel, D. (1996). *Longitude: The True Story of a Lone Genius Who Solved the Greatest Scientific Problem of His Time*, Penguin, New York.
- Spence, I. and Garrison, R.F. (1993). A remarkable scatterplot, *The American Statistician*, 47(1):12–19.
- Stigler, S.M. (1986). *The History of Statistics: The Measurement of Uncertainty before 1900*, Harvard University Press, Cambridge, MA.
- Tartaglia, N.F. (1556). *General Trattato di Numeri et Misure*, Vinegia, Venice. British Library, London: 531.n.7–9; 47.e.4.
- Tierney, L. (1990). *LISP-STAT: An Object-Oriented Environment for Statistical Computing and Dynamic Graphics*, John Wiley and Sons, New York.
- Tilling, L. (1975). Early experimental graphs, *British Journal for the History of Science*, 8:193–213.
- Tufte, E.R. (1983). *The Visual Display of Quantitative Information*, Graphics Press, Cheshire, CT.
- Tufte, E.R. (1997). *Visual Explanations*, Graphics Press, Cheshire, CT.
- Tukey, J.W. (1962). The future of data analysis, *Annals of Mathematical Statistics*, 33:1–67 and 81.
- Tukey, J.W. (1977). *Exploratory Data Analysis*, Addison-Wesley, Reading, MA.
- Tukey, P.A. and Tukey, J.W. (1981). Graphical display of data sets in 3 or more dimensions, in V. Barnett (ed), *Interpreting Multivariate Data*, Wiley and Sons, Chichester, U.K.
- Vauthier, L.L. (1874). Note sur une carte statistique figurant la répartition de la population de Paris, *Comptes Rendus des Séances de l'Académie des Sciences*, 78:264–267. École Nationale des Ponts et Chaussées, Paris: 11176 C612.
- von Huhn, R. (1927). A discussion of the Eells' experiment, *Journal of the American Statistical Association*, 22:31–36.
- von Humboldt, A. (1817). Sur les lignes isothermes, *Annales de Chimie et de Physique*, 5:102–112.
- Wainer, H. (2005). *Graphic Discovery: A Trout in the Milk and Other Visual Adventures*, Princeton University Press, Princeton, NJ.
- Wainer, H. and Velleman, P.F. (2001). Statistical graphics: Mapping the pathways of science, *Annual Review of Psychology*, 52:305–335.

-
- Wallis, H.M. and Robinson, A.H. (1987). *Cartographical Innovations: An International Handbook of Mapping Terms to 1900*, Map Collector Publications, Tring, Herts.
- Warne, F.J. (1916). *Warne's Book of Charts, A Special Feature of Warne's Elementary Course in Chartography*, F.J. Warne, Washington, D.C. 3 p. l., 106 charts. 31×41 cm.
- Washburne, J.N. (1927). An experimental study of various graphic, tabular and textual methods of presenting quantitative material, *Journal of Educational Psychology*, 18:361–376, 465–476.
- Wegman, E.J. (1990). Hyperdimensional data analysis using parallel coordinates, *Journal of the American Statistical Association*, 85(411):664–675.
- Westergaard, H. (1932). *Contributions to the History of Statistics*, P.S. King & Son, London.
- Whitt, H. (2002). Inventing sociology: André-Michel Guerry and the *Essai sur la statistique morale de la France*, Edwin Mellen Press, Lewiston, NY, pp. ix–xxxvii. English translation: Hugh P. Whitt and Victor W. Reinking, Lewiston, N.Y. : Edwin Mellen Press, 2002.
- Young, F.W. (1994a). ViSta: The visual statistics system, *Technical Report RM 94-1*, L.L. Thurstone Psychometric Laboratory, UNC.
- Young, F.W. (1994b). ViSta: The visual statistics system, *Technical Report RM 94-1*, L.L. Thurstone Psychometric Laboratory, UNC.
- Young, F.W., Valero-Mora, P. and Friendly, M. (2006). *Visual Statistics: Seeing Data with Dynamic Interactive Graphics*, Wiley, New York.
- Zeuner, G. (1869). Abhandlungen aus der mathematischen Statistik, Leipzig. British Library, London: 8529.f.12.

Good Graphics?

II.2

Antony Unwin

2.1	<i>Introduction</i>	58
	Content, Context and Construction	58
	Presentation Graphics and Exploratory Graphics	59
2.2	<i>Background</i>	60
	History	60
	Literature	61
	The Media and Graphics	62
2.3	<i>Presentation (What to Whom, How and Why)</i>	62
2.4	<i>Scientific Design Choices in Data Visualization</i>	63
	Choice of Graphical Form	64
	Graphical Display Options	64
2.5	<i>Higher-dimensional Displays and Special Structures</i>	70
	Scatterplot Matrices (Sploms)	70
	Parallel Coordinates	70
	Mosaic Plots	71
	Small Multiples and Trellis Displays	72
	Time Series and Maps	74
2.6	<i>Practical Advice</i>	76
	Software	76
	Bad Practice and Good Practice (Principles)	77
2.7	<i>And Finally</i>	77

Graphical excellence is nearly always multivariate.

– Edward Tufte

2.1

Introduction

This chapter discusses drawing good graphics to visualize the information in data. Graphics have been used for a long time to present data. Figure 2.1 is a scanned image from Playfair's *Commercial and Political Atlas* of 1801, reproduced in Playfair (2005). The fairly continuous increase of both imports and exports, and the fact that the balance was in favour of England from 1720 on, can be seen easily. Some improvements might be made, but overall the display is effective and well drawn.

Data graphics are used extensively in scientific publications, in newspapers and in the media generally. Many of those graphics do not fully convey the information in the data they are supposed to be presenting and may even obscure it. What makes a graphic display of data bad? More importantly, what makes one good? In any successful graphic there must be an effective blending of content, context, construction and design.

2.1.1

Content, Context and Construction

What is plotted comes first, and without content no amount of clever design can bring meaning to a display. A good graphic will convey information, but a graphic is always part of a larger whole, the context, which provides its relevance. So a good graphic will complement other related material and fit in, both in terms of content and also

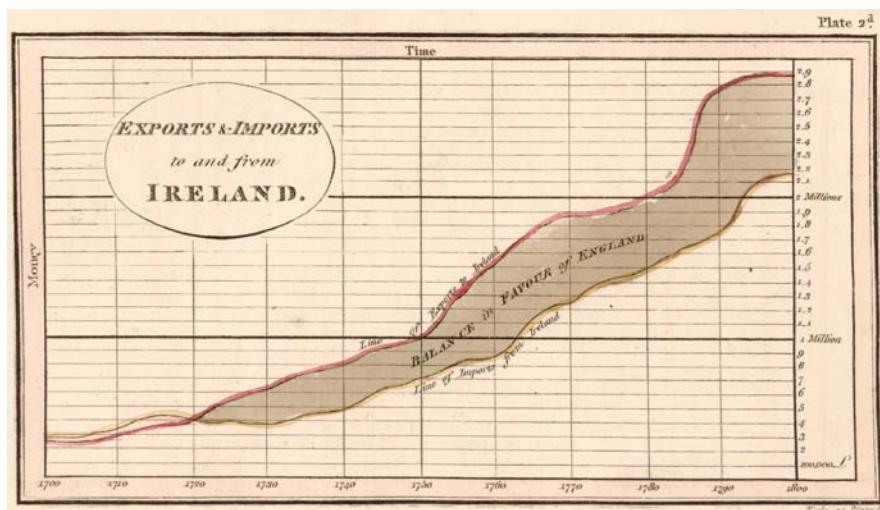


Figure 2.1. Playfair's chart of trade between England and Ireland from 1700 to 1800

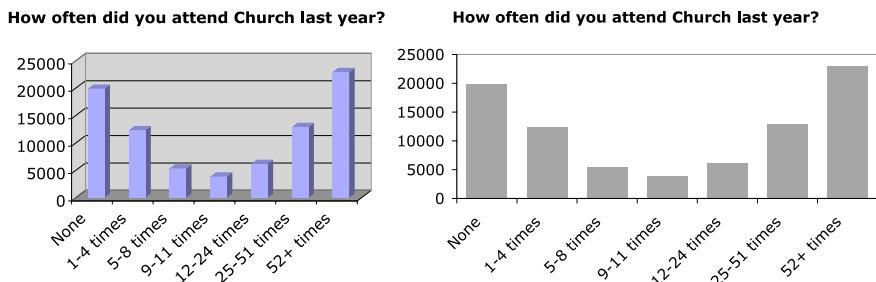


Figure 2.2. Church attendance (DDB Life Style Survey 1975–1998)

with respect to style and layout. Finally, if a graphic is constructed and drawn well, it will look good.

Figure 2.2 shows two similar displays of the same data from the DDB social survey used in Robert Putnam's book *Bowling Alone* (Putnam, 2000). Every year for 24 years, different groups of 3000 people were surveyed. Amongst other questions, they were asked how often they had attended church in the last year.

The left-hand graph includes gridlines and a coloured background and uses 3-D columns to represent the data counts. The right-hand graph sticks to basics. In general, the right-hand display is to be preferred (3-D columns can cross gridlines, and zero values would be misleadingly displayed). For these data there is not much to choose between the two representations; both convey the same overall information. The potential weakness in both graphics is the set of categories. Grouping the data together in different ways could give quite different impressions.

For a given dataset there is not a great deal of advice which can be given on content and context. Those who know their own data should know best for their specific purposes. It is advisable to think hard about what should be shown and to check with others if the graphic makes the desired impression. Design should be left to designers, though some basic guidelines should be followed: consistency is important (sets of graphics should be in similar style and use equivalent scaling); proximity is helpful (place graphics on the same page, or on the facing page, of any text that refers to them); and layout should be checked (graphics should be neither too small nor too large and be attractively positioned relative to the whole page or display). Neither content nor context nor design receives much attention in books offering advice on data graphics; quite properly they concentrate on construction. This chapter will, too.

Presentation Graphics and Exploratory Graphics

2.1.2

There are two main reasons for using graphic displays of datasets: either to present or to explore data. Presenting data involves deciding what information you want to convey and drawing a display appropriate for the content and for the intended audience. You have to think about how the plot might be perceived and whether it will be understood as you wish. Plots which are common in one kind of publication may be unfamiliar to the readers of another. There may only be space for one plot and it may

be available in print for a very long time, so great care should be taken in preparing the most appropriate display. Exploring data is a much more individual matter, using graphics to find information and to generate ideas. Many displays may be drawn. They can be changed at will or discarded and new versions prepared, so generally no one plot is especially important, and they all have a short life span. Clearly principles and guidelines for good presentation graphics have a role to play in exploratory graphics, but personal taste and individual working style also play important roles. The same data may be presented in many alternative ways, and taste and customs differ as to what is regarded as a good presentation graphic. Nevertheless, there are principles that should be respected and guidelines that are generally worth following. No one should expect a perfect consensus where graphics are concerned.

2.2

Background

2.2.1

History

Data graphics may be found going very far back in history, but most experts agree that they really began with the work of Playfair a little more than 200 years ago. He introduced some modern basic plots (including the barchart and the histogram) and produced pertinent and eye-catching displays (Fig. 2.1). Wainer and Spence recently republished a collection of his works (Playfair, 2005). Not all his graphics could be described as good, but most were. In the second half of the 19th century Minard prepared impressive graphics, including his famous chart of Napoleon's advance on and retreat from Moscow. The French Ministry of Public Works used his ideas to attractive, and presumably pertinent, effect in an annual series of publications (*Album de Statistique Graphique*) from 1879 to 1899, presenting economic data geographically for France. Examples can be found in Michael Friendly's chapter in this book.

In the first half of the last century graphics were not used in statistics as much as they might have been. Interestingly, the second chapter in Fisher's *Statistical Methods for Research Workers* in 1925 was on diagrams for data, so he, at least, thought graphics important. In Vienna there was a group led by Otto Neurath which worked extensively on pictograms in the 1920s and early 1930s. They produced some well-crafted displays, which were forerunners of the modern infographics. (Whether Fig. 2.3 is improved by including the symbols at the top to represent the USA is a matter of taste.)

With the advent of computers, graphics went into a relative decline. Computers were initially bad for graphics for two reasons. Firstly, much more complex analytic models could be evaluated and, quite naturally, modelling received a great deal more attention than displaying data. Secondly, only simple and rather ugly graphics could be drawn by early computers. The development of hardware and software has turned all this around. In recent years it has been very easy to produce graphics, and far more can be seen than before. Which is, of course, all the more reason to be concerned that graphics be drawn well.

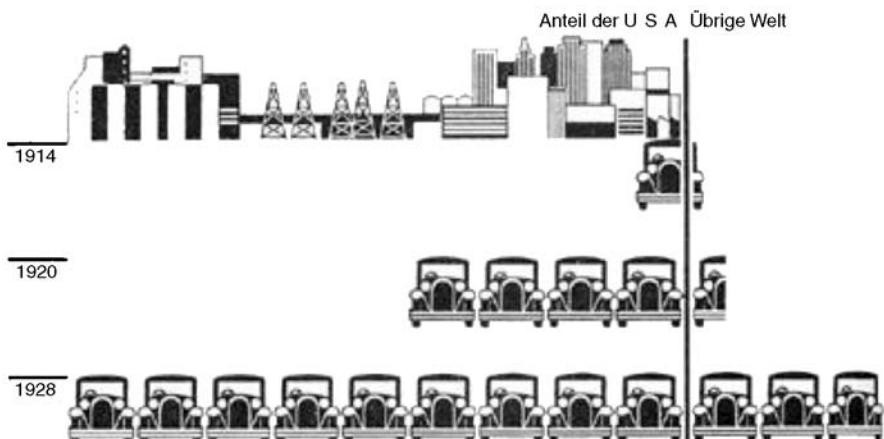


Figure 2.3. Pictogram by Otto Neurath of the number of cars in the USA and the rest of the world in 1914, 1920, and 1928

Literature

2.2.2

Several authors have written excellent books on drawing good statistical graphics, the best known, justifiably, being Edward Tufte. His books (e.g. Tufte, 2001) include many splendid examples (and a few dreadful ones) and describe important principles on how to draw good graphics. Tufte criticizes unsuitable decoration and data misrepresentation, but his advice is restricted to representing data properly. Cleveland's books [for instance, Cleveland (1994) another useful source of advice on preparing data displays] are equally valuable. And this is the way it should be. Statisticians should concentrate on getting the basic statistical display right, and designers may be consulted to produce a polished final version.

While there is a place for applied books full of sound practical advice [other useful references include Burn (1993), Kosslyn (1994), and Robbins (2004)], there is also a need for theory to provide formal structures for understanding practice and to provide a foundation from which progress can be made. Graphics must be one of the few areas in statistics where there is little such theory. Bertin's major work (*Semiology Graphique*) from 1973 contains a number of interesting ideas and is often cited, but it is difficult to point to later work that directly extends it. Wilkinson's *Grammar of Graphics* has received a lot of attention and been quickly revised in a substantially expanded second edition (Wilkinson, 2005).

If there is little theory, then examples become particularly important to show what can be achieved. The two books by Wainer (1997, 2004) contain collections of columns first published in Chance and offer instructive and entertaining examples. Friendly's Gallery of Statistical Visualization (<http://www.math.yorku.ca/SCS/Gallery/>) includes many examples, both good and bad, chronicling the history of graphical developments. The websites ASK E.T. (<http://www.edwardtufte.com>) and Junk Charts (<http://junkcharts.typepad.com>) provide lively discussions and sage advice for par-

ticular examples. It would be invidious, and perhaps unfair, to single out egregious examples here. Readers should be able to find plenty for themselves without having to look far.

2.2.3

The Media and Graphics

Graphical displays of data appear in the press very often. They are a subset of infographics, graphical displays for conveying information of any kind, usually discussed under the heading information visualization (Spence, 2001). Many impressive examples can be found in the New York Times. While there are guidelines which apply to all infographics, this chapter is restricted to the construction of data visualizations.

Data displays in the media are used to present summary information, such as the results of political surveys (what proportion of the people support which party), the development of financial measures over time (a country's trade balance or stock market indices) or comparisons between population groups (average education levels of different sections of the community). There are many other examples. These topics only require fairly basic displays, so it is not surprising that in the media they are commonly embellished with all manner of decoration and ornamentation, sometimes effectively drawing attention both to the graphic and to its subject, sometimes just making it more difficult to interpret the information being presented. What is surprising is that the graphics are often misleading or flawed.

Presentation (What to Whom, How and Why)

How is it possible to make a mess of presenting simple statistical information? Surely there is little that can go wrong. It is astonishing just what distortion can be introduced: misleading scales may be employed; 3-D displays of 2-D data make it difficult to make fair comparisons; areas which are apparently intended to be proportional to values are not; so much information is crammed into a small space that nothing can be distinguished. While these are some of the technical problems that can arise, there are additional semantic ones. A graphic may be linked to three pieces of text: its caption, a headline and an article it accompanies. Ideally, all three should be consistent and complement each other. In extreme cases all four can tell a different story! A statistician cannot do much about headlines (possibly added or amended by a subeditor at the last minute) or about accompanying articles if he or she is not the first author (in the press the journalist chooses the graphic and may have little time to find something appropriate), but the caption and the graphic itself should be “good”.

Some displays in the media highlight a news item or provide an illustration to lighten the text. These are often prepared by independent companies at short notice and sold to the media as finished products. Fitting the graphic to its context may be

awkward. There are displays in scientific publications which are prepared by the authors and should be the product of careful and thorough preparation. In this situation a graphic should match its context well. Whatever kind of data graphic is produced, a number of general principles should be followed to ensure that the graphic is at least correct.

Whether or not a graphic is then successful as a display depends on its subject, on its context and on aesthetic considerations. It depends on what it is supposed to show, on what form is chosen for it and on its audience. Readers familiar with one kind of graphic will have no trouble interpreting another example of the same kind. On the other hand, a graphic in a form which is new to readers may lead to unanticipated interpretation difficulties. When someone has spent a long time on a study and further time on the careful preparation of a graphic display to illustrate the conclusions, they are usually astonished when others do not see what they can see. [This effect is, of course, not restricted to drawing graphics. Designers are frequently shocked by how people initially misunderstand their products. How often have you stared at the shower in a strange hotel wondering how you can get it to work without its scalding or freezing you? Donald Norman's book (Norman, 1988) is filled with excellent examples.]

Other factors have to be considered as well. A graphic may look different in print than on a computer screen. Complex graphics may work successfully in scientific articles where the reader takes time to fully understand them. They will not work well as a brief item in a television news programme. On the other hand, graphics which are explained by a commentator are different from graphics in print. If graphics displayed on the Web can be queried (as with some of the maps on <http://www3.cancer.gov/atlasplus/>, discussed in Sect. 2.5.5), then more information can be provided without cluttering the display.

Scientific Design Choices in Data Visualization

2.4

Plotting a single variable should be fairly easy. The type of variable will influence the type of graphic chosen. For instance, histograms or boxplots are right for continuous variables, while barcharts or piecharts are appropriate for categorical variables. In both cases other choices are possible too. Whether the data should be transformed or aggregated will depend on the distribution of the data and the goal of the graphic. Scaling and captioning should be relatively straightforward, though they need to be chosen with care.

It is a different matter with multivariate graphics, where even displaying the joint distribution of two categorical variables is not simple. The main decision to be taken for a multivariate graphic is the form of display, though the choice of variables and their ordering are also important. In general a dependent variable should be plotted last. In a scatterplot it is traditional to plot the dependent variable on the vertical axis.

2.4.1

Choice of Graphical Form

There are barcharts, piecharts, histograms, dotplots, boxplots, scatterplots, roseplots, mosaicplots and many other kinds of data display. The choice depends on the type of data to be displayed (e.g. univariate continuous data cannot be displayed in a piechart and bivariate categorical data cannot be displayed in a boxplot) and on what is to be shown (e.g. piecharts are good for displaying shares for a small number of categories and boxplots are good for emphasizing outliers). A poor choice graph type cannot be rectified by other means, so it is important to get it right at the start. However, there is not always a unique optimal choice and alternatives can be equally good or good in different ways, emphasizing different aspects of the same data.

Provided an appropriate form has been chosen, there are many options to consider. Simply adopting the default of whatever computer software is being used is unlikely to be wise.

2.4.2

Graphical Display Options

Scales

Defining the scale for the axis for a categorical variable is a matter of choosing an informative ordering. This may depend on what the categories represent or on their relative sizes. For a continuous variable it is more difficult. The endpoints, divisions and tick marks have to be chosen. Initially it is surprising when apparently reliable software produces a really bad scale for some variable. It seems obvious what the scale should have been. It is only when you start trying to design your own algorithm for automatically determining scales that you discover how difficult the task is.

In *Grammar of Graphics* Wilkinson puts forward some plausible properties that ‘nice’ scales should possess and suggests a possible algorithm. The properties (simplicity, granularity and coverage, with the bonus of being called ‘really nice’ if zero is included) are good but the algorithm is easy to outwit. This is not to say that it is a weak algorithm. What is needed is a method which gives acceptable results for as high a percentage of the time as possible, and the user must also check the resulting scale and be prepared to amend it for his or her data. Difficult cases for scaling algorithms arise when data cross natural boundaries, e.g., data with a range of 4 to 95 would be easy to scale, whereas data with a range of 4 to 101 would be more awkward.

There is a temptation to choose scales running from the minimum to the maximum of the data, but this means that some points are right on the boundaries and may be obscured by the axes. Unless the limits are set by the meaning of the data (e.g. with exam marks from 0 to 100, neither negative marks nor marks more than 100 are possible – usually!), it is good practice to extend the scales beyond the observed limits and to use readily understandable rounded values. There is no obligatory requirement to include zero in a scale, but there should always be a reason for not doing so; otherwise it makes the reader wonder if some deception is being practiced. Zero is in fact not the only possible baseline or alignment point for a scale, though it is the most common one. A sensible alignment value for ratios is one, and financial series

are often standardized to all start at 100. In Fig. 2.11 the cumulative times for all the riders who finished the Tour de France cycle race in 2004 are plotted. The data at the end of each stage have been aligned at their means. The interest lies in the differences in times between the riders, not so much in their absolute times.

Figure 2.4 shows histograms for the Hidalgo stamp thickness data (Izenman and Sommer, 1988). The first uses default settings and shows a skew distribution with possibly a second mode around 0.10. The second has rounded endpoints and a rounded binwidth and shows stronger evidence for the second mode. The third is drawn so that each distinct value is in a different bin (the data were all recorded to a thousandth of a millimetre). It suggests that the first mode is actually made of up to two groups and that there may be evidence for several additional modes to the right. It also reveals that rounded values such as 0.07, 0.08, . . . , 0.11 occur relatively more frequently. Izenman and Sommer used the third histogram in their paper. What the data repre-

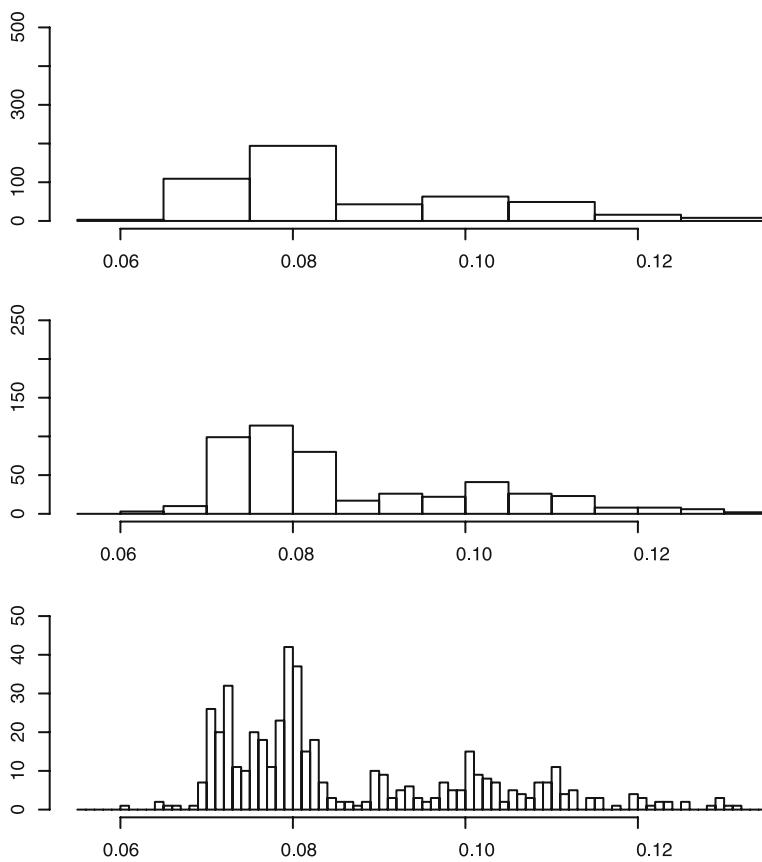


Figure 2.4. Three different histograms of the Hidalgo stamp thickness data, all with the same anchorpoint but with different binwidths. The *horizontal scales* are aligned and the total area of each display is the same (note the different frequency scales). Source: Izenman and Sommer (1988)

sent and how they are collected should be taken into account when choosing scales. Asymptotic optimality criteria only have a minor role to play.

While Fig. 2.4 shows the importance of choosing binwidths carefully, it also illustrates some display issues. The horizontal value axis is clearly scaled, but it would surely be nicer if it extended further to the right. More importantly, the comparison in Fig. 2.4 ideally requires that all three plots be aligned exactly and have the same total area. Not all software provides these capabilities.

Graphics should be considered in their context. It may be better to use a scale in one graphic that is directly comparable with that in another graphic instead of individually scaling both. Common scaling is used in one form or another in Figs. 2.11, 2.13 and 2.14.

It is one thing to determine what scale to use, but quite another to draw and label the axes. Too many labels make a cluttered impression; too few can make it hard for the reader to assess values and differences. (Note that it is not the aim of graphics to provide exact case values; tables are much better for that.) Tick marks in between labels often look fussy and have little practical value. In some really bad situations, they can obscure data points.

Sorting and Ordering

The effect of a display can be influenced by many factors. When more than one variable is to be plotted, the position or order in which they appear in the graphic makes a difference. Examples arise with parallel coordinate plots, mosaicplots and matrix visualizations, all discussed in other chapters. Within a nominal variable with no natural ordering, the order in which the categories are plotted can have a big effect. Alphabetic ordering may be appropriate (a standard default, which is useful for comparison purposes), or a geographic or other grouping (e.g. shares by market sector) might be relevant. The categories could be ordered by size or by a secondary variable. Figure 2.5 shows two barcharts of the same data, the numbers in each class and in the crew on the Titanic. The second ordering would be the same in any language, but the first would vary (for instance, Crew, First, Second, Third in English).

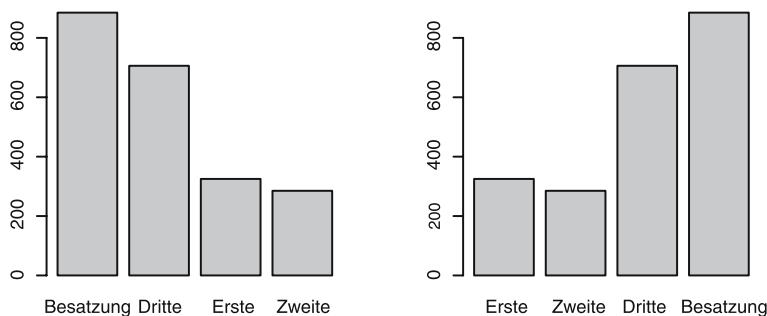


Figure 2.5. Numbers of passengers and crew who travelled on the Titanic, by class, ordered alphabetically (in German) and by status. Source: Dawson (1995)

Adding Model or Statistical Information – Overlaying (Statistical) Information

Guides may be drawn on a plot as a form of annotation and are useful for emphasizing particular issues, say which values are positive or negative. Sloping guides highlight deviations from linearity. Fitted lines, for instance polynomial regression or smoothers, may be superimposed on data not only to show the hypothesized overall structure but also to highlight local variability and any lack of fit. Figure 2.6 plots the times from the first and last stages of a 100-km road race. A lowess smoother has been drawn. It suggests that there is a linear relationship for the faster runners and a flat one for the slower ones.

When multiple measurements are available, it is standard practice in scientific journals to plot point estimates with their corresponding confidence intervals. (95 % confidence intervals are most common, though it is wise to check precisely what has been plotted.) Figure 2.7 displays the results of a study on the deterioration of a thin plastic over time. Measurements could only be made by destructive testing, so all measurements are of independent samples. The high variability at most of the time points is surprising. Adjacent means have been joined by straight lines. A smoothing function would be a better alternative, but such functions are not common for this kind of plot. As the measurement timepoints are far apart and as there is only one dataset, there is no overlapping here. That can very often be a serious problem.

Overlaying information, whether guides or annotation, can lead to overlapping and cluttered displays. Good solutions are possible but may require individual adjustments depending on the shape of the data. A well-spaced and informative display at one size may appear unsatisfactory and unclear when shrunk for publication.

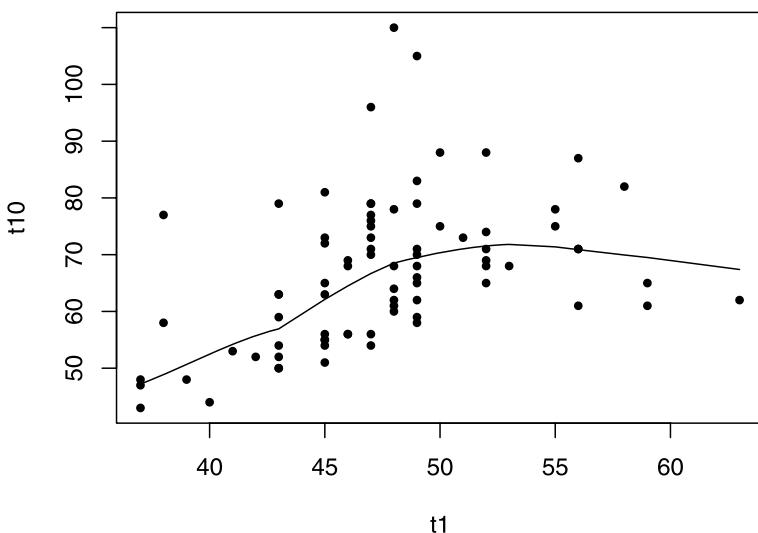


Figure 2.6. Times for 80 runners for the last stage of a road race vs. their times for the first stage, with a lowess smoother. Default scales from R have been used. Source: Everitt (1993)

Captions, Legends and Annotations

Ideally, captions should fully explain the graphic they accompany, including giving the source for the data. Relying on explanations in the surrounding text rarely works. Ideals cannot always be met and very long captions are likely to put off the reader, but the whole point of a graphic is to present information concisely and directly. A compromise where the caption outlines the information in the graphic and a more detailed description can be found in the text can be a pragmatic solution. Graphics which require extensive commentary may be trying to present too much information at one go.

Legends describe which symbols and/or colours refer to which data groups. Tufte recommends that this information be directly on the plot and not in a separate legend, so that the reader's eyes do not have to jump backwards and forwards. If it can be done, it should be.

Annotations are used to highlight particular features of a graphic. For reasons of space there cannot be many of them and they should be used sparingly. They are useful for identifying events in time series, as Playfair did (Playfair, 2005), or for drawing attention to particular points in scatterplots.

Union estimates of protest turnout in Fig. 2.8 are larger than the police estimates by roughly the same factor, except for the two extreme exceptions, Marseille and Paris, where the disagreement is much greater.

Positioning in Text

Keeping graphics and text on the same page or on facing pages is valuable for practical reasons. It is inconvenient to have to turn pages back and forth because graphics and the text relating to them are on different pages. However, it is not always possible to avoid this. Where graphics are placed on a given page is a design issue.

Size, Frames and Aspect Ratio

Graphics should be large enough for the reader to see the information in them clearly and not much larger. This is a rough guideline, as much will depend on the surrounding layout. Frames may be drawn to surround graphics. As frames take up space and add to the clutter, they should best only be used for purposes of separation, i.e. separating the graphic from other graphics or from the text.

Aspect ratios have a surprisingly strong effect on the perception of graphics. This is especially true of time series. If you want to show gradual change, grow the horizontal axis and shrink the vertical axis. The opposite actions will demonstrate dramatic change. For a scatterplot example, see Fig. 2.9, which displays the same data as Fig. 2.6. There is useful advice on aspect ratios in Cleveland (1994), especially the idea of 'banking to 45 degrees' for straight lines.

Colour

Colour should really have been discussed much earlier. It is potentially one of the most effective ways of displaying data. In practice it is also one of the most difficult to get right. A helpful check for colour schemes for maps, Colorbrewer by Cynthia

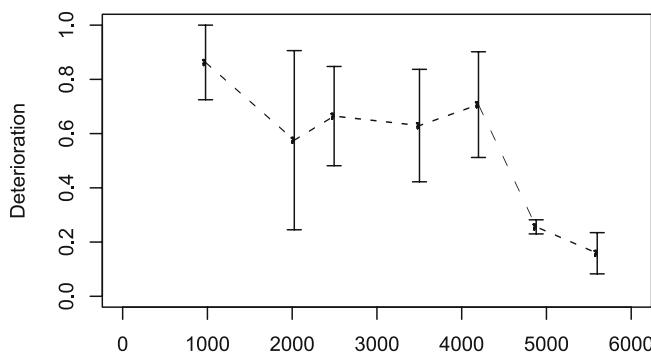


Figure 2.7. Average chemical deterioration and 95 % confidence intervals, measured at different time points. There were either 5 or 10 measurements at each time point. Source: Confidential research data

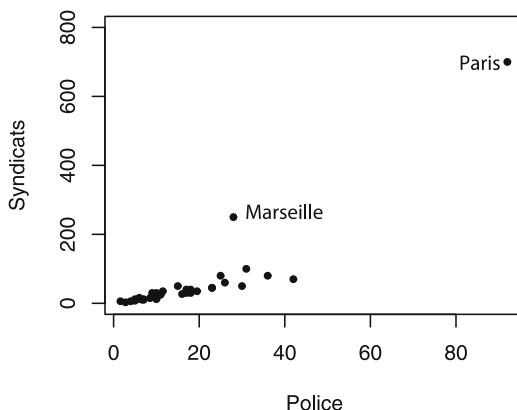


Figure 2.8. Union estimates (in thousands) of the protester turnout in various French cities in the spring of 2006 plotted against police estimates (also in thousands). Source: Le Monde 28.3.06

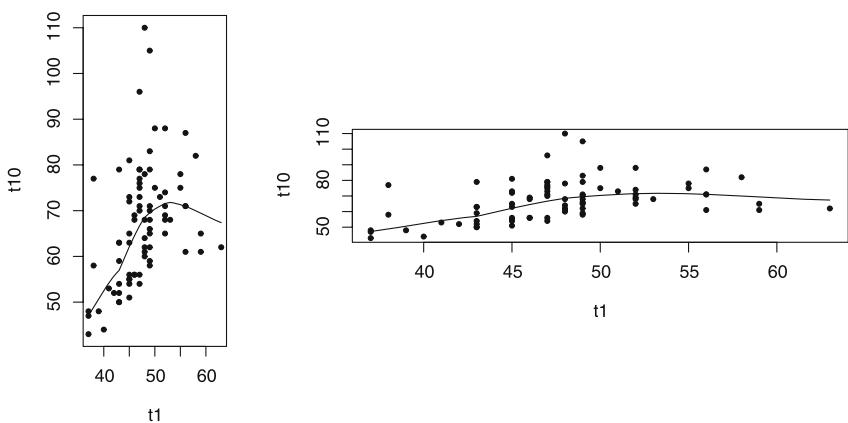


Figure 2.9. The same plot as Fig. 2.6, drawn with different aspect ratios. Data are times for runners for the last stage of a road race vs. their times for the first stage

Brewer, can be found at <http://colorbrewer.org>. Colorbrewer can give suggestions for colour schemes that both blend well and distinguish between different categories.

There remain many factors in the choice of colour which have to be borne in mind: some people are colour blind; colours have particular associations (red for danger or for losses); colours may not be reproduced in print the way they were intended; and colour can be a matter of personal taste. Colour is discussed in more detail in other Handbook chapters.

2.5 Higher-dimensional Displays and Special Structures

2.5.1 Scatterplot Matrices (Sploms)

Plotting each continuous variable against every other one is effective for small numbers of variables, giving an overview of possible bivariate results. Figure 2.10 displays data from emissions tests of 381 cars sold in Germany. It reveals that engine size, performance and fuel consumption are approximately linearly related, as might be expected, that CO₂ measurements and fuel consumption are negatively correlated in batches, which might not be so expected, and that other associations are less conclusive. Packing so many plots into a small space it is important to cut down on scales. Placing the variable names on the diagonal works well, and histograms of the individual variables could also be placed there.

2.5.2 Parallel Coordinates

Parallel coordinate plots (Inselberg, 1999) are valuable for displaying large numbers of continuous variables simultaneously. Showing so much information at once has several implications: not all information will be visible in any one plot (so that several may be needed); formatting and scaling will have a big influence on what can be seen (so that there are many choices to be made); and some overlapping is inevitable (so that α -blending or more sophisticated density estimation methods are useful).

Figure 2.11 plots the cumulative times of the 147 cyclists at the ends of the 21 stages of the 2004 Tour de France. The axes all have the same scale, so that differences are comparable. The best riders take the shortest time and are at the bottom of the plot. The axes have been aligned at their means, as without some kind of alignment little could be seen. α -blending has been applied to reduce the overprinting in the early sprint stages where all riders had almost the same times. If more α -blending is used, then the individual lines for the riders in the later stages of the race become too faint. This single display conveys a great deal about the race. In the early stages, at most a few minutes separates the riders. On the mountain stages there are much larger differences and individual riders gain both time and places (where a line segment

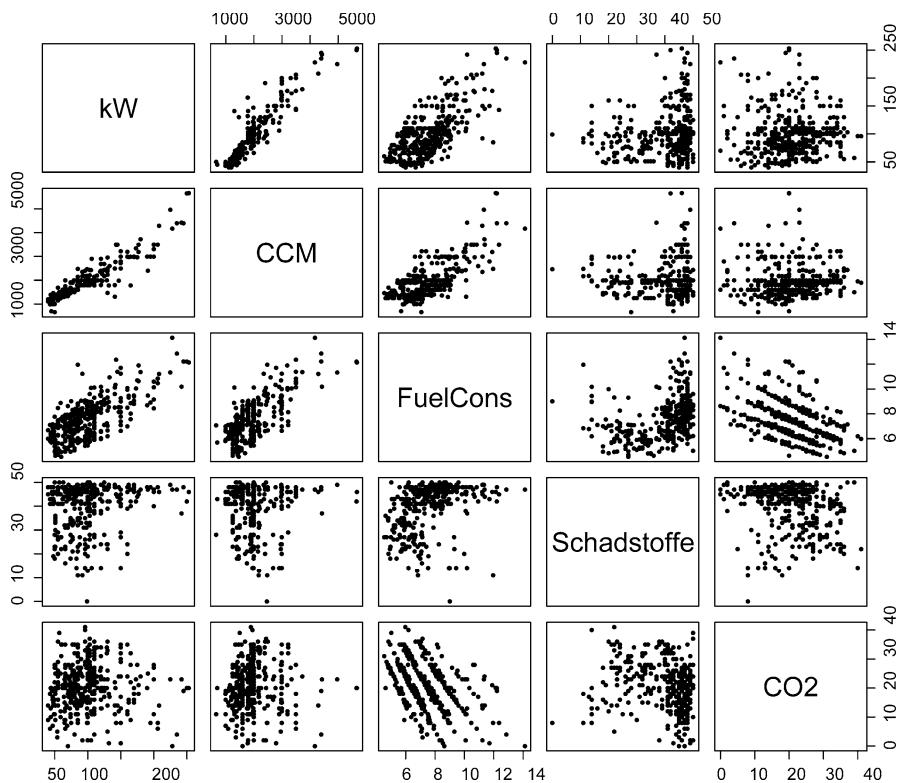


Figure 2.10. A scatterplot matrix of the five main continuous variables from a car emissions dataset from Germany. Source: <http://www.adac.de>, March 2006

crosses many others downwards). Note that there are relatively few line crossings over the later stages of the race, which means, perhaps surprisingly, that not many riders changed their race ranking.

This graphic might be improved in a number of ways: the axes could be labelled (though there is little space for this); the vertical axes could be drawn less strongly; scale information could be added (the range of the vertical axes is about 4 h, though precise values would be better read off a table of results); and the level of α -blending might be varied across the display.

Figure 2.11 shows a special form of parallel coordinate plot. Usually each axis has its own scale and there is no natural ordering of the axes. Other examples of parallel coordinate plots can be found in other chapters of the Handbook.

Mosaic Plots

2.5.3

Mosaic plots display the counts in multivariate contingency tables. There are various types of mosaicplot (Hofmann, 2000) and a 5-D example of a doubledecker plot is displayed in Fig. 2.12. The data are from a study of patterns of arrest based on 5226

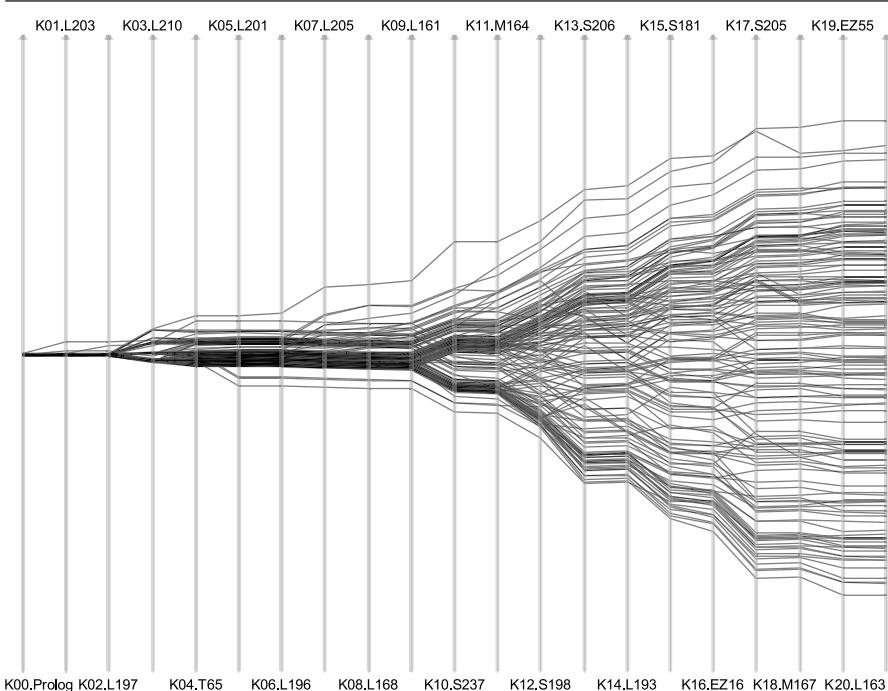


Figure 2.11. Cumulative times for riders in 2004 Tour de France for the 21 stages. The axes have a common scale and are aligned by their means. Each *vertical line* represents a stage, and they have been plotted in date order. Source: <http://www.letour.fr>

cases in Toronto. Each column represents one combination of the four binary variables *Gender*, *Employed*, *Citizen* and *Colour*. The width of a column is proportional to the number with that combination of factors. Those stopped who were not released later have been highlighted. Over 90 % of those stopped were male. Some of the numbers of females in the possible eight combinations are too small to draw firm conclusions. Each pair of columns represents the variable colour, and the proportion not released amongst the males is lower amongst the whites for all combinations of other factors. The general decline in the level of highlighting across the male columns shows that the proportion not released is lower if the person is a citizen and lower still if they are employed. Figure 2.12 shows the difficulties in displaying data of this kind in a graphic for presentation. Colour, aspect ratio and size can make a big difference, but labelling is the main problem.

2.5.4 Small Multiples and Trellis Displays

One way to avoid overloading a single large plot with information is to use a set of smaller, comparable plots instead. This can be effective for subgroup analyses [e.g. trellis displays for conditioning (Becker et al., 1996)] or for geographic data [cf. micromaps Carr (2001)]. A simple example is given in Fig. 2.13. The boxplots on their

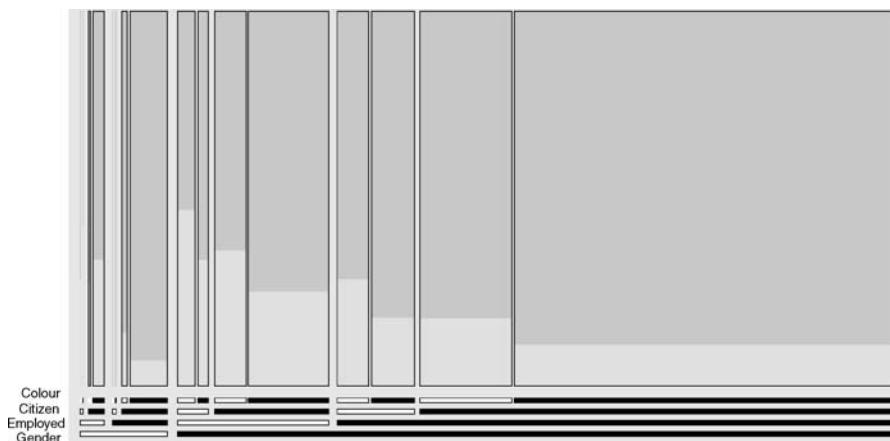


Figure 2.12. A doubledecker plot of Toronto arrest data. Source: Fox (2003)

own show that diesel cars have generally lower fuel consumption (in Europe consumption is measured in litres/100 km). The barchart on the left shows that little attention should be paid to the (Natural) Gas and Hybrid groups as few of these cars were measured. Should these two groups have been left out or perhaps be replaced by dotplots? Small groups are always a problem. It should also be noted that the units for natural gas cars are different (kg/100 km) from the others.

Small multiples can work well, but careful captioning is necessary to ensure that it is clear which smaller plot is which, and common scaling is obviously essential. Figure 2.14 is a trellis display of emissions data for the 374 petrol or diesel cars. They have been grouped by engine type (rows) and engine size (columns). An equal count grouping has been used for engine size, which is why the shaded parts of the cc bars have different lengths. Engine size seems to make little difference as the plots in each row are similar to one another. The type of engine makes more difference, with diesel

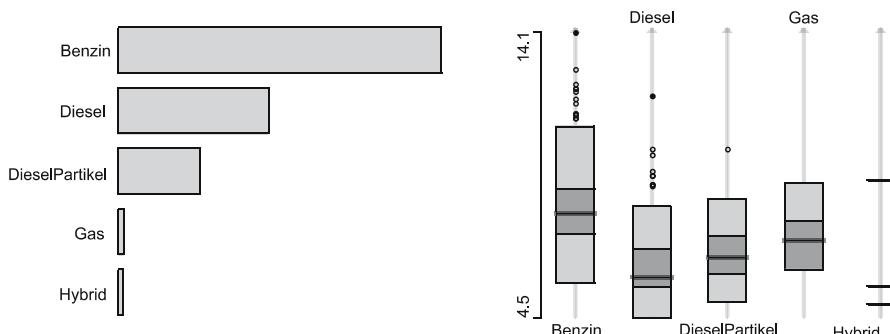


Figure 2.13. Boxplots of fuel consumption by engine type data from Germany. The barchart shows the relative numbers of cars involved. The total number was 381. Source: <http://www.adac.de>, March 2006

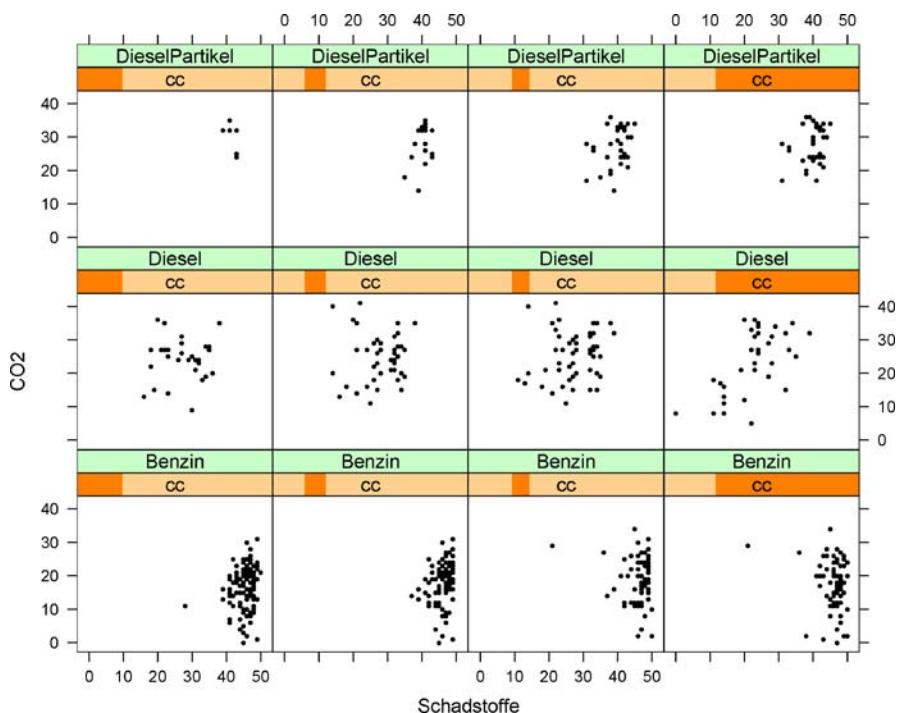


Figure 2.14. Trellis display of car emissions data from Germany. Each *panel* is a scatterplot of two pollution measures. *Rows*: type of engine; *columns*: engine size. Source: <http://www.adac.de>, March 2006

engines in particular being different from the other two types. There are a few local outliers amongst the petrol cars.

When several plots of the same kind are displayed, they can be plots of subsets of the same data, as in trellis displays, or plots of different variables for the same dataset, as in a parallel coordinates plot. It should always be obvious from the display which is the case.

2.5.5 Time Series and Maps

Time Series

Time series are special because of the strict ordering of the data, and good displays respect temporal ordering. It is useful to differentiate between value measurements at particular time points (e.g. a patient's weight or a share price) and summary measurements over a period (e.g. how much the patient ate in the last month or how many shares were traded during the day).

Time scales have to be carefully chosen. The choice of time origin is particularly important, as anyone who looks at the advertised performance of financial funds will know. Time points for value measurements may not match the calendar scale (e.g.

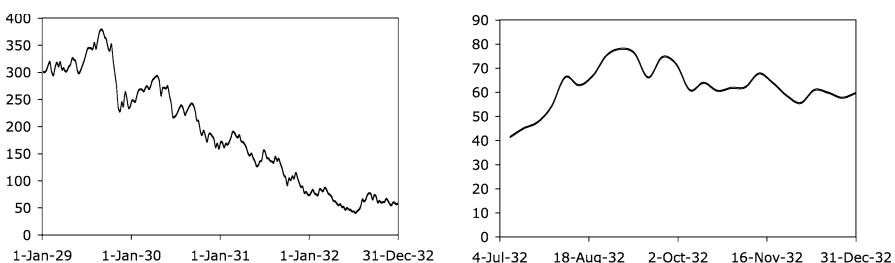


Figure 2.15. Weekly Dow Jones Industrial Average: **a** Four years from 1929 to 1932. **b** Six months from July to December 1932. The maximum vertical axis value on the *left* is over four times the maximum on the *right*

daily share prices only being available on days the market is open). Time units for summary measurements may be of unequal length (e.g. months). The time period chosen and the aspect ratio used for a time series plot can make a big difference in the interpretation of the data (Fig. 2.15).

If several time series are plotted in the same display, then it is necessary to ensure that they are properly aligned in time (e.g. two annual economic series may be published at different times of the year), that their vertical scales are matched (the common origin and the relative ranges) and that they can be distinguished from one another. Depending on the data, this can be tricky to do successfully.

Maps

Geographic data are complex to analyse, though graphical displays can be very informative. Bertin discussed many ways of displaying geographic data in his book, and MacEachren's book contains a lot of sound advice (MacEachren, 1995), though more from a cartographic point of view. The main problems to be solved lie in the fact that areas do not reflect the relative importance of regions (e.g. Montana has fewer people than New York City but is much bigger) and spatial distance is not directly associated with similarity or nearness (e.g. where countries are divided by natural borders, like mountain ranges). There is a substantial research literature in geography on these and other display issues, such as how to use colour scales to show values ('choropleth maps') and how to choose colour schemes (e.g. Colorbrewer referred to above). Some instructive examples can be found in the cancer atlas maps of US health authorities on the Web and in the book by Devesa et al. (1999). Figure 2.16 shows that cancer rates are highest along the East Coast and lowest in the Midwest. State Economic Areas (SEAs) have been chosen because using states oversmooths the data (consider Nevada in the West with its high cancer rate around Las Vegas, but its lower rate elsewhere), while using counties undersmooths. The map on the website is in colour, on a scale from deep red for high rates to dark blue for low. Naturally, this would not reproduce well in a grey-scale view, so the webpage provides the alternative version that is used here. Offering multiple versions of the same image on the Web is readily possible but not often done. This is one of several reasons why the cancer atlas webpages are exemplary.

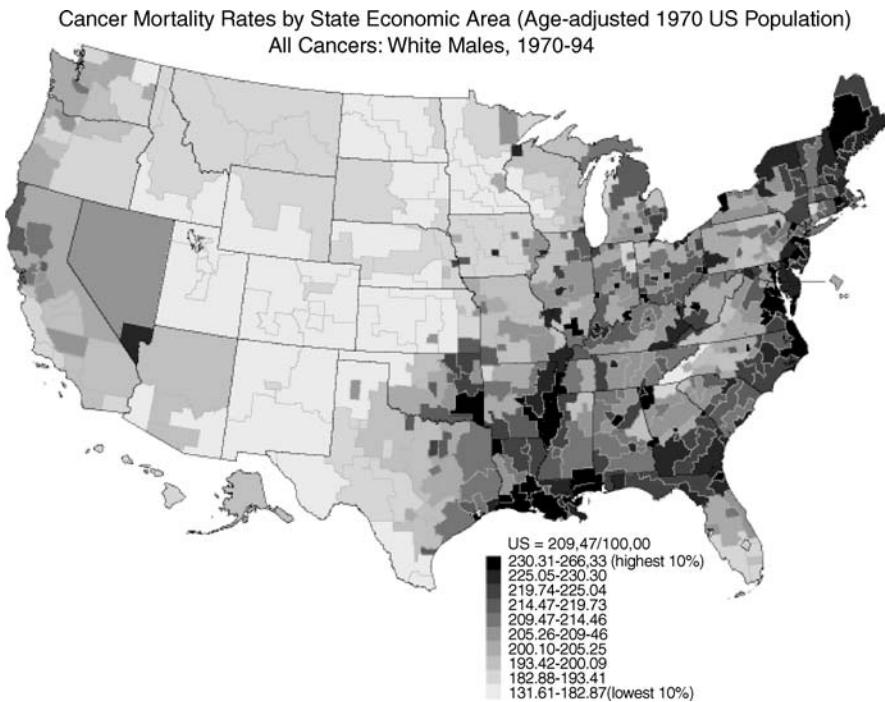


Figure 2.16. Cancer mortality rates for white males in the USA between 1970 and 1994 by State Economic Area. The scale has been chosen so that each interval contains 10 % of the SEAs. Source: <http://www3.cancer.gov/atlasplus/>

2.6 Practical Advice

2.6.1 Software

For a long time all graphics had to be prepared by draftsmen by hand. The volumes of the *Album de Statistique Graphique* produced towards the end of the 19th century contain many exceptional displays which must have taken much painstaking preparation. Such graphics may be individually designed with special features for the particular data involved. Nowadays graphics are produced by software, and this has tended to mean that certain default displays are adopted by many as a matter of course. If it takes a few minutes to prepare a graphic that is standard in your field, why bother to prepare something novel? This has advantages – standards avoid possible gross errors and are readily understood by readers familiar with them – and disadvantages – not all data fit the existing standards and interesting new information may be obscured rather than emphasized by a default display. As software becomes more sophisticated and user interfaces become more intuitive, this may change. Currently (in 2006), there are software packages which give users substantial control over all aspects of the displays they wish to draw, but these are still only for experts in the

software (Murrell, 2005). It is reasonable to assume that there will be a steady progression to a situation where even non-experts will be able to draw what they wish. Whether good graphics are the result will depend on the users' statistical good sense and on their design ability. Like the quality of a scientific article, the quality of a data visualization graphic depends on content and presentation. How has the quality of scientific articles changed since scientists have been able to prepare their own drafts with sophisticated text preparation software?

Bad Practice and Good Practice (Principles)

2.6.2

Sometimes it is easier to see what has gone wrong than to explain how to do something right. Take the simple task of preparing a barchart to display univariate categorical data. What could possibly go wrong? The bars may be too thin (or too fat); the gaps between the bars may be too narrow (or too wide): the labelling of the bars may be unclear (because it is difficult to fit long category names in); the order of the bars may be confusing; the vertical scale may be poorly chosen; there may be superfluous gridlines; irrelevant 3-D effects may have been used; colours or shading may have been unnecessarily added; or the title may be misleading and the caption confusing. Doubtless there are even more ways of ruining a barchart.

It is not possible to give rules to cover every eventuality. Guiding principles like those outlined in this chapter are needed.

And Finally

2.7

The lack of formal theory bedevils good graphics. The only way to make progress is through training in principles and through experience in practice. Paying attention to content, context and construction should ensure that sound and reliable graphics are produced. Adding design flair afterwards can add to the effect, so long as it is consistent with the aims of the graphic.

Gresham's Law in economics states that 'bad money drives out good.' Fortunately this does not seem to apply to graphics, for while it is true that there are very many bad graphics displays prepared and published, there are also many very good ones. All serious data analysts and statisticians should strive for high standards of graphical display.

References

- Becker, R., Cleveland, W. and Shyu, M.-J. (1996). The visual design and control of trellis display, *JCGS* 5:123–155.
- Burn, D. (1993). Designing effective statistical graphs, in C. Rao (ed), *Handbook of Statistics*, Vol. 9, Elsevier, pp. 745–773.
- Carr, D.B. (2001). Designing linked micromap plots for states with many counties, *Statistics In Medicine* 20:1331–1339.

- Cleveland, W. (1994). *The Elements of Graphing Data*, revised edn, Hobart Press, Summit, New Jersey, USA.
- Dawson, R. (1995). The ‘unusual episode’ data revisited, *Journal of Statistics Education (Online)* 3(3).
- Devesa, S., Grauman, D., Blot, W., Pennello, G., Hoover, R. and Fraumeni, J.J. (1999). *Atlas of cancer mortality in the United States, 1950-1994*, US Govt Print Off, Washington, DC.
- Everitt, B. (1993). *Cluster Analysis*, 3rd edn, Edward Arnold, London.
- Fox, J. (2003). Effect displays in r for generalised linear models, *Journal of Statistical Software* 8(15).
- Hofmann, H. (2000). Exploring categorical data: interactive mosaic plots, *Metrika* 51(1):11–26.
- Inselberg, A. (1999). Don’t panic ... do it in parallel, *Computational Statistics* 14(1):53–77.
- Izenman, A. and Sommer, C. (1988). Philatelic mixtures and multimodal densities, *Journal of the American Statistical Association* 83(404):941–953.
- Kosslyn, S. (1994). *Elements of Graph Design*, Freeman, New York.
- MacEachren, A. (1995). *How Maps Work*, Guildford Press, New York.
- Murrell, P. (2005). *R Graphics*, Chapman & Hall, London.
- Norman, D. (1988). *The Design of Everyday Things*, Doubleday, New York.
- Playfair, W. (2005). *Playfair’s Commercial and Political Atlas and Statistical Breviary*, Cambridge, London.
- Putnam, R. (2000). *Bowling Alone*, Touchstone, New York.
- Robbins, N. (2004). *Creating More Effective Graphs*, John Wiley.
- Spence, R. (2001). *Information Visualization*, Addison-Wesley, New York.
- Tufte, E. (2001) *The Visual Display of Quantitative Information*, 2nd edn, Graphic Press, Cheshire, Connecticut.
- Wainer, H. (1997). *Visual Revelations*, Springer, New York.
- Wainer, H. (2004). *Graphic Discovery: a Trout in the Milk and other Visual Adventures*, Princeton UP.
- Wilkinson, L. (2005). *The Grammar of Graphics*, 2nd edn, Springer, New York.

Static Graphics

II.3

Paul Murrell

3.1	<i>Complete Plots</i>	81
	Sensible Defaults	82
	User Interface	84
3.2	<i>Customization</i>	84
	Setting Parameters	84
	Arranging Plots.....	87
	Annotation.....	88
	The User Interface.....	92
3.3	<i>Extensibility</i>	92
	Building Blocks	93
	Combining Graphical Elements	97
	The User Interface.....	98
3.4	<i>Other Issues</i>	98
	3-D Plots	98
	Speed	98
	Output Formats	99
	Data Handling	99
3.5	<i>Summary</i>	100

This chapter describes the requirements for a modern statistical graphics system for the production of static plots. There is a discussion of the production of complete plots, customizing plots, adding extra output to plots and creating entirely new plots.

Statistical graphics is described as an extension of a general graphics language. There is an emphasis on the importance of support for sophisticated graphics facilities such as semitransparent colours, image compositing operators and the complex arrangement of graphical elements.

Static displays of information continue to be the primary graphical method for the display and analysis of data. This is true both for presentation purposes, where the vast majority of data displays produced for articles and reports are still static in nature, and for data exploration, where many important statistical discoveries have been made based simply on static displays (e.g. Cleveland's barley data discovery using Trellis plots; Cleveland, 1993). The recent advances in dynamic and interactive displays (e.g. Swayne et al., 2003; Theus, 2002) provide us with wonderful additional tools, but static displays still play a fundamental role in the presentation of data.

There are very many software packages (some of them statistical) that provide ways to produce static displays of data. This is good, because these computer programs allow us to produce more complex graphics, and graphics in greater volumes, than was ever possible when working just with pen and paper. But how good is the software for displaying data? More importantly, how good *could the software be?* What should we expect from our statistical graphics software?

This chapter addresses these questions by discussing the important features which software for the static display of data should provide. In addition, there are descriptions of ways to provide those features. For each topic, there will be an abstract discussion of the issue followed by concrete examples implemented in R (R Development Core Team, 2005). The use of R is natural for me due to my personal familiarity with the system, but it is also justified by the fact that R is widely acknowledged as being pretty good at producing static displays of data, and, to my knowledge, some of the ideas can only be demonstrated in R.

The Grammar of Graphics

A comprehensive overview of statistical graphics is provided by Wilkinson's *Grammar of Graphics* (Wilkinson, 1999, 2004). Wilkinson outlines a system in which statistical graphics are described in a high-level, abstract language and which encompasses more than just static graphical displays.

This chapter provides a different view, where statistical graphics is seen as an extension of a general graphics language like PostScript (Inc., 1990) or SVG (Ferraiolo et al., 2003). This view is lower level, more explicit about the basic graphical elements which are drawn and more focused on static graphics.

To emphasize the difference, consider a simple barplot of birth rate for three different types of government (Fig. 3.1). A *Grammar of Graphics* description (or part

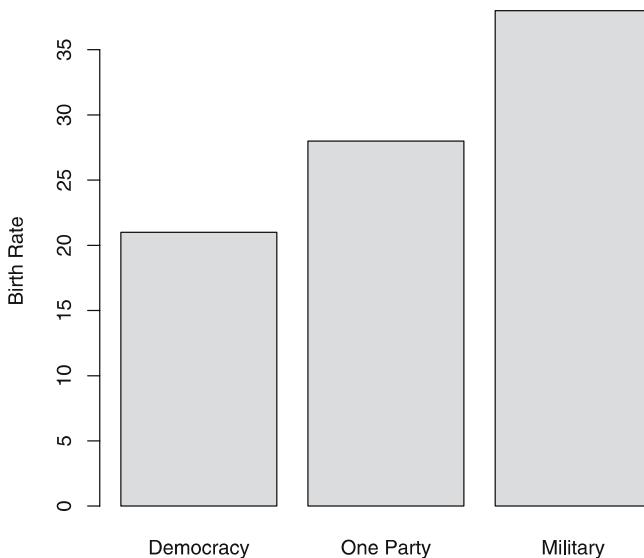


Figure 3.1. A simple barchart of birth rate for three different types of government

thereof) for the barplot would be a statement of the following form (from p. 88 of the *Grammar of Graphics*, 1st edn. Wilkinson, 1999):

FRAME: **gov*birth**

GRAPH: **bar()**

A description consistent with this chapter would involve a description of the coordinate systems and graphical shapes that make up the plot. For example, the barplot consists of a plotting region and several graphical elements. The plotting region is positioned to provide margins for axes and has scales appropriate to the range of the data. The graphical elements consist of axes drawn on the edges of the plotting region, plus three rectangles drawn relative to the scales within the plotting region, with the height of the rectangles based on the birth rate data.

Complete Plots

3.1

We will start with the most obvious feature of statistical graphics software: the user should be able to produce graphical output. In other words, the user should be able to draw something. In most cases, the user will want to draw some sort of plot consisting of axes, labels and data symbols or lines to represent the data. Figure 3.2 shows an example consisting of a basic scatterplot.

This is one distinction between statistical graphics software and a more general graphics language such as PostScript. The user does not just want to be able to draw

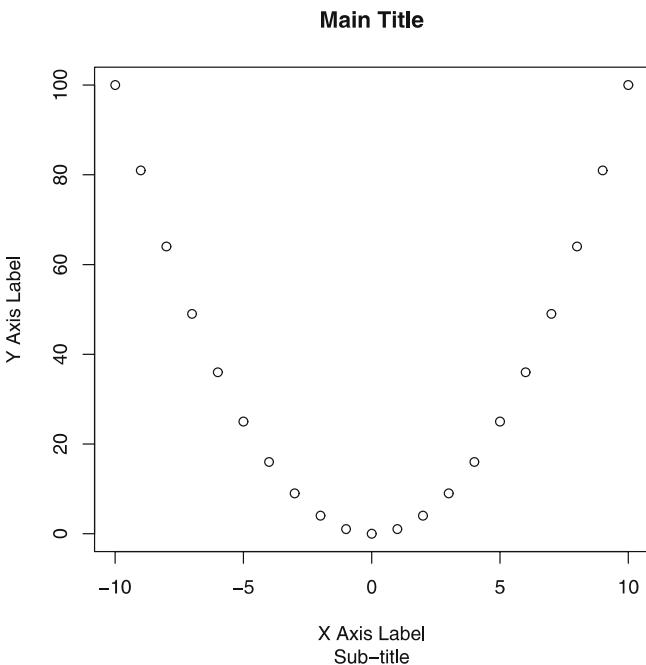


Figure 3.2. A basic scatterplot consisting of *axes*, *labels* and *data symbols*

lines and rectangles (though he may also want to do that; see Sect. 3.2.3). The user wants to be able to create an entire plot. To be even more explicit, the user wants to be able to draw an entire plot *with a single command* (or via a single menu selection).

This is so far quite uncontroversial, and all statistical software packages provide this feature in one way or another (though they may differ in terms of the range of different sorts of plots that can be produced). In R, the following command usually does the trick (where the variable `somedata` contains the data values to plot).

```
> plot(somedata)
```

3.1.1 Sensible Defaults

Take another look at the basic plot in Fig. 3.2. As we have mentioned, it consists of a standard set of components: axes, labels and data symbols. But there are other important aspects to this plot. For a start, these components are all in sensible locations; the title is at the top and, very importantly, the data symbols are at the correct locations relative to the axes (and the scales on the axes ensure that there is sufficient room for all of the data points).

Some of these aspects are inevitable; no one would use a program that drew data symbols in the wrong locations or created axis scales so that none of the data could be seen. However, there are many aspects that are less obvious.

Why should the title be at the top? Did you notice that the title uses a sans serif font? Why is that? Something else the software has done is to position the tick marks at sensible locations within the range of the data. Also, the axes have their tick marks and tick labels pointing away from the region where the data are plotted (other software may do this differently). Does that matter?

In some of these cases, there are clear reasons for doing things a certain way (e.g. to improve clarity or visual impact; Cleveland, 1993, 1985; Robbins, 2005; Tufte, 1989). In other cases, the choice is more subjective or a matter of tradition. The main point is that there are a number of ways that the software could do these things. What is important is that the software should provide a good default choice.

Trellis Plots

A good example of a graphics system that provides sensible defaults is the Trellis system (Becker et al., 1996). The choice of default values in this system has been guided by the results of studies in human perception (Cleveland and McGill, 1987) so that the information within a plot will be conveyed quickly and correctly to the viewer. In R, the *lattice* package (Sarkar, 2002) implements Trellis plots. Figure 3.3 shows a Trellis version of a basic scatterplot. One subtle, but well-founded, difference with Fig. 3.2 is the fact that the labels on the tick marks of the *y*-axis are horizontal so

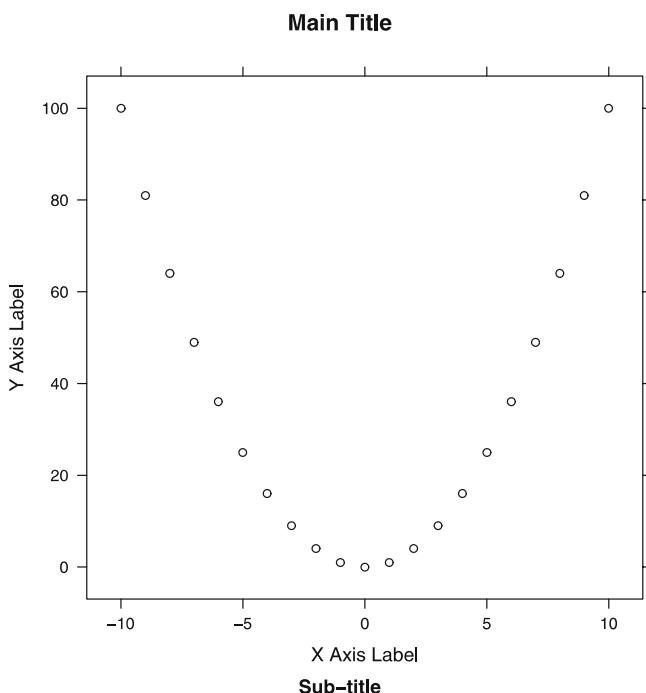


Figure 3.3. A basic Trellis scatterplot, which has a different default appearance from the scatterplot in Fig. 3.2

that they are easier to read. The subtitle of Fig. 3.3 is also more heavily emphasized by using a bold font face. The Trellis defaults extend to selections of plotting symbols and colours in plots of multiple data series, which are chosen so that different data series can be easily distinguished by the viewer.

3.1.2

User Interface

A sometimes controversial aspect of statistical graphics software is the user interface. The choice is between a command line, where the user must type textual commands (or function calls), and a graphical user interface (GUI), consisting of menus and dialogue boxes. A batch system is considered to be a command-line interface; the important point is that the user has to do everything by typing on the keyboard rather than by pointing and clicking with a mouse. Often both a command line and a GUI will be offered.

The interface to a piece of software is conceptually orthogonal to the set of features that the software provides, which is our main focus here. Nevertheless, in each section of this chapter we will briefly discuss the user interface because there are situations where the interface has a significant impact on the accessibility of certain features.

For the purpose of producing complete plots, the choice of user interface is not very important. Where one system might have an option on a GUI menu to produce a histogram, another system can have a command or function to do the same thing.

With R, the standard interface is a command line, but a number of GUI options exist, notably Rcmdr (Fox, 2005), JGR (Helbig et al., 2004) and the standard GUI on the Mac platform (Iacus and Urbanek, 2004).

3.2

Customization

Let us assume that your statistical software allows you to produce a complete plot from a single command and that it provides sensible defaults for the positioning and appearance of the plot. It is still quite unlikely that the plot you end up with will be exactly what you want. For example, you may want a different scale on the axes, or the tick marks in different positions, or no axes at all. After being able to draw something, the next most important feature of statistical graphics software is the ability to control what gets drawn and how it gets drawn.

3.2.1

Setting Parameters

For any particular piece of output, there will be a number of free parameters that must be specified. As a very basic example, it is not sufficient to say something like ‘I want to draw a line’; you must also specify where the line should start and where it should end. You might be surprised how many free parameters there are in even simple cases like this; in order to fully specify the drawing of a single straight line, it is necessary to provide not only a start and end point, but a colour, a line style (perhaps

dashed rather than solid), how thick to draw the line and even a method for how to deal with the ends of the line (should they be rounded or square?).

When producing plots, you deal with more complex graphical output than just a single line, and more complex graphical components have their own sets of parameters. For example, when drawing an axis, one parameter might control the number of tick marks on the axis and another might control the text for the axis label. When drawing a complete plot, an important parameter is the data to plot(!), but there may also be parameters to control whether axes are drawn, whether a legend (or key) is provided, and so on.

Wherever there is a parameter to control some aspect of graphical output, the user should have the ability to provide a value for that parameter.

In R, each graphics function provides a set of parameters to control aspects of the output. The following code shows how a plot can be created with no axes and no labels by specifying arguments for `axes` and `ann` respectively. A line is added to the plot with control over its location and its colour, line type and line width.

```
> plot(1:10, axes=FALSE, ann=FALSE)
> lines(1:10, col="red", lty="dashed", lwd=3)
```

Graphical Parameters

There is a common set of ‘graphical’ parameters that can be applied to almost any graphical output to affect the appearance of the output. This set includes such things as line colour, fill colour, line width, line style (e.g. dashed or solid) and so on. This roughly corresponds to the concept of graphics state in the PostScript language.

In order to be able to have complete control over the appearance of graphical output, it is important that statistical graphics software provides a complete set of graphical parameters. Examples of parameters that may sometimes be overlooked

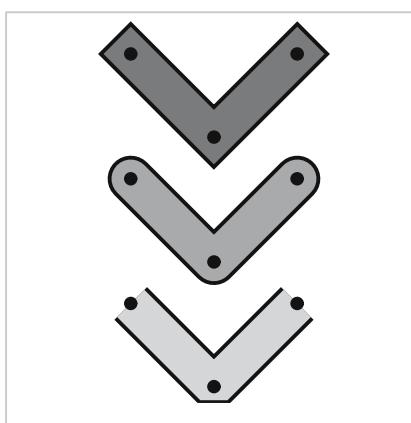


Figure 3.4. Line join and line ending styles. Three *thick lines* have been drawn with different line end and line join styles. The *top line* has ‘square’ ends and ‘mitre’ joins, the *middle line* has ‘round’ ends and ‘round’ joins, and the *bottom line* has ‘butt’ ends and ‘bevel’ joins. In each case, the three *points* that the *line* goes through are indicated by *black circles*

are semitransparent colours, line joins and endings (Fig. 3.4) and full access to a variety of fonts. Edward Tufte has recommended (Tufte, 2001) the use of professional graphics software such as Adobe Illustrator to achieve quality results, but even better is the ability to provide the control within the statistical graphics software itself.

In R there is a large set of graphical parameters that allow control over many aspects of graphical output, such as colours, line types and fonts (see the previous example code demonstrating the control of colour, line type and line width), but this could be extended further to include any of the basic drawing parameters and operators that you will find in a sophisticated graphics language such as SVG. Examples are gradient fills (where a region is filled with a smoothly varying colour), general pattern fills and composition of output.

An example of the use of composition operators is the addition of a legend to a plot, both of which have a transparent background, but where the plot has grid lines. If we do not want the grid lines to appear in the legend background, one way to achieve

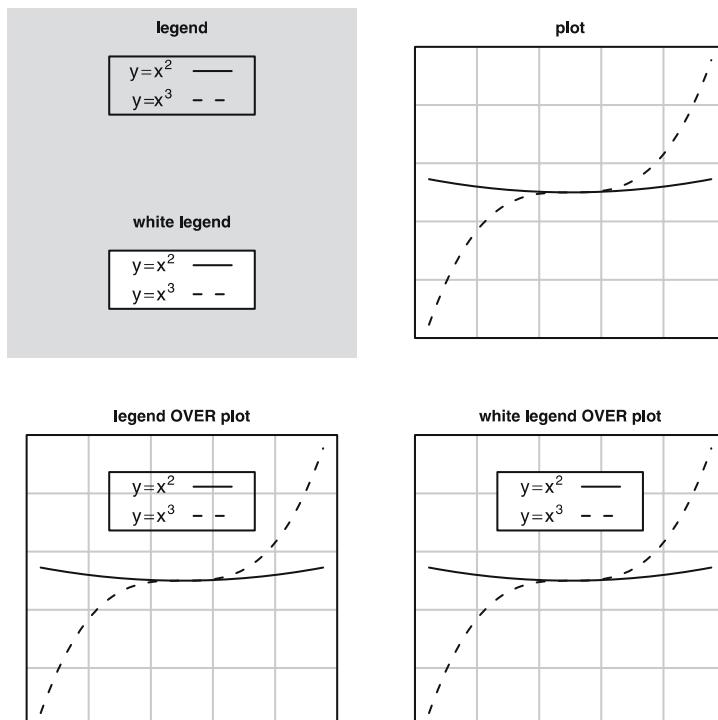


Figure 3.5. Composing graphical elements on a white background. There are three elements being composed: two legends, one with a transparent background and one with a white background (*top left*), and a plot with a transparent background (*top right*). In the *bottom left*, the legend with a transparent background is drawn over the *plot* and the *grid lines* in the *plot* are visible behind the legend. In the *bottom right*, the legend with a white background is drawn over the *plot* and the *grid lines* in the *plot* are not visible behind the legend

that is to combine the legend with the plot in such a way that the legend output completely replaces the plot output over the region that the legend is drawn. Figure 3.5 shows how just drawing the legend on top of the plot produces the wrong result (there are grid lines visible behind the legend). Using an opaque background in the legend does the job as long as we can anticipate the background colour that the overall plot will be drawn on (Fig. 3.5). However, this is not a good general solution because it fails badly if a different background colour is encountered (Fig. 3.6). A general solution involves more complex image manipulations, such as negating the alpha channel

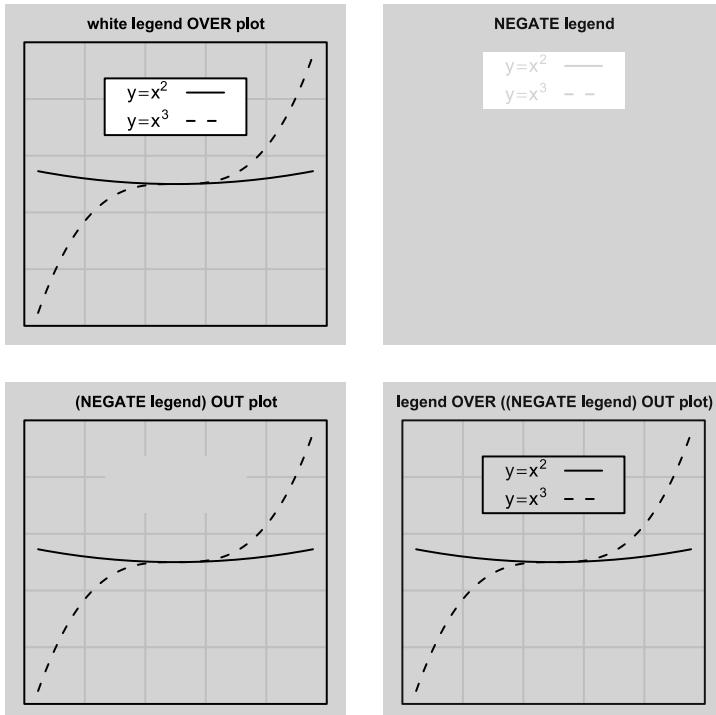


Figure 3.6. Composing graphical elements on a grey background to show that the use of an opaque background for a legend (as in Fig. 3.5) is not suitable if the background of the final image is a different colour. There are three elements being composed: two legends, one with a transparent background and one with a white background (*top left* in Fig. 3.5), and a plot with a transparent background (*top right* in Fig. 3.5). In the *top left* in this figure, the legend with a white background is drawn over the *plot* and the *grid lines* in the *plot* are not visible behind the legend, but the white background of the legend does not match the background of the *plot*, so the result is unpleasant. In the *top right*, the legend with a transparent background has had its alpha channel (opacity) negated, so that the background is the only part that is opaque. In the *bottom left*, the negated legend is composited with the *plot* using an ‘*out*’ operator, thereby creating a ‘*hole*’ in the *plot*. In the *bottom right*, the (untransformed) legend with a transparent background is drawn over the *plot* with a hole and the *grid lines* in the *plot* are not visible behind the legend, but the background of the final image is still grey, which is the desired result

(inverting the opacity) of the legend and using a Porter–Duff (Porter and Duff, 1984) ‘out’ compositing operator to create a ‘hole’ for the legend within the plot (Fig. 3.6).

3.2.2

Arranging Plots

Where several plots are produced together on a page, a new set of free parameters becomes available, corresponding to the location and size of each complete plot. It is important that statistical graphics software provides some way to specify an arrangement of several plots.

In R, it is easy to produce an array of plots all of the same size, as shown by the code below.

```
> par(mfrow=c(2, 2))
```

It is also possible to produce arrangements where plots have different sizes. The following code gives a simple example (Fig. 3.7):

```
> layout(rbind(c(1, 2),
+               c(0, 0),
+               c(3, 3)),
+         heights=c(1.5, 1cm(0.5), 1))
```

The idea of arranging several plots can be generalized to the arrangement of arbitrary graphical elements; we will discuss this more in Sect. 3.3.

3.2.3

Annotation

A more complex sort of customization involves the addition of further graphical output to a plot. For example, it can be useful to add an informative label to one or more data symbols in a plot.

Graphical Primitives

The first requirement for producing annotations is the ability to produce very basic graphical output, such as simple text labels. In this way, statistical graphics software needs to be able to act like a generic drawing program, allowing the user to draw lines, rectangles, text and so on. In other words, it is not good if the software can ‘only’ draw complete plots.

In R, there are functions for drawing a standard set of graphical primitives. The following code demonstrates how rectangles, lines, polygons and text can be added to a basic plot (Fig. 3.8):

```
> x <- rnorm(20)
> plot(x)
> polygon(c(1, 1:20, 20), c(0, x, 0),
+           col="grey", border=NA)
> rect(1, -0.5, 20, 0.5,
+       col="white", lty="dotted")
> lines(x)
> points(x, pch=16)
> text(c(0.7, 20.3), 0, c("within", "control"), srt=90)
```

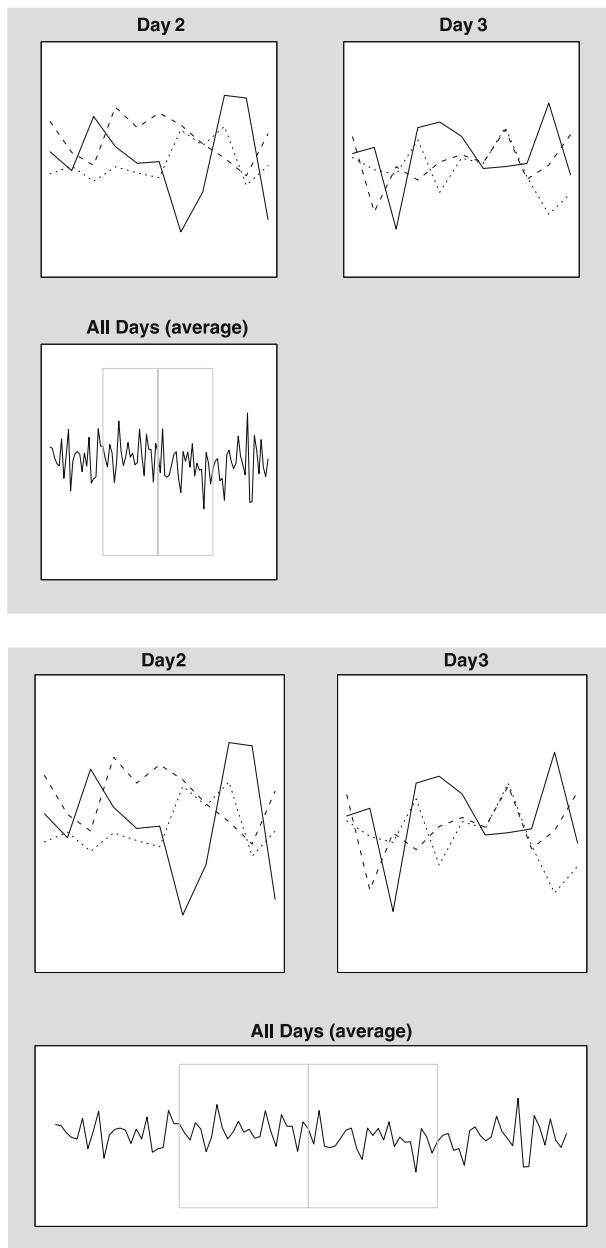


Figure 3.7. Two arrangements of multiple plots on a page. In the *top* example, all of the plots have the same size; in the *bottom* example, several plots of different sizes are arranged

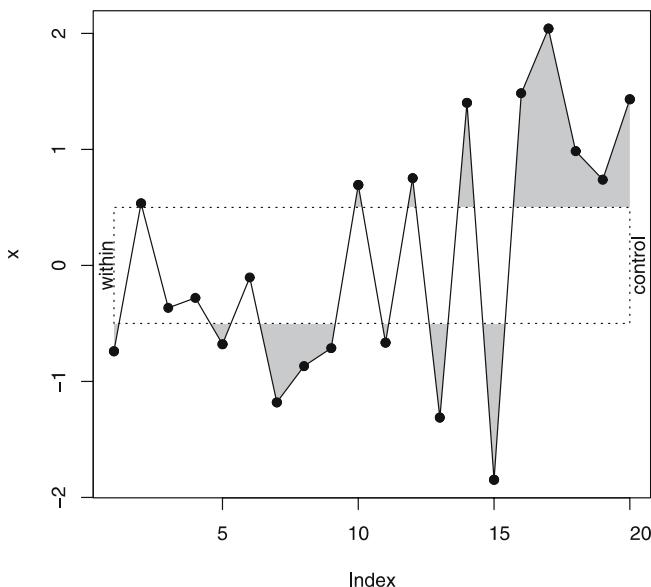


Figure 3.8. Basic scatterplot with extra *rectangles*, *lines*, *polygons* and *text* added to it

Some slightly more complex primitives (not currently natively supported by R) are spline curves, arbitrary paths (as in PostScript or SVG) and polygons with holes, which are useful for drawing maps. An example of a polygon with a hole is an island within a lake within an island, where both islands are part of the same country or state and so are usefully represented as a single polygon.

Coordinate Systems

One of the most important and distinctive features of statistical graphics software is that it is not only capable of producing many pieces of graphical output at once (lots of lines, text, and symbols that together make up a plot), but that it is also capable of positioning the graphical output within more than one coordinate system. Here are some examples (Fig. 3.9):

- The title of a plot might be positioned halfway across a page. That is, the title is positioned relative to a ‘normalized’ coordinate system that covers the entire page, where the location 0 corresponds to the left edge of the page and the location 1 corresponds to the right edge.
- The data symbols in a scatterplot are positioned relative to a coordinate system corresponding to the range of the data that only covers the area of the page bounded by the plot axes.
- The axis labels might be positioned halfway along an axis. That is, the axis labels are positioned relative to a ‘normalized’ coordinate system that only covers the area of the page bounded by the plot axes.

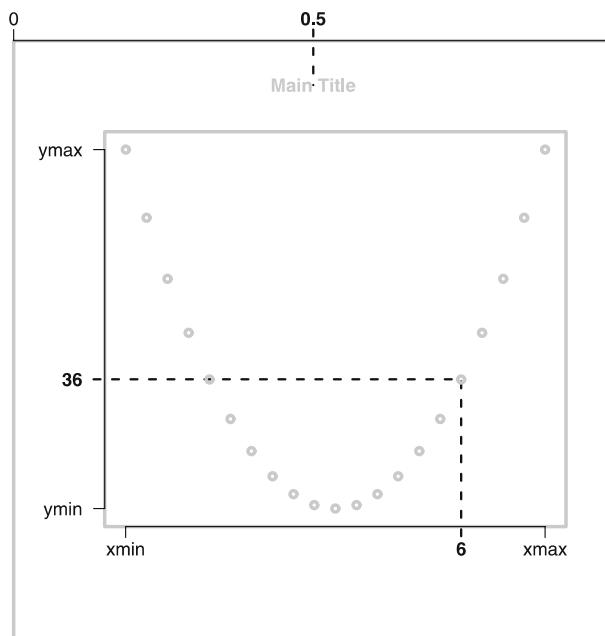


Figure 3.9. Two of the coordinate systems involved in producing a simple scatterplot. A ‘normalized’ coordinate system that covers the whole page is used to centre the plot title, and a coordinate system based on the range of the data that covers the plot region is used to position the *data symbols*

Many users of statistical graphics software produce a plot and then export it to a format which can be easily edited using third-party software (e.g. export to WMF and edit using Microsoft Office products). This has the disadvantage that the coordinate systems used to produce the plot are lost and cannot be used to locate or size annotations. Furthermore, it makes it much harder to automate or programmatically control the annotation, which is essential if a large number of plots are being produced.

When it comes to annotating a plot, it is important that output can be added relative to the coordinate systems which were used to draw the original plot. For example, in Fig. 3.8 all additional output is positioned relative to the scales on the plot axes.

Because there are several coordinate systems used in the construction of a graph, there must be some way to specify which coordinate system to use when adding further output.

In R’s traditional graphics, each function for adding additional output to a plot only works with a single coordinate system. For example, the `text()` function only positions text relative to the scales on the axes and the `mttext()` function only positions text relative to the plot margins (where the axis labels and plot titles are drawn). R’s `grid` package (Murrell, 2002) provides a more general approach; it is described in more detail in Sect. 3.3.

Non-Cartesian Coordinates

There are many examples of useful plots and diagrams that require non-cartesian coordinates, so it is desirable for statistical graphics software to support or at least allow the construction of a variety of coordinate systems. For example, a number of data sources suit polar coordinate displays, such as wind diagrams; when plotting a single categorical variable with exactly three levels, ternary plots can be effective; hierarchical data are naturally displayed as trees or graphs (with nodes and edges).

3.2.4

The User Interface

The user interface for providing parameters to control graphical output can be adequately provided by either a command line or GUI. In a command-line environment, function calls can be made with an argument provided for each control parameter; GUIs tend to provide dialog boxes full of various options.

One issue that arises with statistical graphics is the ‘explosion’ of parameters for higher-level graphical elements. Consider a matrix of scatterplots: the matrix contains many plots; each plot contains several axes; each axis consists of multiple lines and pieces of text. How can you provide parameters to control each piece of text in every axis on every plot? That is a lot of parameters. The problem essentially is one of being able to uniquely specify a particular component of an overall plot.

A mouse input device provides a very good way of specifying elements in an image. It is very natural to point at the element you want. However, there are issues when selecting components of a plot because there is often ambiguity due to the hierarchical structure inherent in a plot. If you click on a piece of text on an axis tick mark, it is not clear whether you want to select just the text, or the entire axis, or even the entire plot. The advantage of using a command line to select objects is that, although it may be less convenient, you can typically be more expressive, or more precise. For example, in the grid graphics system in R, the text for a particular axis might be expressed as the following ‘path’: “`plot1::xaxis::label`”.

Another problem with the GUI approach is that it is hard to capture a particular editing operation. For example, if the same editing operation is required on another plot, the same series of actions must be repeated by the user. In a command-line environment, operations can be captured and repeated easily.

3.3

Extensibility

The ability to produce complete plots, control all aspects of their appearance and add additional output represents a minimum standard for what statistical graphics software should provide. A more advanced feature is the ability to extend the system to add new capabilities, such as new types of plots.

In some respects, creating a new sort of plot is just an extreme version of customization, but there are two distinguishing features: you are starting from a blank slate rather than building on an existing plot as a starting point (i.e. it is not just annotation) and, more importantly, extensibility means that the new plot that you create

is made available for others to use in exactly the same way as existing plots. To be more explicit, in an extensible system you can create a new menu item or function that other users can access.

So what sorts of features are necessary or desirable to support the development of new plots? For a start, the system must allow new functions or menu items to be added, and these must be able to be added *by the user*. The next most important features are that low-level building blocks must be available and there must be support for combining those building blocks into larger, coherent graphical elements (plots).

Building Blocks

3.3.1

What are the fundamental building blocks from which plots are made? At the lowest level, a plot is simply basic graphical shapes and text, so these must be available (see ‘Graphical Primitives’ in Sect. 3.2.3). In addition, there must be some way to define coordinate systems so that graphical elements can be conveniently positioned in sensible locations to make up a plot. Surprisingly, that’s about it. Given the ability to draw shapes and locate them conveniently, you can produce a huge variety of results. Controlling coordinate systems is a special case of being able to define arbitrary transformations on output, such as is provided by the current transformation matrix in PostScript or transform attributes on group elements in SVG.

We have already seen that R provides basic graphical elements such as lines and text (Sect. 3.2.3). R also provides ways to control coordinate systems; this discussion will focus on the features provided by the `grid` system because they are more flexible.

The `grid` system in R provides the concept of a ‘viewport’, which represents a rectangular region on the page and contains several different coordinate systems. Viewports can be nested (positioned within each other) to produce quite complex arrangements of regions. The following code provides a simple demonstration (Fig. 3.10). First of all, we create a region centred on the page, but only 80 % as wide and high as the page.

```
> pushViewport(viewport(width=0.8, height=0.8,
                        xscale=c(0, 3), yscale=c(0, 10)))
```

This now is where drawing occurs, so rectangles and axes are drawn relative to this viewport.

```
> grid.rect(gp=gpar(fill="light grey"))
> grid.xaxis(at=1:2, gp=gpar(cex=0.5))
> grid.yaxis(gp=gpar(cex=0.5))
```

Now we define a new viewport, which is located at (1, 4) relative to the axis scales of the first viewport. This also demonstrates the idea of multiple coordinate systems; the width and height of this new viewport are specified in terms of absolute units, rather than relative to the axis scales of the previous viewport.

```
> pushViewport(viewport(unit(1, "native"),
                        unit(4, "native"),
                        width=unit(1, "cm"),
                        height=unit(1, "inches")))
```

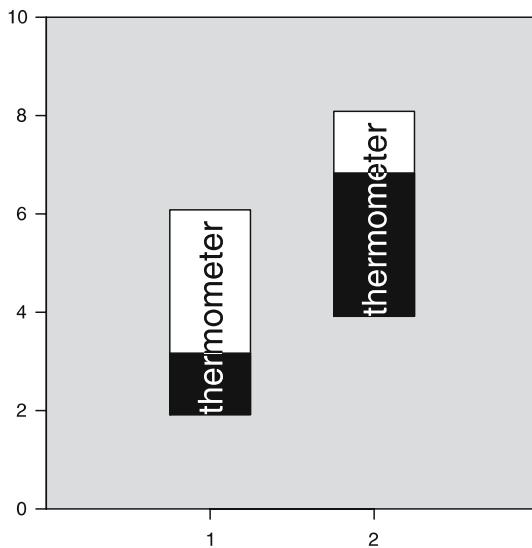


Figure 3.10. A demonstration of grid viewports. The overall data region (bounded by the *axes*) is a viewport, each overall thermometer is another viewport, and the *black region* within each thermometer is yet another viewport. The *white text* within each thermometer is also drawn within its own (clipped) viewport

We draw a rectangle around this new viewport and then draw the word ‘thermometer’.

```
> grid.rect(gp=gpar(fill="white"))
> grid.text("thermometer",
           y=0, just="left", rot=90)
```

We create yet another viewport, which is just the bottom 30 % of the second viewport, and draw a filled rectangle within that.

```
> pushViewport(viewport(height=0.3, y=0,
                        just="bottom"))
> grid.rect(gp=gpar(fill="black"))
```

Finally, we create a viewport in exactly the same location as the third viewport, but this time with clipping turned; when we draw the word ‘thermometer’ again in white, it is only drawn within the filled black rectangle.

```
> pushViewport(viewport(clip=TRUE))
> grid.text("thermometer",
           y=0, just="left", rot=90,
           gp=gpar(col="white"))
```

A second thermometer has been drawn in a similar manner in Fig. 3.10 (code not shown).

This sort of facility provides great power and flexibility for producing complex plots such as the Trellis plots produced by the lattice system (Fig. 3.11) and more besides.

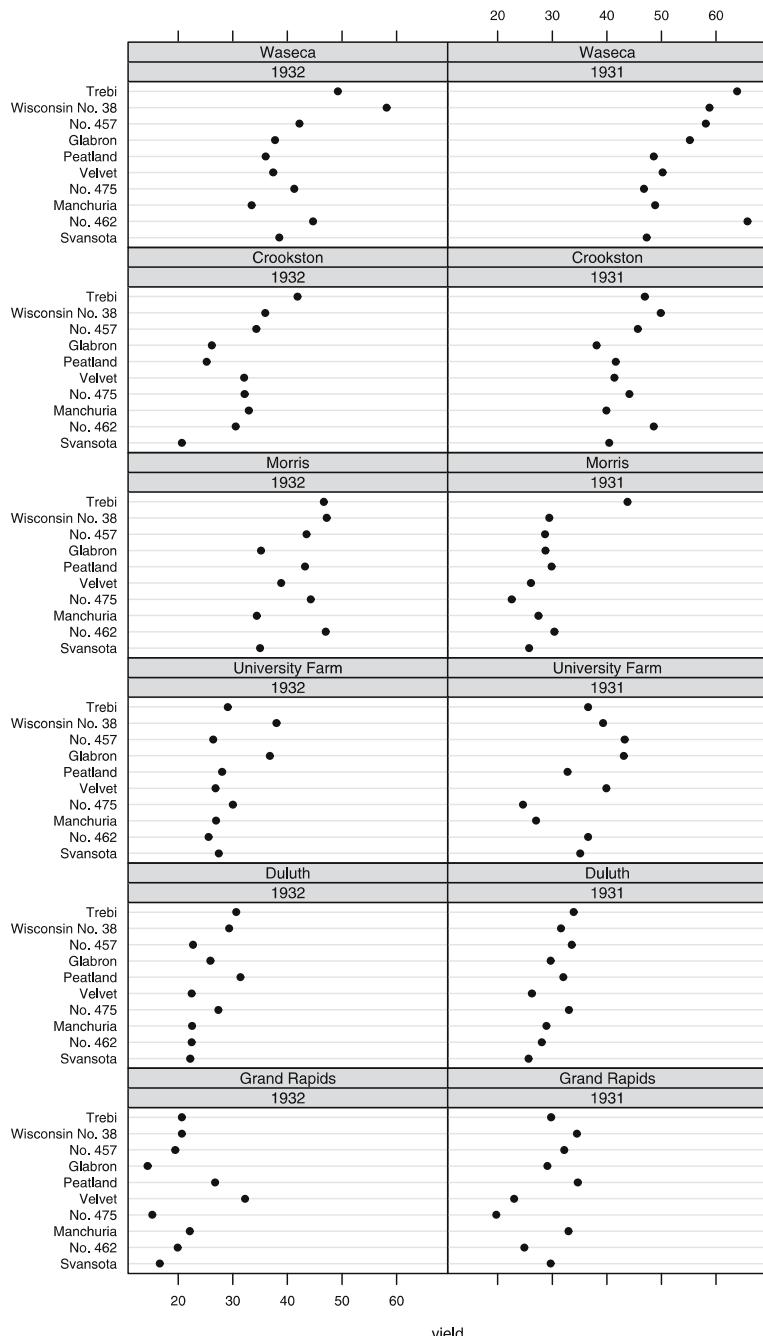


Figure 3.11. Example of a complex Trellis plot. The data are yields of several different varieties of barley at six sites, over 2 years. The plot consists of 12 *panels*, one for each year at each site. Each *panel* consists of a *dotplot* showing yield for a particular site in a particular year and a *strip* showing the year and the name of the site

Graphical Layout

The production of a complex plot involves positioning multiple elements within multiple coordinate systems. The arrangement of output within a coordinate system is typically very explicit; for example, a data symbol is drawn at a precise location and is a fixed proportion of the plotting region in size. By contrast, the arrangement of coordinate systems (or entire plots) relative to each other is more implicit. These arrangements are more along the lines of a number of rows and columns of plots (and let the software figure out exactly what that means in terms of the size and location of the plots on the page). These sorts of arrangements have echoes of typesetting or page-layout operations like those in L^AT_EX (Lamport, 1994) or HTML (Raggett et al., 1999), or even the generation of GUI components such as Java layout managers (Using Layout Managers, 2005). It is therefore useful for a statistical graphics system to provide a means for defining implicit arrangements of elements.

In R there is the concept of a ‘layout’ (Murrell, 1999) (a simple example was given in Sect. 3.2.2). A layout divides a rectangular region into rows and columns, each with a different height or width if desired. In the grid system, a viewport can be positioned relative to a layout rather than via an explicit location and size. For example, the following code creates a viewport with a layout that defines a central region so that the margins around the central region are guaranteed to be identical on all sides and are one quarter of the minimum of the width and height of the central region.

```
> pushViewport(viewport(layout=grid.layout(3, 3,
  widths=c(1, 4, 1),
  heights=c(1, 4, 1),
  respect=rbind(c(1, 0, 1),
    c(0, 0, 0),
    c(1, 0, 1)))))
```

This next code shows another viewport being positioned in the central region of the layout (Fig. 3.12). The location and size of this viewport will depend on the size and shape of the parent viewport that defined the layout.

```
> pushViewport(viewport(layout.pos.col=2,
  layout.pos.row=2))
```

With the ability to nest viewports, it is possible to specify complex *implicit* arrangements of graphical elements in R (this is how the panels are arranged in a lattice plot).

Transformations in Statistical Graphics

An important difference between transformations in a general graphics language and transformations in statistical software is that statistical software does not apply transformations to all output. This arises from the difference between statistical graphics and general graphics images (art). A good example is that in PostScript or SVG the current transformation applies to text as well as all other shapes. In particular, if the current transformation scales output, all text is scaled. This is not desirable when drawing a statistical plot because we would like the text to be readable, so in statistical graphics, transformations apply to the locations of output and the size of shapes such as rectangles and lines, but text is sized separately (Fig. 3.13).

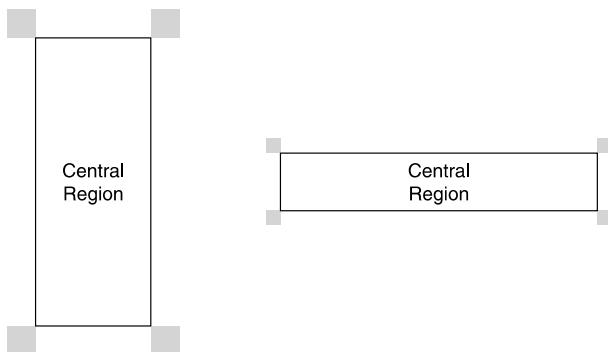


Figure 3.12. Two views of a layout that defines a central region with equal-sized margins all around (indicated by the grey rectangles). The location and shape of the central region depend on the size and shape of the ‘page’ which the layout is applied to; the left-hand page is tall and thin and the right-hand page is short and wide

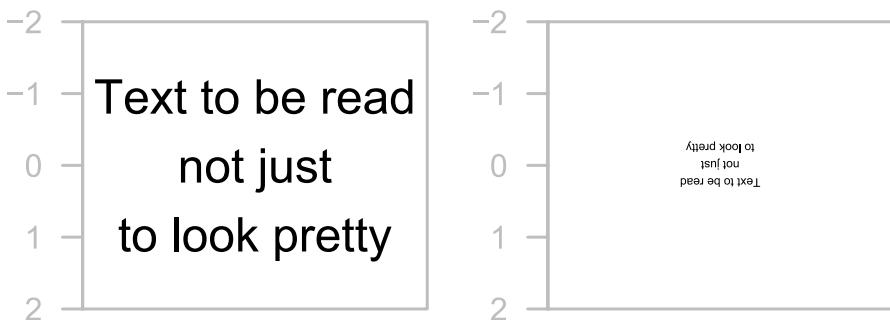


Figure 3.13. The difference between transformations in statistical graphics (*left*) and a general graphics language (*right*). In statistical graphics, the location of the text depends on the coordinate system, but the size of text is controlled separately from coordinate-system transformations. In a general graphics system, all output, including text size, is affected by the current transformation; in this case, the text gets flipped upside down and drawn one-quarter of the size of normal text

Combining Graphical Elements

3.3.2

In addition to allowing the user to compose basic graphics shapes and position them flexibly, a statistical graphics system should allow the user to ‘record’ a composition of graphics shapes. For example, the user should be able to write a function that encapsulates a series of drawing operations. This does two things: the complete set of operations becomes easily available for other people to use, and the function represents a higher-level graphical element that can be used as part of further compositions.

3.3.3

The User Interface

Extending a system is one area where the user interface is crucial. In almost all cases, an extensible system must provide a *language* for developing new graphics. In other words, you must write code (type commands) to extend a system. This is not to say that it is impossible to produce a graphical programming interface (see, for example, the ViSta system; Young, 1996), but a command line offers by far the best environment for power and flexibility. As an absolute minimum, a GUI must provide some way to record code equivalents of GUI actions.

Another detail is that the language for extending the system should ideally be the same language that is used to develop the system. This has two implications: first, the user has *full* access to the graphics system and, second, a scripting language, such as R or Python, is preferable to a ‘heavy-duty’ language such as C or Java because scripting languages are easier to get started with.

3.4

Other Issues

This section draws together a number of issues that overlap with the production of static graphics but are described in more detail elsewhere.

3.4.1

3-D Plots

Static 3-D plots have limited usefulness because 3-D structures are often difficult to perceive without motion. Nevertheless, it is important to be able to produce 3-D images for some purposes. For example, a 3-D plot can be very effective for visualizing a prediction surface from a model.

R provides only simple functionality for drawing 3-D surfaces via the `persp()` function, but the `rgl` (Adler, 2004) add-on package provides an interface to the powerful OpenGL 3-D graphics system (Schreiner, 1999).

3.4.2

Speed

In dynamic and interactive statistical graphics, speed is essential. Drawing must be as fast as possible in order to allow the user to change settings and have the graphics update in real time. In static graphics, speed is less of an issue; achievability of a particular result is more important than how long it takes to achieve it. It is acceptable for a plot to take on the order of seconds to draw rather than milliseconds.

This speed allowance is particularly important in terms of the user interface. For example, in R a lot of graphics code is written in interpreted R code (which is much slower than C code). This makes it easier for users to see the code behind graphics functions, to possibly modify the code, and even to write their own code for graphics.

Nevertheless, a limit is still required because the time taken to draw a single plot can be multiplied many times when producing plots of a large number of observations and when running batch jobs involving a large number of plots.

In R, complex plots, such as Trellis plots produced by the lattice package, can be slow enough to see individual panels being drawn, but most users find this acceptable. The entire suite of figures for a medium-sized book can still be generated in much less than a minute.

Output Formats

3.4.3

When producing plots for reports, it is necessary to produce different formats depending on the format of the report. For example, reports for printing are best produced using PostScript or PDF (Bienz and Cohn, 1993) versions of plots, but for publication on the World Wide Web, it is still easiest to produce some sort of raster format such as PNG. There are many excellent pieces of software for converting between graphics formats, which reduces the need for statistical graphics software to produce output in many formats; simply produce whatever format the statistical graphics software supports and then convert it externally.

Nevertheless, there are still some reasons for statistical graphics software to support multiple formats. One example is that software can raise the bar for the lowest-common-denominator format. For example, R performs clipping of output for formats that do not have their own clipping facilities (e.g. the FIG format; Sutanthavibul, 1985). Another example is that some formats, especially modern ones, provide features that are unavailable in other formats, such as transparency, hyperlinks and animation. It is not possible to convert a more basic format into a more sophisticated format without adding information. Essentially this says that if you are going to aim for a single format, aim high.

Finally, it is worth noting that a description of a plot in the original language of a statistical graphics software system is a viable and important persistent storage option. For example, when producing plots with R, it is advisable to record the R code that was used to produce the plot in addition to saving the plot in any ‘traditional’ formats such as PDF or PostScript. One important advantage with retaining such a high-level format is that it is then possible to modify the image using high-level statistical graphics concepts. For example, an extra text label can be positioned relative to the scales on a plot by modifying the original R code, but this sort of manipulation would be inconvenient, inaccurate and hard to automate if you had to edit a PDF or PostScript version of the plot.

Data Handling

3.4.4

The description of statistical graphics software in this chapter has largely ignored the issue of where the data come from. On one hand, this is deliberate because by separating data from graphics there is a greater flexibility to present any data using any sort of graphic.

However, we should acknowledge the importance of functionality for generating, importing, transforming and analysing data. Without data, there is nothing interesting to plot.

In an ideal situation, statistical graphics facilities are provided as part of a larger system with data-handling features, as is the case with R.

3.5

Summary

Statistical graphics software should provide a straightforward way to produce complete plots. It should be possible to customize all aspects of the plot, add extra output to the plot and extend the system to create new types of plots.

Statistical graphics software can be thought of as an extension of a sophisticated graphics language, providing a fully featured graphics system, a programming language and extensions to specifically support statistical graphics.

References

- Adler, D. (2004). *rgl: 3D visualization device system (OpenGL)*. R package version 0.64-13. <http://wsopuppenkiste.wiso.uni-goettingen.de/~dadler/rgl>.
- Becker, R.A., Cleveland, W.S. and Shyu, M.-J. (1996). The visual design and control of Trellis display, *Journal of Computational and Graphical Statistics* 5:123–155.
- Bienz, T. and Cohn, R. (1993). *Portable Document Format Reference Manual*, Addison-Wesley, Reading, MA, USA.
- Cleveland, W.S. (1985). *The Elements of Graphing Data*, Wadsworth Publ. Co.
- Cleveland, W.S. (1993). *Visualizing Data*, Hobart Press.
- Cleveland, W.S. and McGill, R. (1987). Graphical perception: The visual decoding of quantitative information on graphical displays of data (C/R, pp. 210–229), *Journal of the Royal Statistical Society, Series A, General* 150:192–210.
- Ferraiolo, J., Jun, F. and Jackson, D. (2003). Scalable vector graphics (SVG) 1.1, <http://www.w3.org/TR/SVG11/>.
- Fox, J. (2005). *Rcmdr: R Commander*. R package version 0.9-17. <http://socserv.socsci.mcmaster.ca/jfox/Misc/Rcmdr/>.
- Helbig, M., Urbanek, S. and Theus, M. (2004). Java GUI for R, <http://stats.math.uni-augsburg.de/JGR/index.shtml>.
- Iacus, S. and Urbanek, S. (2004). Cocoa-based GUI for R for Mac OS X, <http://cran.r-project.org/bin/macosx/>.
- Inc., C.A.S. (1990). *PostScript language reference manual (2nd ed.)*, Addison-Wesley, Boston, MA, USA.
- Lamport, L. (1994). *LaTeX: A document preparation system*, Addison-Wesley.
- Murrell, P. (2002). The grid graphics package, *R News* 2(2):14–19. <http://CRAN.R-project.org/doc/Rnews/>.
- Murrell, P.R. (1999). Layouts: A mechanism for arranging plots on a page, *Journal of Computational and Graphical Statistics* 8:121–134.

-
- Porter, T. and Duff, T. (1984). Compositing digital images, *SIGGRAPH '84 Conference Proceedings, Association for Computing Machinery*, Vol. 18.
- R Development Core Team (2005). *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria. <http://www.R-project.org>
- Raggett, D., Hors, A.L. and Jacobs, I. (1999). HTML 4.01 specification, <http://stat18.stat.auckland.ac.nz/stats220/resource/html401/cover.html>.
- Robbins, N. (2005). *Creating More Effective Graphs*, Wiley.
- Sarkar, D. (2002). Lattice, *R News* 2(2):19–23. <http://CRAN.R-project.org/doc/Rnews/>.
- Schreiner, D. (1999). *OpenGL Reference Manual: The Official Reference Document to OpenGL, Version 1.2*, Addison-Wesley Longman.
- Sutanthavibul, S. (1985). FIG: Facility for interactive generation of figures, <http://duke.usask.ca/~macphed/soft/fig/FORMAT3.2.txt>.
- Swayne, D.F., Lang, D.T., Buja, A. and Cook, D. (2003). GGobi: evolving from XGobi into an extensible framework for interactive data visualization, *Computational Statistics and Data Analysis* 43:423–444.
- Theus, M. (2002). Interactive data visualization using Mondrian, *Journal of Statistical Software*, 7(11). <http://www.jstatsoft.org/v07/i11/>.
- Tufte, E. (2001). Graphing Software in ASK E.T., http://www.edwardtufte.com/bboard/q-and-a?topic_id=1.
- Tufte, E.R. (1989). *The Visual Display of Quantitative Information*, Graphics Press.
- Using Layout Managers. in The Java Tutorial (2005). <http://java.sun.com/docs/books/tutorial/uiswing/layout/using.html>.
- Young, F.W. (1996). ViSta: The visual statistics system, *Technical Report 94-1(c)*, UNC L.L. Thurstone Psychometric Laboratory Research Memorandum.
- Wilkinson, L. (1999). *The Grammar of Graphics*, Springer.
- Wilkinson, L. (2004). The grammar of graphics, in J. Gentle, W. Härdle and Y. Mori (eds), *Handbook of Computational Statistics: concepts and methods*, Springer.

Data Visualization Through Their Graph Representations

George Michailidis

4.1	<i>Introduction</i>	104
4.2	<i>Data and Graphs</i>	104
4.3	<i>Graph Layout Techniques</i>	106
	Force-directed Techniques	109
	Multidimensional Scaling	110
	The Pulling Under Constraints Model.....	113
	Bipartite Graphs	114
4.4	<i>Discussion and Concluding Remarks</i>	118

Introduction

The amount of data and information collected and retained by organizations and businesses is constantly increasing, due to advances in data collection, computerization of transactions, and breakthroughs in storage technology. Further, many attributes are also recorded, resulting in very high-dimensional data sets. Typically, the applications involve large-scale information banks, such as data warehouses that contain interrelated data from a number of sources. Examples of new technologies giving rise to large, high-dimensional data sets are high-throughput genomic and proteomic technologies, sensor-based monitoring systems, etc. Finally, new application areas such as biochemical pathways, web documents, etc. produce data with inherent *structure* that cannot be simply captured by numbers.

To extract useful information from such large and structured data sets, a first step is to be able to *visualize* their structure, identifying interesting patterns, trends, and complex relationships between the items. The main idea of visual data exploration is to produce a representation of the data in such a way that the human eye can gain insight into their structure and patterns. Visual data mining techniques have proven to be of particularly high value in exploratory data analysis, as indicated by the research in this area (Eick and Wills 1993a, b).

In this exposition, we focus on the visual exploration of data through their graph representations. Specifically, it is shown how various commonly encountered structures in data analysis can be represented by graphs. Special emphasis is paid to categorical data for which many commonly used plotting techniques (scatterplots, parallel coordinate plots, etc.) prove problematic. Further, a rigorous mathematical framework based on optimizing an objective function is introduced that results in a graph layout. Several examples are used to illustrate the techniques.

Data and Graphs

Graphs are useful entities since they can represent relationships between sets of objects. They are used to model complex systems (e.g., computer and transportation networks, VLSI and Web site layouts, molecules, etc.) and to visualize relationships (e.g., social networks, entity-relationship diagrams in database systems, etc.). In statistics and data analysis, we usually encounter them as dendrograms in cluster analysis, as trees in classification and regression, and as path diagrams in structural equation models and Bayesian belief diagrams. Graphs are also very interesting mathematical objects, and a lot of attention has been paid to their properties. In many instances, the right picture is the key to understanding. The various ways of visualizing a graph provide different insights, and often hidden relationships and interesting patterns are revealed. An increasing body of literature is considering the problem of how to draw a graph [see for instance the book by Di Battista et al. (1998) on graph drawing, the Proceedings of the Annual Conference on Graph Drawing, and the annotated bibliography by Di Battista et al. (1994)]. Also, several problems in distance geometry

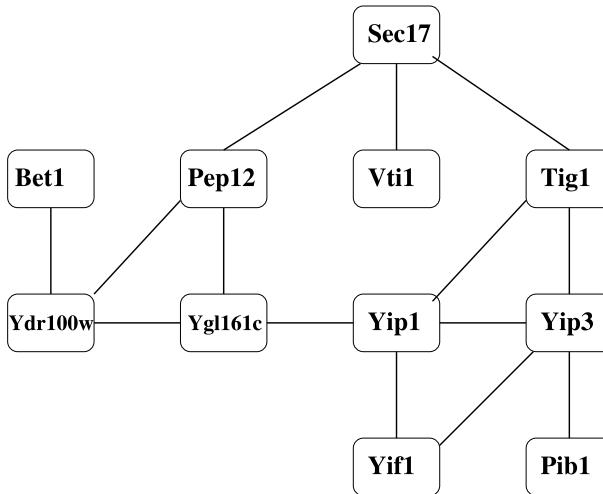


Figure 4.1. Graph representation of a small protein interaction network, with nodes corresponding to proteins and links to their physical interactions

and in graph theory have their origin in the problem of graph drawing in higher-dimensional spaces. Of particular interest in this study is the representation of data sets through graphs. This bridges the fields of multivariate statistics and graph drawing.

Figure 4.1 shows the graph representation of the protein interaction network implicated in the membrane fusion process of vesicular transport for yeast (Ito et al., 2000), with the nodes representing the proteins and the links the physical interactions between them.

However, graphs are also capable of capturing the structure of data commonly encountered in statistics, as the following three examples show. The first example deals with a contingency table (Table 4.1 and Fig. 4.2), where the nodes correspond to the categories and the weighted links represent the frequencies.

The second example deals with a small correlation matrix (Table 4.2 and Fig. 4.3), which can also be represented by a weighted graph, with the nodes representing the variables and the links the strength of the correlation.

Table 4.1. Contingency table of 5387 school children from Caithness, Scotland, classified according to two categorical variables, hair and eye color (Fisher, 1938)

Eye color	Hair color				
	Fair	Red	Medium	Dark	Black
Light	688	116	584	188	4
Blue	326	38	241	110	3
Medium	343	84	909	412	26
Dark	98	48	403	681	85

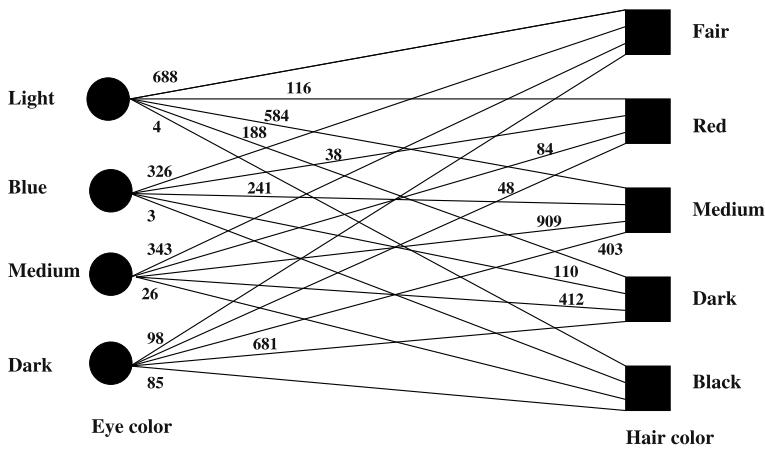


Figure 4.2. Weighted graph representation of a contingency table

Table 4.2. A small correlation matrix for four variables

	Var 1	Var 2	Var 3	Var 4
Var 1	1.00			
Var 2	0.72	1.00		
Var 3	0.43	0.84	1.00	
Var 4	0.96	0.57	0.05	1.00

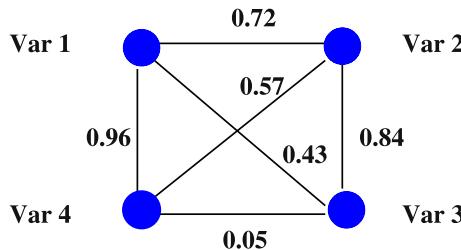


Figure 4.3. Representation of a small correlation matrix by a weighted graph

Another interesting data structure that can be represented successfully by a graph is that corresponding to a multivariate categorical data set, as the following example attests (Table 4.3). The data on 21 sleeping bags and their characteristics come from Prediger (1997) and have also been discussed in Michailidis and de Leeuw (2001).

4.3 Graph Layout Techniques

The problem of graph drawing/layout has received a lot of attention from various scientific communities. It is defined as follows: given a set of nodes connected by a set

Table 4.3. The superindicator matrix representation (Gifi, 1990) of a categorical data set

	Sleeping Bag	Price	Cheap	Not expensive	Expensive	Down fibers	Synthetic fibers	Good	Acceptable	Bad
1	One Kilo Bag	1	0	0	0	1	1	0	0	0
2	Sund	1	0	0	0	1	0	0	0	1
3	Kompakt Basic	1	0	0	0	1	1	0	0	0
4	Finmark Tour	1	0	0	0	1	0	0	0	1
5	Interlight Lyx	1	0	0	0	1	0	0	0	1
6	Kompakt	0	1	0	0	1	0	0	1	0
7	Touch the Cloud	0	1	0	0	1	0	0	1	0
8	Cat's Meow	0	1	0	0	1	1	0	0	0
9	Igloo Super	0	1	0	0	1	0	0	0	1
10	Donna	0	1	0	0	1	0	0	1	0
11	Tyin	0	1	0	0	1	0	0	1	0
12	Travellers Dream	0	1	0	1	0	1	0	0	0
13	Yeti Light	0	1	0	1	0	1	0	0	0
14	Climber	0	1	0	1	0	0	0	1	0
15	Viking	0	1	0	1	0	1	0	0	0
16	Eiger	0	0	1	1	0	0	0	1	0
17	Climber light	0	1	0	1	0	1	0	0	0
18	Cobra	0	0	1	1	0	1	0	0	0
19	Cobra Comfort	0	1	0	1	0	0	0	1	0
20	Foxfire	0	0	1	1	0	1	0	0	0
21	Mont Blanc	0	0	1	1	0	1	0	0	0

of edges, identify the positions of the nodes in some space and calculate the curves that connect them. Hence, in order to draw a graph, one has to make the following two choices: (i) selection of the space and (ii) selection of the curves. For example, grid layouts position the nodes at points with integer coordinates, while hyperbolic layouts embed the points on a sphere. Most graph drawing techniques use straight lines between connected nodes, but some use curves of a certain degree (Di Battista et al., 1998).

Many layout algorithms are based on a set of *aesthetic* rules that the drawing needs to adhere to. Popular rules are that nodes and edges must be evenly distributed, edges should have similar lengths, edge crossings must be kept to a minimum, etc. Some of these rules are important in certain application areas. Further, many of these rules lead to a corresponding optimization problem, albeit intractable in certain cases. For example, the edge-crossing minimization is provably NP-hard and hence computationally intractable (Di Battista et al., 1998). In many cases, a basic layout is obtained by a computationally fast algorithm, and the resulting drawing is postprocessed to

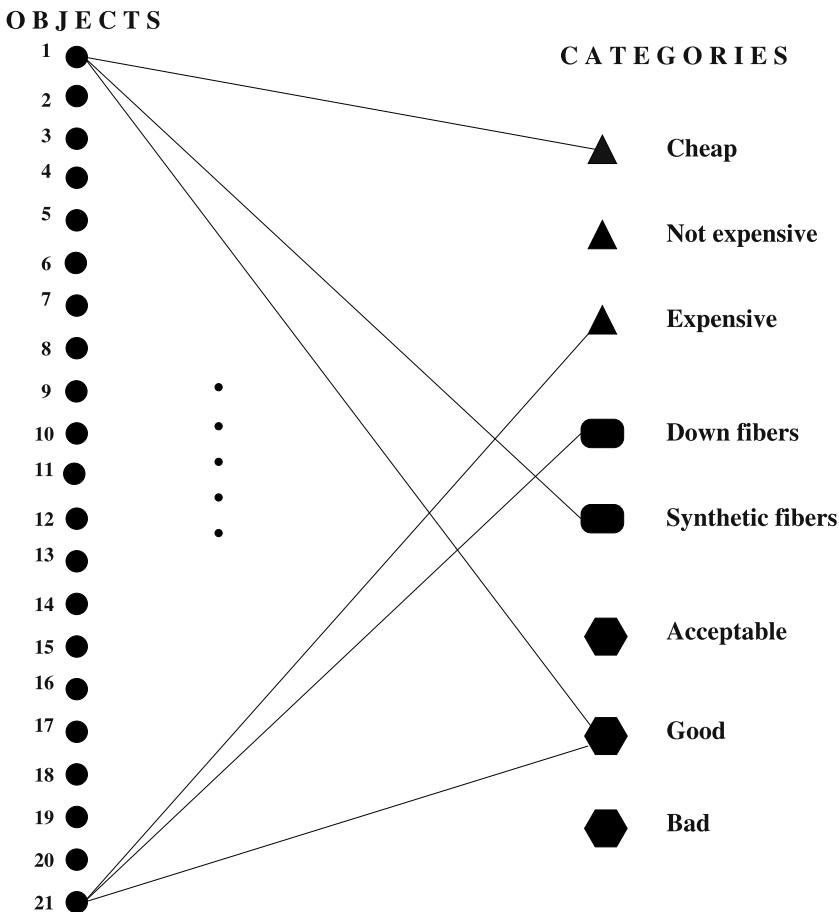


Figure 4.4. Graph representation of the sleeping bag data set presented in Table 4.2, with the *left set of nodes* corresponding to the objects (sleeping bags), the *right set of nodes* to the categories of the three attributes (price, fiber, quality), and selected *edges* capturing the relationship between objects and categories.

adhere to such aesthetic rules. The latter strategy proves particularly useful in the presence of large graphs and is adopted by several graph drawing systems, such as Niceworks (Wills, 1997), GVF (Herman et al., 2000), and H3Viewer (Muentzer, 1998). Many systems also allow manual postprocessing of the resulting layout; see for example the Cytoscape visualization system (www.cytoscape.org).

The general problem of graph drawing discussed in this paper is to represent the edges of a graph as points in \mathbb{R}^p and the vertices as lines connecting the points. Graph drawing is an active area in computer science, and it is very ably reviewed in the recent book by Di Battista et al. (1998). The choice of \mathbb{R}^p is due to its attractive underlying geometry and the fact that it renders the necessary computations more manageable.

There are basically two different approaches to making such drawings. In the *metric* or *embedding* approach, one uses the path-length distance defined between the vertices of the graph and tries to approximate these distances by the Euclidean distance between the points. The area of embedding graph-theoretical distances is related to distance geometry, and it has been studied a great deal recently. In this paper, we adopt primarily the *adjacency model*, i.e., we do not emphasize graph-theoretical distances, but we pay special attention to which vertices are adjacent and which are not. Obviously, this is related to distance, but the emphasis is different. We use objective (loss) functions to measure the *quality* of the resulting embedding.

Force-directed Techniques

4.3.1

The class of graph-drawing techniques most useful for data visualization are *force-directed techniques*. This class of techniques borrows an analogy from classical physics, with the vertices being bodies with masses that attract and repel each other due to the presence of springs, or because the vertices have electric charges. This implies that there are ‘physical’ forces pulling and pushing the vertices apart, and the optimal graph layout will be one in which these forces are in equilibrium. An objective (loss) function that captures this analogy is given next:

$$Q(X|A, B) = \sum_{i=1}^n \sum_{j=1}^n a_{ij} \phi(d_{ij}(X)) - \sum_{i=1}^n \sum_{j=1}^n b_{ij} \psi(d_{ij}(X)), \quad (4.1)$$

where the $n \times p$ matrix X contains the coordinates of the n vertices in \mathbb{R}^p and $d_{ij}(X)$ denotes the distances between points with coordinates x_i and x_j . The weights a_{ij} correspond to those in the adjacency matrix A of the graph G , while the pushing weights $B = \{b_{ij}\}$ could be derived either from the adjacency matrix or from an external constraint. Finally, the functions $\phi(\cdot)$ and $\psi(\cdot)$ are transformations whose role is to impose some aesthetic considerations on the layout. For example, a convex ϕ function will reinforce large distances by rendering them even larger and thus enable one to detect unique features in the data, while a concave transformation will dampen the effect of isolated vertices. Notice that this framework can accommodate both simple (i.e., $a_{ij} \in \{0, 1\}$) and weighted (i.e., $a_{ij} \geq 0$) graphs. A popular force-directed technique that employs this pull-push framework is discussed in Di Battista et al. (1998), where the pulling is done by springs obeying Hooke’s law (i.e., the force is proportional to the difference between the distance of the vertices and the zero-energy length of the spring), while the pushing is done by electrical forces following an inverse square law. Variations on this physical theme are used in several other algorithms (Fruchterman and Reingold 1991 and references therein).

Another way of incorporating a pushing component in the above objective function is through a normalization constraint. For example, one can require that $\eta(X) = 1$, and then the objective function takes the form by forming the Lagrangian:

$$Q(X|A) = \sum_{i=1}^n \sum_{j=1}^n a_{ij} \phi(d_{ij}(X)) - \lambda(\eta(X) - 1). \quad (4.2)$$

It then becomes clear that the constraint term in the Lagrangian corresponds to the push component of $Q(\cdot)$. Examples of $\eta(X)$ include $\eta(X) = \text{trace}(X'X)$ or $\eta(X) = \det(X'X)$. Other possibilities include requiring the orthonormality of the points in the layout, such as $X'X = I_p$ or even fixing some of the X s (Tutte, 1963).

Finally, this formulation allows one to incorporate into the force-directed framework the *metric* approach of graph drawing, where one works not with the adjacency matrix of the graph but with a distance matrix defined on the graph G . The goal then becomes to approximate graph-theoretic distances by Euclidean distances. Hence, the goal becomes to minimize

$$Q(X|W) = \sum_{i=1}^n \sum_{j=1}^n w_{ij} \rho(d_{ij}(X)), \quad (4.3)$$

where

$$\rho(d_{ij}(X)) = (\eta(\delta_{ij}) - \eta(d_{ij}(X)))^2, \quad (4.4)$$

where $W = \{w_{ij}\}$ is a set of weights. The δ_{ij} correspond to path-length distances defined on the graph G , whereas the transformation η is usually the identity, the square, or the logarithm. Obviously $\rho(d(X))$ is not increasing and does not pass through zero; nevertheless, by expanding the square it becomes clear that it is equivalent to minimizing $Q(X|W)$ with $\phi(d) = \eta^2(d)$, $\psi(d) = \eta(d)$, and $b_{ij} = 2\eta(\delta_{ij})w_{ij}$. Thus, all points are pulling together, but points with large path-length distances are being pushed apart.

Next we examine in more detail the *metric* or *embedding* approach and the pulling under constraints model, which have proved particularly useful for drawing graphs obtained from data.

4.3.2 Multidimensional Scaling

The *metric* approach previously discussed corresponds to one version of multidimensional scaling (MDS). MDS is a class of techniques where a set of given distances is approximated by distances in low-dimensional Euclidean space. Formally, let $\{\delta_{ij}, i, j = 1, \dots, n\}$ be a set of distances. The goal is to identify the coordinates of n points x_i in \mathbb{R}^p such that the Euclidean distance $d(x_i, x_j) \equiv d_{ij}(X)$ is approximately equal to δ_{ij} .

As mentioned before, for graph-drawing purposes, the δ_{ij} correspond to the shortest path distances defined on a graph G . A discussion of MDS as a graph-drawing technique is provided in Buja and Swaine (2002), where in addition other choices beyond Euclidean space are studied for the embedding space. The least-squares-loss (fit) function (known in the literature as Stress) introduced in Kruskal (1964) has the form

$$\sigma(X) = \sum_{i=1}^n \sum_{j=1}^n w_{ij} (\delta_{ij} - d_{ij}(X))^2 \quad (4.5)$$

that is minimized over X . The w_{ij} are weights that can be chosen to reflect variability, measurement error, or missing data. This is precisely the objective function (4.3) derived from the general framework of force-directed techniques previously introduced and discussed.

A number of variations of (4.5) have appeared in the literature. In McGee (1966), the loss function has weights δ_{ij}^{-2} . The loss function is interpreted as the amount of *physical work* that must be done on elastic springs to stretch or compress them from an initial length δ_{ij} to a final length d_{ij} . On the other hand, the following choice of weights $w_{ij} = \delta_{ij}^{-1}$ is discussed in Sammon (1969).

Minimization of the loss function (4.5) can be accomplished either by an iterative majorization algorithm (Borg and Groenen 1997; De Leeuw and Michailidis 1998) or by a steepest descent method (Buja and Swayne 2002). The latter method is used in the implementation of MDS in the GGobi visualization system (Swayne et al., 1998). A 2-D MDS solution for the sleeping bag data set is shown in Fig. 4.5. It can be seen that the solution spreads the objects in the data set fairly uniformly in the plane, and edge crossings are avoided.

We discuss next a fairly recent application of MDS. In many instances, the data exhibit nonlinearities, i.e., they lie on a low-dimensional manifold of some curvature. This has led to several approaches that still rely on the embedding (MDS) approach for visualization purposes but appropriately alter the input distances $\{\delta_{ij}\}$. A pop-

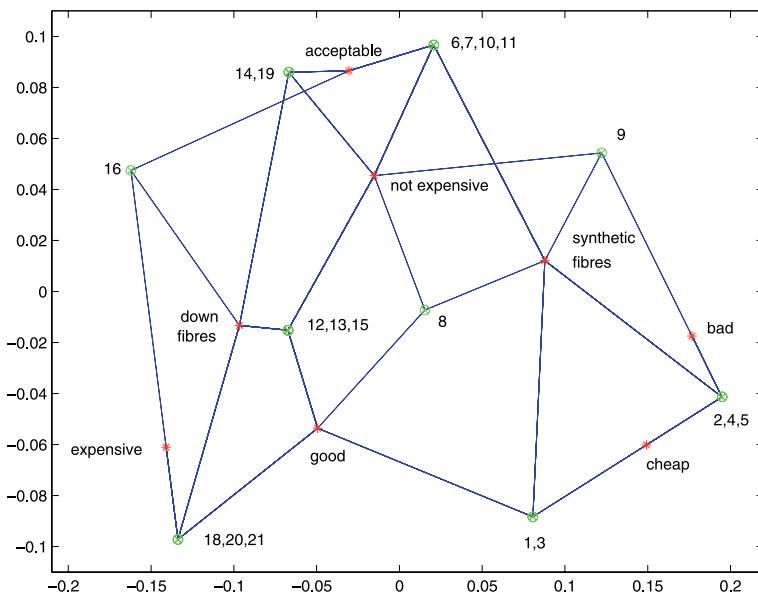


Figure 4.5. MDS representation of sleeping bag data set based on χ^2 -distances. Due to the discrete nature of the data, multiple objects are mapped onto the same location, as shown in the plot. Further, for reference purposes, the categories to which the sleeping bags belong have been added to the plot at the centroids of the object points

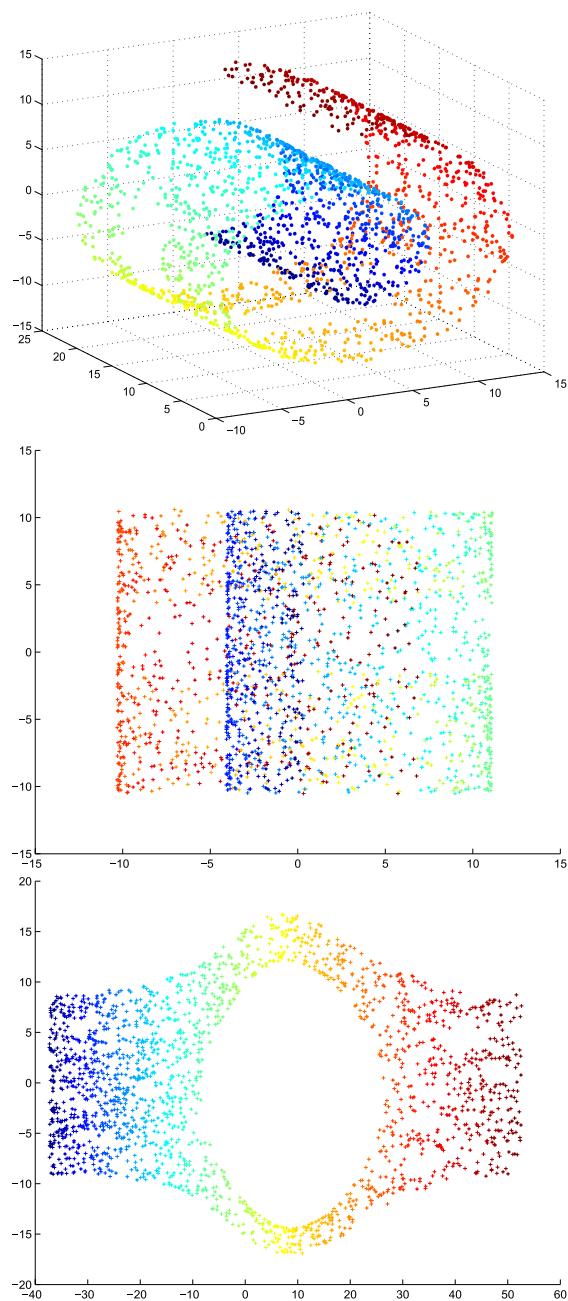


Figure 4.6. [This figure also appears in the color insert.] *Top panel:* original data (2000 data points) arranged along a nonlinear surface (Swiss Roll). *Middle panel:* 2-D MDS representation based on a complete weighted graph. *Bottom panel:* 2-D MDS representation based on 20 nearest-neighbor graph

ular and fairly successful nonlinear embedding technique is the Isomap algorithm (Tenenbaum et al., 2000). The main algorithm computes for each point in the data set a K -nearest neighbor graph and then stitches them together in the adjacency matrix. It then calculates distances using the resulting graph and then applies MDS. The main idea in the first step of the construction is to capture well the local geometry of the data. An illustration of the idea based on the Swiss Roll data set is shown next. Specifically, 2000 random points lying on a roll have been generated and their Euclidean pairwise distances computed. In addition, a graph that connects data points only with their closest 20 neighbors in the Euclidean metric was computed and shortest path distances calculated. Subsequently, MDS was applied to both distance matrices and a 2-D embedding obtained. The coloring scheme shows that straightforward MDS does not capture the underlying geometry of the roll (since the points do not follow their progression on the roll, blue, cyan, green, etc.), whereas the first dimension using the Isomap algorithm recovers the underlying structure. The hole in the middle is mostly due to the low density of orange points.

The Pulling Under Constraints Model

4.3.3

In this model, the similarity of the nodes is important. In the case of a simple graph, only connections between nodes are taken into consideration, whereas in a weighted graph, edges with large weights play a more prominent role. However, the normalization constraint, as discussed in Sect. 3, pushes points apart and avoids the trivial solution of all points collapsing to the origin. This model, under various distance functions, has been studied in a series of papers by Michailidis and de Leeuw (2001, 2004, 2005). We examine next the case of squared Euclidean distances, where $\phi(d(X)) \equiv d_{ij}^2(X)/2$, which turns out to be particularly interesting from a data visualization point of view. Some algebra shows that the objective function can be written in the following matrix algebra form:

$$Q(X|A) = \text{trace}(X'LX), \quad (4.6)$$

where $L = D - A$ is the graph Laplacian (Chung, 1997), with D being a diagonal matrix containing the row sums of the adjacency matrix A . It can be seen that by minimizing (4.6), nodes sharing many connections would be pulled together, whereas nodes with few connections would end up on the periphery of the layout. For a weighted graph, the larger the weights, the stronger the bond between nodes and hence the more pronounced the clustering pattern.

A normalization constraint that leads to a computationally easy-to-solve problem is $X'DX = I_p$. Some routine calculations show that minimizing (4.6) subject to this constraint corresponds to solving a generalized eigenvalue problem. Further, notice that the solution is not orthogonal in Euclidean space, but in weighted (by D) Euclidean space. Figure 4.7 shows the graph layout for the small protein interactions network shown in Fig. 4.1. It can be seen that proteins PIB1 and BET1 that have very few interactions are located on the periphery of the layout. Moreover, the ‘hub’ pro-

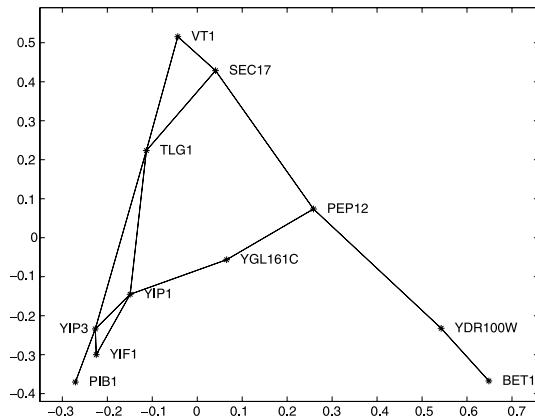


Figure 4.7. Two-dimensional layout of the small protein interaction network shown in Fig. 4.1

teins TLG1 and YIP1 are positioned close to the center of the plot, signifying their central role in this network.

The next example comes from the UCI machine learning repository. The data set consists of features of handwritten numerals (0–9) extracted from a collection of Dutch utility maps. There are 200 patterns per class, and 240 variables characterizing the pixel intensity of the underlying digital image of the digit have been collected. The pixel intensities are categorical and take values in the 0 to 7 range. This is an example where linear techniques such as principal component analysis fail to separate the classes (see top panel of Fig. 4.8).

The next set of plots in Fig. 4.9 shows the layouts of a few large graphs that have been used for testing graph partitioning algorithms (Walshaw, 2003). The first graph is comprised of 4720 vertices and 13 722 edges, the second of 10 240 vertices and 30 380 edges, and the third of 4253 vertices and 12 289 edges. They are derived from computational mechanics meshes and characterized by extreme variations in the mesh density and the presence of “holes.” The layouts shown are based on weighted graphs that were built by considering for each vertex its ten nearest neighbors in the Euclidean metric and calculating exponentially decreasing weights. It can be seen that the layouts capture, to a large extent, the underlying structure of the graphs in terms of density and the presence of “holes.”

4.3.4 Bipartite Graphs

As noted in Sect. 2, the graph representation of a contingency table and of a categorical data set has some special features, namely, the node set V can be partitioned into two subsets. For example, in the case of a contingency table, the categories of one variable form the first subset and those of the other variable the second one. Notice that there are only connections between members of these two subsets. An analogous situation arises in the case of categorical data, where the first subset of nodes corresponds to the objects (e.g., the sleeping bags) and the second subset to the categories

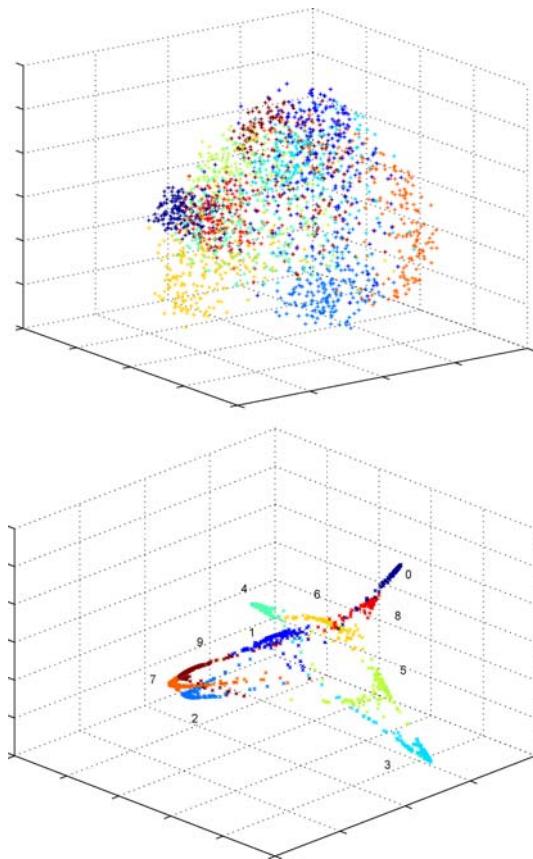


Figure 4.8. [This figure also appears in the color insert.] PCA layout of digits dataset (*top panel*) and the 3-D graph layout (*bottom panel*)

of all the variables. These are two instances where the resulting graph representation of the data gives rise to a *bipartite* graph. A slight modification of the $Q(\cdot)$ objective function leads to interesting graph layouts of such data sets. Let $X = [Z' \ Y']$, where Z contains the coordinates of the first subset of the vertices and Y those of the second subset. The objective function for squared Euclidean distances can then be written as (given the special block structure of the adjacency matrix A)

$$Q(Z, Y|A) = \text{trace}(Z'D_Z Z + Y'D_Y Y - 2Y'AZ), \quad (4.7)$$

where D_Y is a diagonal matrix containing the column sums of A and D_Z another diagonal matrix containing the row sums of A . In the case of a contingency table, both D_Y and D_Z contain the marginal frequencies of the two variables, while for a multivariate categorical data set D_Y contains again the univariate marginals of all the categories of all the variables and $D_Z = JI$ is a constant multiple of the identity matrix, with J denoting the number of variables in the data set. A modification of

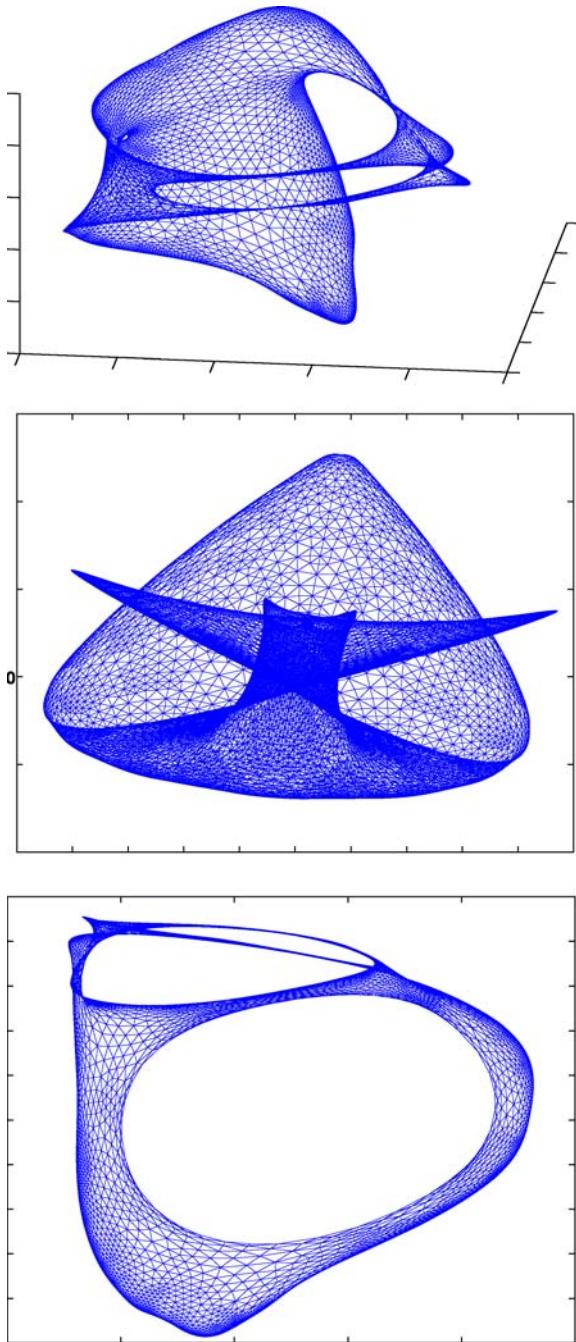


Figure 4.9. Layouts of large graphs derived from computational mechanics meshes and characterized by varying degrees of mesh density

the normalization constraint to this setting, namely, $Z'D_ZZ = I_p$, gives the following solution, which can be obtained through a block relaxation algorithm (Michailidis and de Leeuw, 1998):

$$Z = J^{-1}AY \text{ and } Y = D_Y^{-1}A'XZ.$$

Hence, the optimal solution satisfies the *centroid principle* (Gifi, 1990), which says that the category points in the optimal layout are at the center of gravity of the objects that belong to them. The above graph-drawing solution is known in multivariate analysis for contingency tables as correspondence analysis and for multivariate categorical data sets as multiple correspondence analysis (Michailidis and de Leeuw, 1998).

Figure 4.10 shows the graph layout of the sleeping bags data set. The solution captures the basic patterns in the data set, namely, that there are good-quality, expensive sleeping bags filled with down fibers and cheap, bad-quality sleeping bags filled with synthetic fibers. Further, there exist some sleeping bags of intermediate quality and price filled with either down or synthetic fibers. Notice that the centroid principle resulting from the partitioning of the vertex set proves useful in the interpretation of the layout. Further, the resulting layout is less ‘uniform’ than the one obtained through MDS and thus better captures features of the data.

It is interesting to note that the choice of the distance function coupled with a particular normalization has a significant effect on the aesthetic quality of the resulting

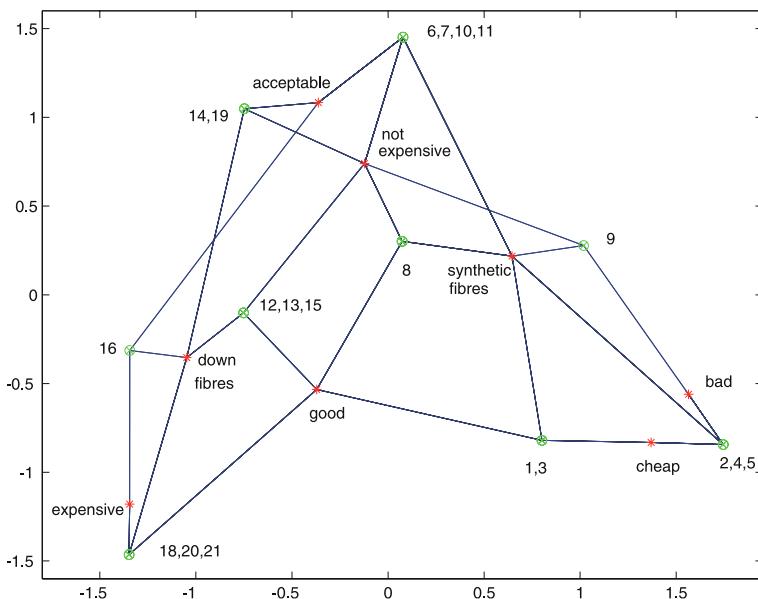


Figure 4.10. Graph layout of sleeping bags data based on objective function (4.7). Due to the discrete nature of the data, multiple objects are mapped on the same location, as shown in the plot. Further, for reference purposes, the categories to which the sleeping bags belong have been added to the plot at the *centroids* of the object points

graph layout. An extreme case occurs when $\phi(d(X)) \equiv d_{ij}(X)$ corresponds to Euclidean distances. Then, under the orthonormality constraint, the solution is rather uninteresting and consists of exactly $p + 1$ points, where p is the dimensionality of the embedding space. A mathematical explanation of this result is given for the 1-D case ($p = 1$) in de Leeuw and Michailidis (2004) and is also illustrated for the higher-dimensional case ($p \geq 2$) in Michailidis and de Leeuw (2005).

4.4

Discussion and Concluding Remarks

In this paper, the problem of data visualization through layouts of their graph representations is considered. A mathematical framework for graph drawing based on force-directed techniques is introduced, and several connections to well-known multivariate analysis techniques such as multidimensional scaling, correspondence, and multiple correspondence analysis are made.

Several extensions that may improve the quality of the graph layout are possible within this general framework. For example, logistic loss functions are explored in de Leeuw (2005), together with the arrangement of the nodes along Voronoi cells. The visualization of several related data sets through multilevel extensions of multiple correspondence analysis are explored in Michailidis and de Leeuw (2000). Finally, a version of multidimensional scaling for data that change over time is discussed in Costa et al. (2004).

Acknowledgement. This work was partially supported by NIH grant 5P41RR018627-03. The author would like to thank Jan de Leeuw for many fruitful discussions and suggestions on the topic over the course of the last 10 years and the editors of the Handbook for many comments that improved the presentation of the material.

References

- Borg, I. and Groenen, P. (1997). *Modern Multidimensional Scaling: Theory and Applications*. Springer, Berlin Heidelberg New York.
- Buja, A. and Swayne, D. (2002). Visualization methodology for multidimensional scaling. *J Classificat*, 19:7–43.
- Chung, F.R.K. (1997). *Spectral Graph Theory*. American Mathematical Society, Providence, RI.
- Costa, J., Patwari, N. and Hero, A.O. (2004). Distributed multidimensional scaling with adaptive weighting for node localization in sensor networks. *ACM J Sensor Network*, 2(1):39–64.
- www.cytoscape.org.
- De Leeuw, J. (2005). *Nonlinear Principal Component Analysis and Related Techniques*. Preprint # 427, Department of Statistics, University of California, Los Angeles.
- De Leeuw, J. and Michailidis, G. (1998). Graph layout techniques and multidimensional data analysis. In: Bruss, F.T., Le Cam, L. (eds) *Game Theory, Optimal Stopp-*

- ping, *Probability and Statistics*. IMS Lecture Notes – Monograph Series, 35:219–248.
- De Leeuw, J. and Michailidis, G. (2004). Weber correspondence analysis: the one-dimensional case. *J Comput Graph Stat*, 13:946–953.
- Di Battista, G., Eades, P., Tamassia, R. and Tollis, I. (1994). Algorithms for drawing graphs: an annotated bibliography. *Comput Geom Theory Appl*, 4:235–282.
- Di Battista, G., Eades, P., Tamassia, R. and Tollis, I. (1998). *Graph Drawing: Algorithms for Geometric Representation of Graphs*. Prentice-Hall, Upper Saddle River, NJ.
- Eick, S.G. and Wills, G.J. (1993a) Navigating large networks with hierarchies. In: *Proceedings of Visualization Conference*, pp. 204–210.
- Eick, S.G. and Wills, G.J. (1993b) High interaction graphics. *Eur J Operat Res*, 84:445–459.
- Fisher, R.A. (1938). The precision of discriminant functions. *Ann Eugen*, 10:422–429.
- Fruchterman, T.M.J. and Reingold, E.M. (1991). Graph drawing by force-directed placement. *Software: practice and engineering*, 21:1129–1164.
- Gifi, A. (1990). *Nonlinear Multivariate Analysis*. Wiley, Chichester.
- www.ggobi.org.
- Herman, I., Melancon, G. and Marshall, M.S. (2000). Graph visualization and navigation in information visualization. *IEEE Trans Visualizat Comput Graph*, 6:24–43.
- Ito, T., Tashiro, K., Muta, S., Ozawa, R., Chiba, T., Nishizawa, M., Yamamoto, K., Kuhara, S. and Sakaki, Y. (2000). Toward a protein-protein interaction map of the budding yeast: a comprehensive system to examine two-hybrid interactions in all possible combinations between the yeast interactions. *Proc Natl Acad Sci USA*, 97:1143–1147.
- Kruskal, J.B. (1964). Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrika*, 29:1–28.
- McGee, V.E. (1966). The multidimensional analysis of elastic distances. *Br J Math Stat Psychol*, 19:181–196.
- Michailidis, G. and de Leeuw, J. (1998). The Gifi system of descriptive multivariate analysis. *Stat Sci*, 13:307–336.
- Michailidis, G. and de Leeuw, J. (2000). Multilevel homogeneity analysis with differential weighting. *Comput Stat Data Anal*, 32:411–442.
- Michailidis, G. and de Leeuw, J. (2001). Data visualization through graph drawing. *Comput Stat*, 16:435–450.
- Michailidis, G. and de Leeuw, J. (2005). Homogeneity analysis using absolute deviations. *Comput Stat Data Anal*, 48:587–603.
- Muentzer, T. (1998). Drawing large graphs with H3Viewer and Site Manager. In: Proceedings of Graph Drawing 98. *Lecture Notes in Computer Science*, vol 1547. Springer, Berlin, Heidelberg, New York, pp. 384–393.
- Prediger, S. (1997). Symbolic objects in formal concept analysis. In: Mineau, G., Fall, A. (eds) *Proceedings of the 2nd international symposium on knowledge, retrieval, use and storage for efficiency*.
- Sammon, J.W. (1969). A nonlinear mapping for data structure analysis. *IEEE Trans Comput*, 18:401–409.

- Swayne, D.F., Cook, D. and Buja, A. (1998). XGobi: interactive dynamic data visualization in the X Window system. *J Comput Graph Stat*, 7:113–130.
- Tenenbaum, J.B., da Silva, V. and Langford, J.C. (2000). A global geometric framework for nonlinear dimensionality reduction. *Science*, 22:2319–2323.
- Tutte, W.T. (1963). How to draw a graph. *Proc Lond Math Soc*, 13:743–767.
- Walshaw, C. (2003). A multilevel algorithm for force-directed graph drawing *J Graph Algor Appl*, 7:253–285.
- Wills, G.J. (1997). NicheWorks – interactive visualization of very large graphs. In: *Proceedings of the conference on graph drawing*. Springer, Berlin Heidelberg New York.

Graph-theoretic Graphics

II.5

Leland Wilkinson

5.1	<i>Introduction</i>	122
5.2	<i>Definitions</i>	122
	Trees	123
5.3	<i>Graph Drawing</i>	124
	Hierarchical Trees	125
	Spanning Trees	131
	Networks	134
	Directed Graphs	134
	Treemaps	135
5.4	<i>Geometric Graphs</i>	136
	Disk Exclusion.....	137
	Disk Inclusion	141
5.5	<i>Graph-theoretic Analytics</i>	143
	Scagnostics	143
	Sequence Analysis	144
	Graph Matching	147
	Conclusion	148

Introduction

This chapter will cover the uses of graphs for making graphs. This overloading of terms is an unfortunate historical circumstance that conflated graph-of-a-function usage with graph-of-vertices-and-edges usage. Vertex-edge graphs have long been understood as fundamental to the development of algorithms. It has become increasingly evident that vertex-edge graphs are also fundamental to the development of statistical graphics and visualizations.

One might assume this chapter is about laying out graphs on a plane, in which vertices are represented by points and edges by line segments. Indeed, this problem is covered in the chapter. Nevertheless, we take the point of view of the *grammar of graphics* (Wilkinson 2005), in which a graphic has an underlying model. Thus, we assume a graph-theoretic graph is any graph that maps aspects of geometric forms to vertices and edges of a graph.

We begin with definitions of graph-theoretic terminology. These definitions are assumed in later sections, so this section may be skipped and used later as a glossary by those not interested in details.

Definitions

A *graph* is a set V together with a relation on V . We usually express this by saying that a graph $G = (V, E)$ is a pair of sets, V is a set of *vertices* (sometimes called *nodes*), and E is a set of *edges* (sometimes called *arcs* or *links*). An edge $e(u, v)$, with $e \in E$ and $u, v \in V$, is a pair of vertices.

We usually assume the relation on V induced by E is symmetric; we call such a graph *undirected*. If the pair of vertices in an edge is ordered, we call G a *directed graph*, or *digraph*. We denote direction by saying, with respect to a node, that an edge is *incoming* or *outgoing*.

A graph is *weighted* if each of its edges is associated with a real number. We consider an unweighted graph to be equivalent to a weighted graph whose edges all have a weight of 1.

A graph is *complete* if there exists an edge for every pair of vertices. If it has n vertices, then a complete graph has $n(n - 1)/2$ edges.

A *loop* is an edge with $u = v$. A *simple* graph is a graph with no loops. Two edges (u, v) and (s, t) are *adjacent* if $u = s$ or $u = t$ or $v = s$ or $v = t$. Likewise, a vertex v is adjacent to an edge (u, v) or an edge (v, u) .

A *path* is a list of successively adjacent, distinct edges. Let $\langle e_1, \dots, e_k \rangle$ be a sequence of edges in a graph. This sequence is called a path if there are vertices $\langle v_1, \dots, v_k \rangle$ such that $e_i = (v_{i-1}, v_i)$ for $i = 2, \dots, k$.

Two vertices u, v of a graph are called *connected* if there exists a path from vertex u to vertex v . If every pair of vertices of the graph is connected, the graph is called connected.

A path is *cyclic* if a node appears more than once in its corresponding list of edges. A graph is cyclic if any path in the graph is cyclic. We often call a directed acyclic graph a DAG.

A *topological sort* of the vertices of a DAG is a sequence of distinct vertices $\{v_1, \dots, v_n\}$. For every pair of vertices v_i, v_j in this sequence, if (v_i, v_j) is an edge, then $i < j$.

A *linear graph* is a graph based on a list of n vertices; its $n-1$ edges connect vertices that are adjacent in the list. A linear graph has only one path.

Two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ are *isomorphic* if there exists a bijective mapping between the vertices in V_1 and V_2 and there is an edge between two vertices of one graph if and only if there is an edge between the two corresponding vertices in the other graph. A graph $G_1 = (V_1, E_1)$ is a *subgraph* of a graph $G_2 = (V_2, E_2)$ if $V_1 \subseteq V_2$ and $E_1 \subseteq E_2 \cap (V_1 \times V_1)$.

The *graph-theoretic distance* (or *geodesic distance*) between connected nodes u and v is the sum of the weights of the edges in any shortest path connecting the nodes. This distance is a metric, namely, symmetry, identity, and the triangle inequality apply.

The *adjacency matrix* for a graph G with n vertices is an $n \times n$ matrix with entries a_{ij} having a value 1 if vertex i is adjacent to vertex j and zero otherwise. The set of eigenvalues of this matrix is called the *graph spectrum*. The spectrum is useful for identifying the dimensionality of a space in which a graph may be embedded or represented as a set of points (for vertices) and a set of connecting lines (for edges).

A *geometric graph* $G_g = [f(V), g(E), S]$ is a mapping of a graph to a metric space S such that vertices go to points and edges go to curves connecting pairs of points. We will discuss various types of geometric graphs in this chapter. When the meaning is clear, we will omit the subscript and refer to G as a geometric graph. The usual mapping is to Euclidean space. Sometimes we will measure and compare the Euclidean distance between points to the graph-theoretic distance between the corresponding vertices of the graph.

A *proximity graph* $G_p = [V, f(V)]$ is a graph whose edges are defined by a proximity function $f(V)$ on points in a space S . The range of $f(V)$ is pairs of vertices. One may regard $f(V)$ as an indicator function in which an edge exists when $g(u, v) < d$, where d is some nonnegative real value and $g()$ is a real-valued function associated with $f()$.

A *random graph* is a function defined on a sample space of graphs. Although random graphs are relevant to statistical data, this chapter will not cover them because of space limitations. Marchette (2004) is a standard reference.

Trees

5.2.1

A *tree* is a graph in which any two nodes are connected by exactly one path. Trees are thus acyclic connected graphs. Trees may be directed or undirected. A tree with one node labeled *root* is a *rooted tree*. Directed trees are rooted trees; the root of a directed tree is the node having no incoming edges.

A *hierarchical tree* is a directed tree with a set of *leaf* nodes (nodes of degree 1) representing a set of objects and a set of *parent* nodes representing relations among the objects. In a hierarchical tree, every node has exactly one parent, except for the

root node, which has one or more children and no parent. Examples of hierarchical trees are those produced by decision-tree and hierarchical clustering algorithms.

A *spanning tree* is an undirected geometric tree. Spanning trees have $n - 1$ edges that define all distances between n nodes. This is a restriction of the $n(n - 1)/2$ edges in a complete graph. A *minimum spanning tree* (MST) has the shortest total edge length of all possible spanning trees.

Ultrametric Trees

If the node-to-leaf distances are monotonically nondecreasing (i.e., no parent is closer to the leaves than its children are), then a hierarchical tree is *ultrametric*. An ultrametric is a metric with a strong form of the triangle inequality, namely,

$$d(x, y) \leq \max [d(x, z), d(y, z)] .$$

In an ultrametric tree, the graph-theoretic distances take at most $n - 1$ possible values, where n is the number of leaves. This is because of the ultrametric three-point condition, which says we can rename any x, y, z such that

$$d(x, y) \leq d(x, z) = d(y, z) .$$

Another way to see this is to note that the distance between any two leaves is determined by the distance of either to the common ancestor.

Additive Trees

Let D be a symmetric n by n matrix of distances d_{ij} . Let T be a hierarchical tree with one leaf for each row/column of D . T is an additive tree for D if, for every pair of leaves (t_i, t_j), the graph theoretic distance between the leaves is equal to d_{ij} . Additive trees rest on a weaker form of the triangle inequality than do ultrametric trees, namely,

$$d(x, y) \leq [d(x, z) + d(y, z)] .$$

Graph Drawing

A graph is *embeddable* on a surface if it can be drawn on that surface so that edges meet only at vertices. A graph is *planar* if it is embeddable on a sphere (and, by implication, the unbounded plane). We can use a theorem by Euler to prove a particular graph is *not* planar, but we can prove a particular graph is planar only by drawing it without edge crossings. Drawing graphs is more than a theoretical exercise, however.

Finding compact planar drawings of graphs representing electrical circuits, for example, is a critical application in the semiconductor industry. Other applications involve metabolic pathways, kinetic models, and communications and transportation networks. Spherical applications involve mapping the physical nodes of the Internet and world trade routes.

The graph-drawing (or graph-layout) problem is as follows. Given a planar graph, how do we produce an embedding on the plane or sphere? And if a graph is not planar, how do we produce a planar layout that minimizes edge crossings? The standard text on graph drawing is Di Battista et al. (1999), which is a comprehensive bibliography. Kruja et al. (2001) give a history of the problem. See also the various years of the *Proceedings of the International Symposium on Graph Drawing*, published by Springer.

Different types of graphs require different algorithms for clean layouts. We begin with trees. Then we discuss laying out networks and directed cyclic graphs. In most examples, the basic input data are nonnumeric. They consist of an unordered list of vertices (node labels) and an unordered list of edges (pairs of node labels). If a graph is connected, then we may receive only a list of edges. If we have a weighted graph, the edge weights may be used in the loss function used to define the layout. We will discuss other forms of input in specific sections.

Hierarchical Trees

5.3.1

Suppose we are given a recursive list of single parents and their children. In this list, each child has one parent and each parent has one or more children. One node, the root, has no parent. This tree is a directed graph because the edge relation is asymmetric. We can encapsulate such a list in a node class:

```
Node{
    Node parent;
    NodeList children;
}
```

Perhaps the most common example of such a list is the directory structure of a hierarchical file system. A display for such a list is called a *tree browser*. Creating such a display is easy. We simply walk the tree, beginning at the root, and indent children in relation to their parents. Figure 5.1 shows an example that uses the most common vertical layout. Figure 5.2 shows an example of a horizontal layout. Interestingly, the “primitive” layout in Fig. 5.1 has been found to be quite effective when compared to more exotic user-interface tree layouts (Kobsa 2004).

Suppose now we are given only a list of edges and told to lay out a rooted tree. To lay out a tree using only an edge list, we need to inventory the parent-child relationships. First, we identify leaves by locating nodes appearing only once in the edge list. We then assign a layer value to each node by finding the longest path to any leaf from that node. Then we begin with the leaves, group children by parent, and align parents above the middle child in each group. After this sweep, we can move leaves up the hierarchy to make shorter branches. Figure 5.3 shows an example using this layout algorithm. The data are adapted from weblogs of a small website. The thicknesses of the branches of the tree are proportional to the number of visitors navigating between pages represented by nodes in the tree.

If the nodes of a tree are ordered by an external variable such as joining or splitting distance, then we may locate them on a scale instead of using paternity to determine ordering. Figure 5.4 shows an example of this type of layout using a cluster tree. The

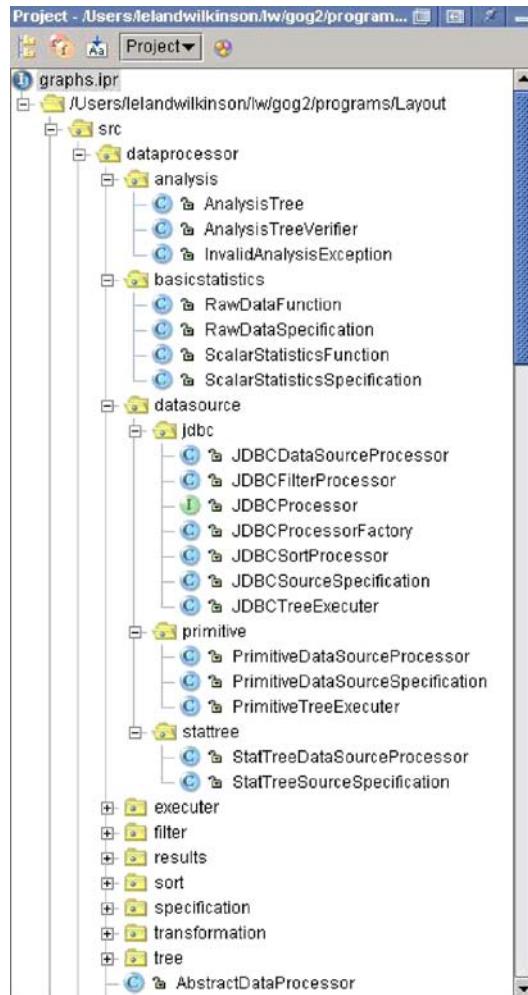


Figure 5.1. Linear tree browser for a Java project

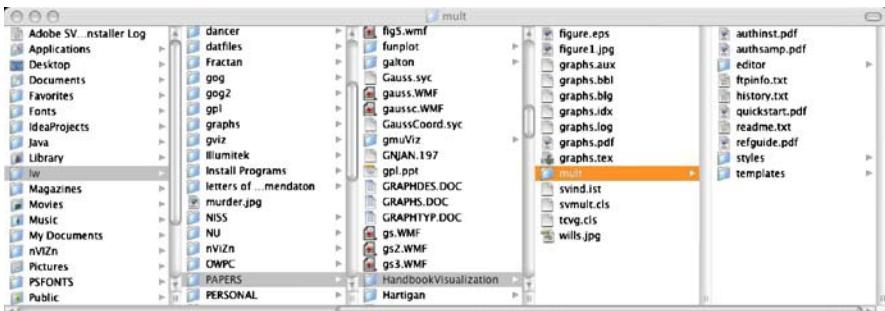


Figure 5.2. Hierarchical tree browser for a file system

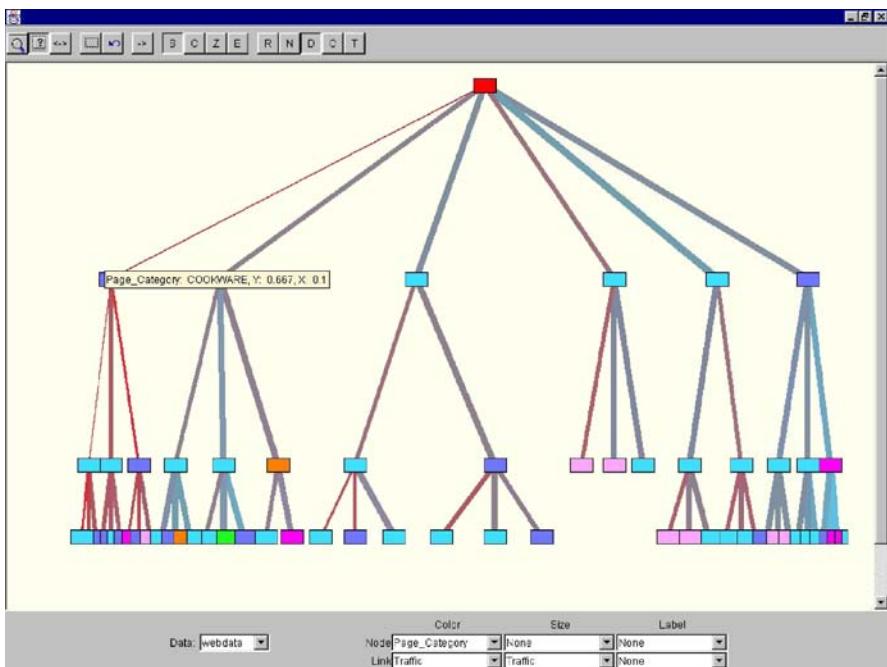


Figure 5.3. Layout of a website tree

data consist of FBI-reported murder rates for US states in 1970. A single linkage cluster analysis with leaves ordered by murder rates produced the tree.

This is an interesting example for several reasons. First, we ordinarily do not think of clustering a set of objects on a single variable. Clustering in one dimension is equivalent to mode hunting or bump hunting, however. Hierarchical clustering (as in this example) can yield a 1-D partitioning into relatively homogeneous blocks. We are seeking intervals in which observations are especially dense. We see, for example, that there are clusters of southern and midwestern states whose murder rates are relatively similar. The mode tree (Minnotte and Scott 1993) is another instance of a tree representation of a 1-D dataset. This tree plots the location of modes in the smoothed nonparametric density estimator as a function of kernel width. Second, a topological sort on a total order is the same as an ordinary sort. That is, by sorting the leaves of this tree on murder values, we have produced a topological sort. For hierarchical clustering trees on more variables there exist more than one topological sort to help organize a tree for viewing. Wilkinson (1999) discusses some of these strategies.

Hierarchical trees with many leaves can become unwieldy in rectangular layouts. In Fig. 5.5 we lay out the same cluster tree in polar coordinates. Other types of circular layouts (e.g. Lamping et al. 1995) can accommodate even larger trees. Circular layouts are popular in biological applications involving many variables because of their space-saving characteristics. It is best, of course, if the polar orientation has an

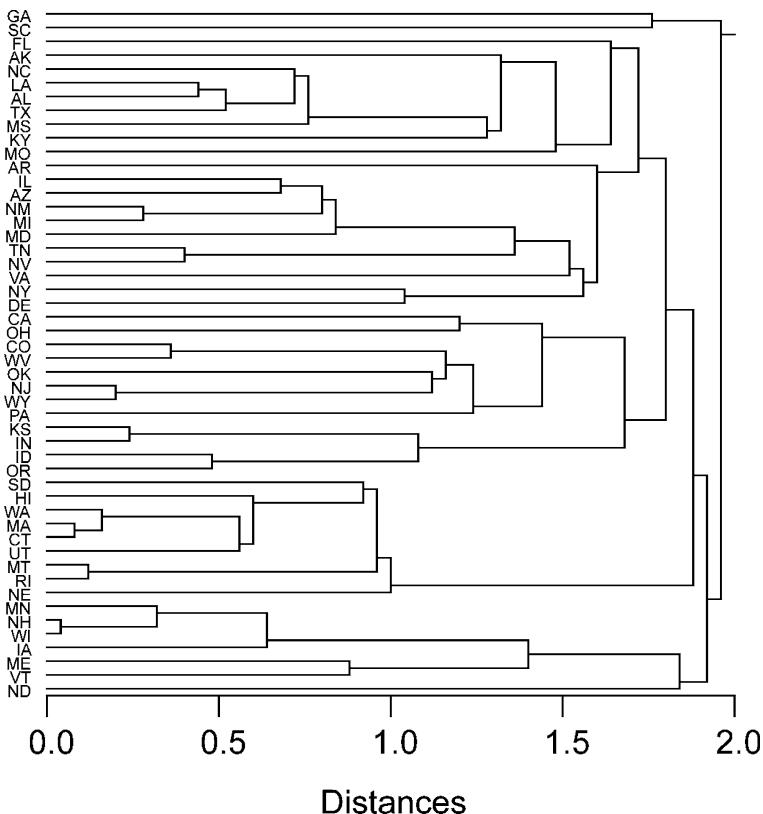


Figure 5.4. Hierarchical cluster tree of US murder rates

intrinsic meaning, but sometimes making room for labels and other information is sufficient justification.

In some cases, the nodes of hierarchical trees may represent nested collections of objects. Classification and regression trees, for example, hierarchically partition a set of objects. For these applications, Wilkinson (1999) invented a tree display called a *mobile*. Figure 5.6 shows an example using data on employees of a bank. Each node contains a dot histogram; each dot represents a bank employee. The dot histograms are hierarchical; a parent histogram aggregates the dots of its children. The horizontal branches represent a beam balancing two sibling dot histograms. By using this model, we highlight the marginality of splits. That is, outlying splits are shifted away from the bulk of the display. This layout is relatively inefficient with regard to space, and it is not well suited to a polar arrangement because the balance metaphor has no meaning in that context.

Figure 5.7 shows an alternative display for classification trees (Urbanek 2003). This form uses the width of branches to represent the size of subsplits. This tree is similar to earlier graphics shown in Kleiner and Hartigan (1981), Dirschedl (1991), Lausen et al. (1994) and Vach (1995).

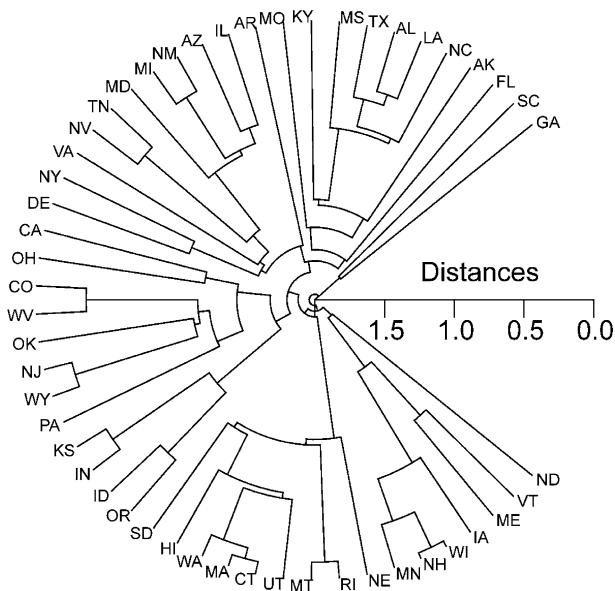


Figure 5.5. Polar hierarchical cluster tree of US murder rates

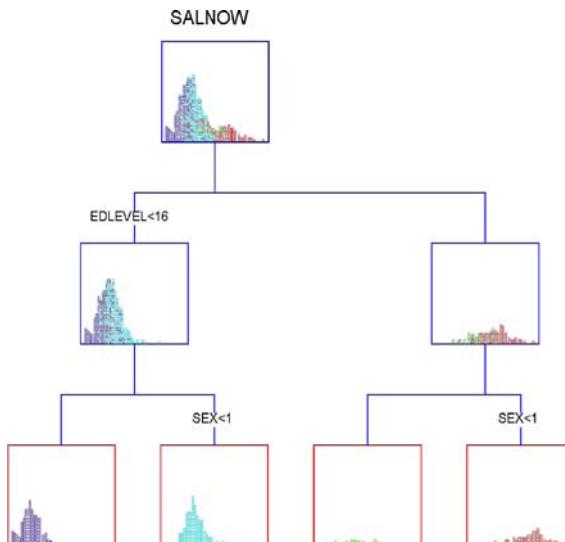


Figure 5.6. [This figure also appears in the color insert.] Mobile of bank employee data

Suppose we have a directed geometric tree with one root having many children. Such a tree may represent a flow from a source at the root branching to sinks at the leaves. Water and migration flows are examples of such a tree. Phan et al. (2005) present a suite of algorithms (including hierarchical cluster analysis and force-directed layout) for rendering a flow tree. The data consist of the geographic location of the

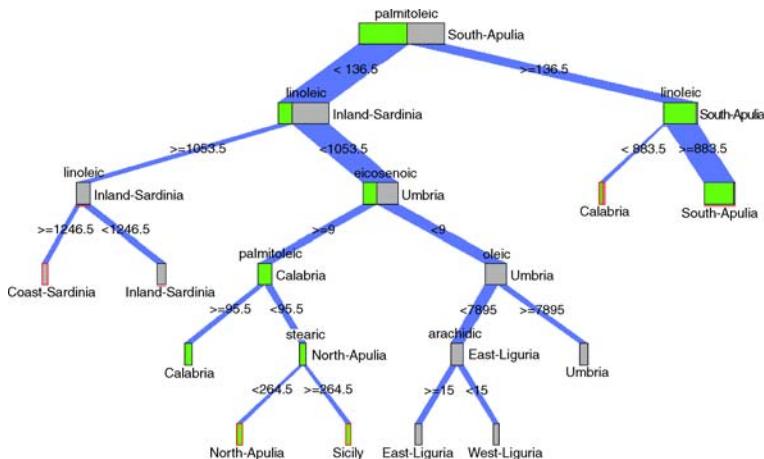


Figure 5.7. Urbanek classification tree

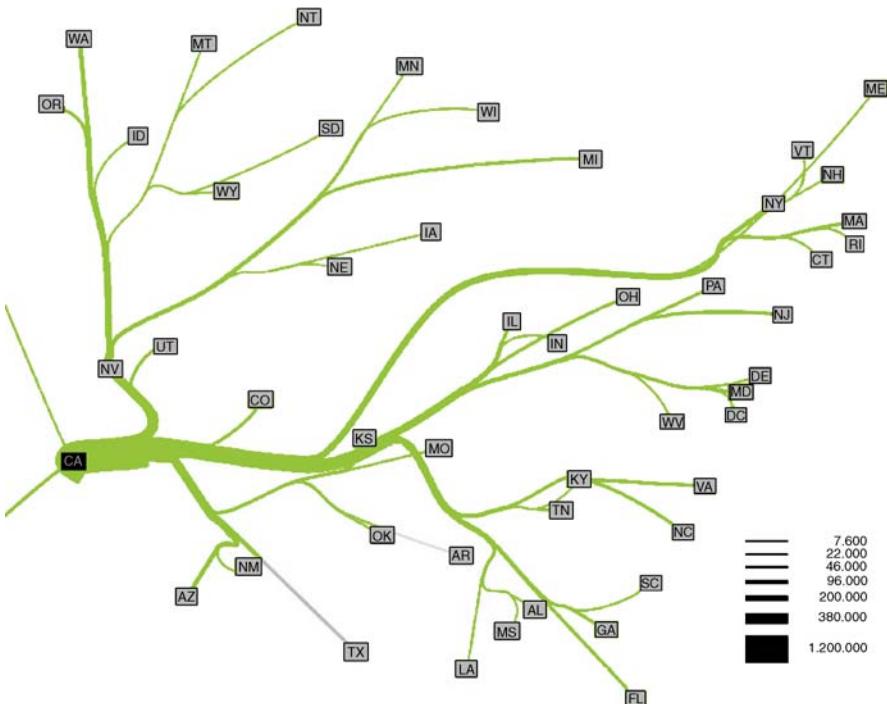


Figure 5.8. Flow map

source and the locations of the sinks. There is one edge in the tree for each sink. Figure 5.8 shows an example using Colorado migration data from 1995 to 2000. Notice that edges are merged as much as possible without compromising the smoothness and distinctness of the terminal flows.

Spanning Trees

5.3.2

It makes sense that we might be able to lay out a spanning tree nicely if we approximate graph-theoretic distance with Euclidean distance. This should tend to place adjacent vertices (parents and children) close together and push vertices separated by many edges far apart. The most popular algorithm for doing this is a variant of multidimensional scaling called the *springs algorithm*. It uses a physical analogy (springs under tension represent edges) to derive a loss function representing total energy in the system (similar to MDS stress). Iterations employ steepest descent to reduce that energy.

Laying out a Simple Tree

Figure 5.9 (Wilkinson 2005) shows an example using data from a small website. Each node is a page and the branches represent the links between pages; their thickness represents traffic between pages (this website has no cross-links). It happens that the root is located near the center of the display. This is a consequence of the force-directed algorithm. Adjacent nodes are attracted and nonadjacent nodes are repelled.

The springs algorithm brings to mind a simple model of a plant growing on a surface. This model assumes branches should have a short length so as to maximize

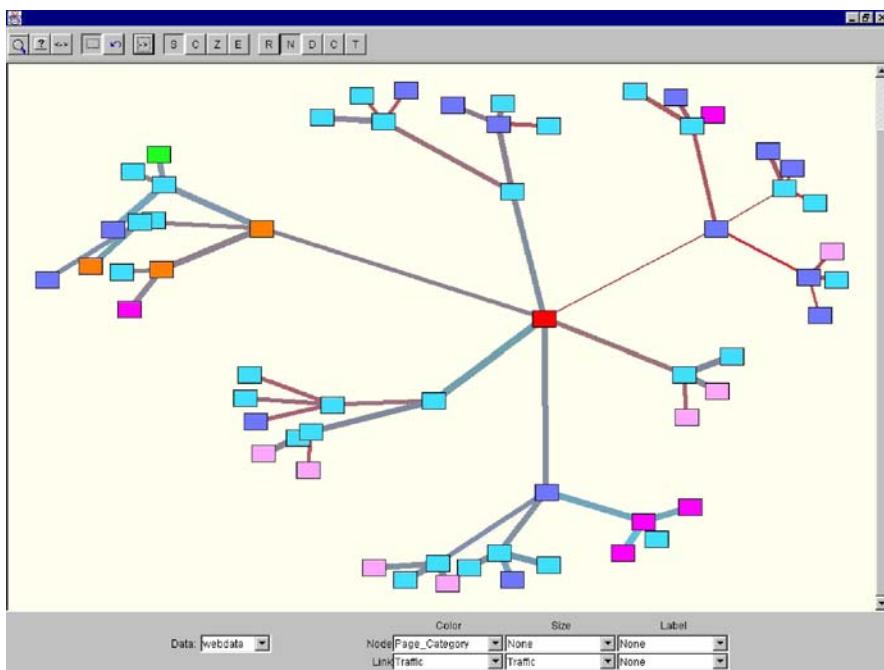


Figure 5.9. Force-directed layout of a website tree



Figure 5.10. A rooted tree, otherwise known as Knotweed (*Polygonum arenastrum*). Photo courtesy of Bill Hosken

water distribution to the leaves and assumes leaves should be separated as much as possible so as to maximize exposure to sunlight. Figure 5.10 shows an example. Given a planar area for uninhibited growth and uniform sunshine, this weed has assumed a shape similar to the web tree in Fig. 5.9.

Laying out Large Trees

Laying out large spanning trees presents special problems. Even in polar form, large trees can saturate the display area. Furthermore, the springs algorithm is computationally expensive on large trees.

One alternative was developed by Graham Wills (Wills 1999), motivated by the hexagon binning algorithm of Carr (Carr et al. 1987). Wills uses the hexagon layout to make edges compact and improve computation through binning. Figure 5.11 shows an example based on website resources (Wills 1999).

Additive Trees

Additive trees require rather complex computations. We are given a (presumably additive) distance matrix on n objects and are required to produce a spanning tree in which the graph-theoretic distances between nodes correspond as closely as possible to the original distances. Figure 5.12 shows an example from White et al. (1998). The gray rectangles highlight three clusters in the data. The article notes that the angles between edges are not significant. The edges are laid out simply to facilitate tracing paths.

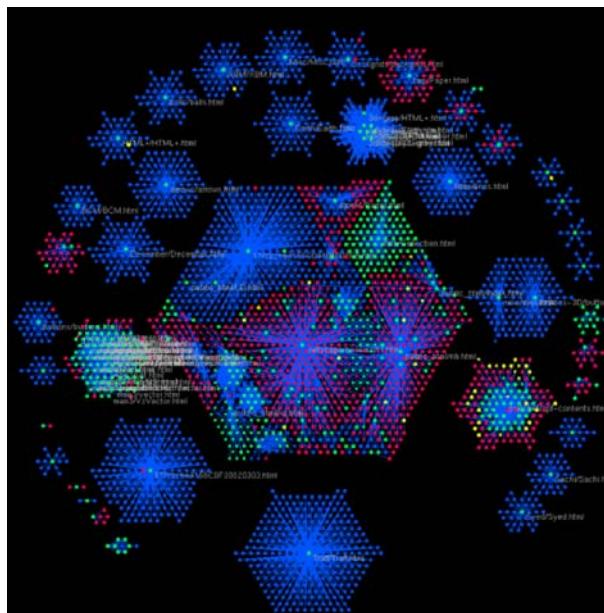


Figure 5.11. Wills hexagonal tree

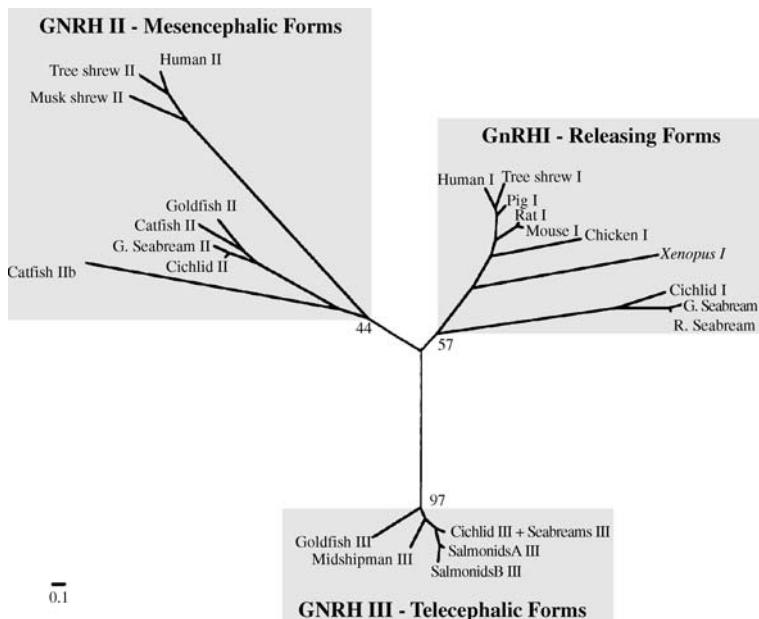


Figure 5.12. Additive tree

5.3.3

Networks

Networks are, in general, cyclic graphs. Force-directed layout methods often work well on networks. There is nothing in the springs algorithm that requires a graph to be a tree. As an example, Fig. 5.13 shows an associative network of animal names from an experiment in Wilkinson (2005). Subjects were asked to produce a list of animal names. Names found to be adjacent in subjects' lists were considered adjacent in a graph.

5.3.4

Directed Graphs

Directed graphs are usually arranged in a vertical (horizontal) partial ordering with source node(s) at top (left) and sink node(s) at bottom (right). Nicely laying out a directed graph requires a topological sort. We temporarily invert cyclical edges to convert the graph to a directed acyclic graph (DAG) so that the paths-to-sink can be identified. Then we do a topological sort to produce a linear ordering of the DAG such that for each edge (u, v) , vertex u is above vertex v . After sorting, we iteratively arrange vertices with tied sort order so as to minimize the number of edge crossings.

Minimizing edge crossings between layers is NP-hard. We cannot always be sure to solve the problem in polynomial time. It amounts to maximizing Kendall's τ correlation between adjacent layers. Heuristic approaches include using direct search, simulated annealing, or constrained optimization.

Figure 5.14 shows a graph encapsulating the evolution of the UNIX operating system. It was computed by the AT&T system of graph layout programs.

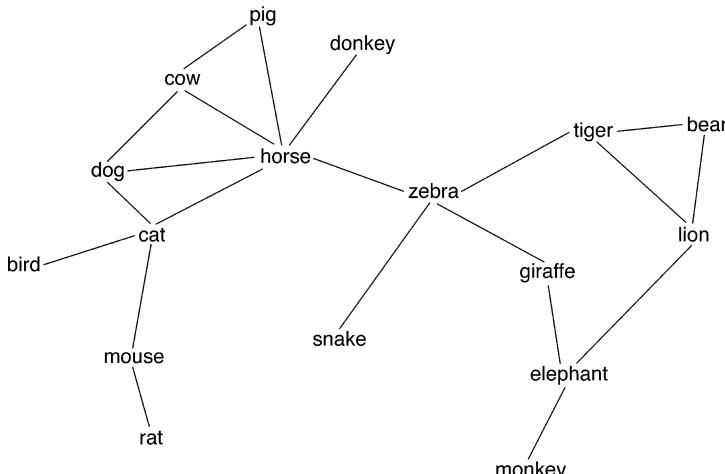


Figure 5.13. Cyclic graph of animal names

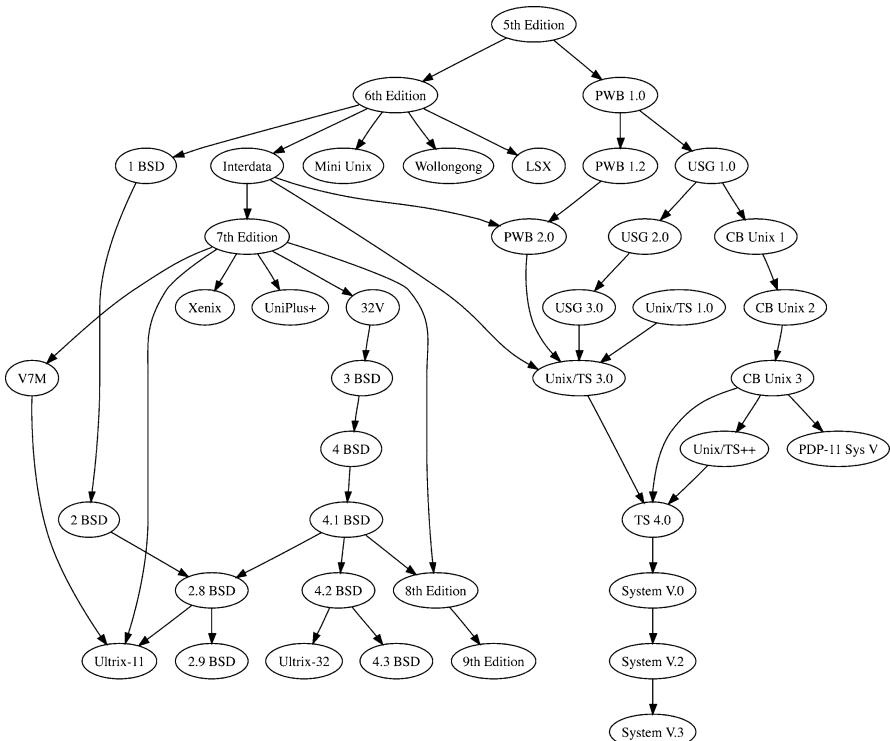


Figure 5.14. Evolution of UNIX operating system; directed graph layout produced by Graphviz (Pixelglow software), courtesy Ian Darwin, Geoff Collyer, Stephen North and Glen Low

Treemaps

5.3.5

Treemaps are recursive partitions of a space. The simplest form is a nested rectangular partitioning of the plane (Johnson and Shneiderman 1991). To transform a binary tree into a rectangular treemap; for example, we start at the root of the tree. We partition a rectangle vertically; each block (tile) represents one of the two children of the root. We then partition each of the two blocks horizontally so that the resulting nested blocks represent the children of the children. We apply this algorithm recursively until all the tree nodes are covered. The recursive splits alternate between vertical and horizontal. Other splitting algorithms are outlined in Bederson et al. 2002).

If we wish, we may color the rectangles using a list of additive node weights. Otherwise, we may use the popular device of resizing the rectangles according to the node weights. Figure 5.15 shows an example that combines color (to represent politics, sports, technology, etc.) and size (to represent number of news sources) in a visualization of the Google news site. This map was constructed by Marcos Weskamp and Dan Albritton.

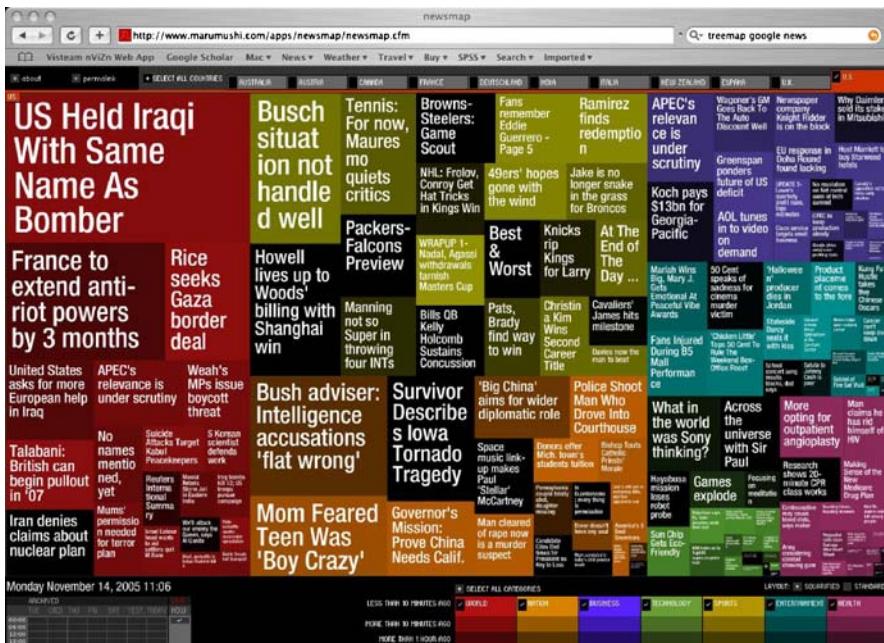


Figure 5.15. Treemap of Google news headlines

5.4 Geometric Graphs

Geometric graphs form the basis for many data mining and analytic graphics methods. The reason for this is the descriptive richness of geometric graphs for characterizing sets of points in a space. We will use some of these graphs in the next section, for example, to develop visual analytics.

Given a set of points in a metric space, a geometric graph is defined by one or more axioms. We can get a sense of the expressiveness of this definition by viewing examples of these graphs on the same set of points in this section; we use data from the famous Box–Jenkins airline dataset (Box and Jenkins 1976), as shown in Fig. 5.16.

We restrict the geometric graphs in this section to:

- 1 *Undirected* (edges consist of unordered pairs)
- 2 *Simple* (no edge pairs a vertex with itself)
- 3 *Planar* (there is an embedding in \mathbb{R}^2 with no crossed edges)
- 4 *Straight* (embedded edges are straight-line segments)

There have been many geometric graphs proposed for representing the “shape” of a set of points X on a plane. Most of these are proximity graphs. A *proximity graph* (or *neighborhood graph*) is a geometric graph whose edges are determined by an indicator function based on distances between a given set of points in a metric space. To define this indicator function, we use an open disk D . We say D *touches* a point if that point is on the boundary of D . We say D *contains* a point if that point is in D . We call the

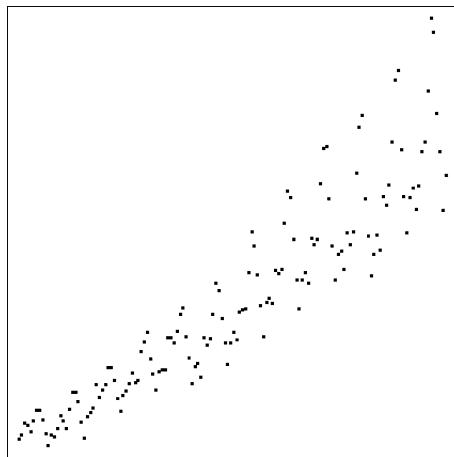


Figure 5.16. Airline dataset

smallest open disk touching two points D_2 ; the radius of this disk is half the distance between the two points and the center of this disk is halfway between the two points. We call an open disk of fixed radius $D(r)$. We call an open disk of fixed radius and centered on a point $D(p, r)$.

Disk Exclusion

5.4.1

Several proximity graphs are defined by empty disks. That is, edges exist in these graphs when disks touching pairs of points are found to be empty.

Delaunay Triangulation

- In a *Delaunay graph*, an edge exists between any pair of points that can be touched by an open disk D containing no points.

The Delaunay triangulation and its dual, the Voronoi tessellation, are powerful structures for characterizing distributions of points. While they have higher-dimensional generalizations, their most frequent applications are in two dimensions. There are several proximity graphs that are subsets of the Delaunay triangulation:

Convex Hull

A *polygon* is a closed plane figure with n vertices and $n - 1$ faces. The *boundary* of a polygon can be represented by a geometric graph whose vertices are the polygon vertices and whose edges are the polygon faces. A *hull* of a set of points X in Euclidean space R^2 is a collection of one or more polygons that have a subset of the points in X for their vertices and that collectively contain all the points in X . This definition includes entities that range from a single polygon to a collection of polygons each consisting of a single point. A polygon is *convex* if it contains all the straight-line segments connecting any pair of its points.

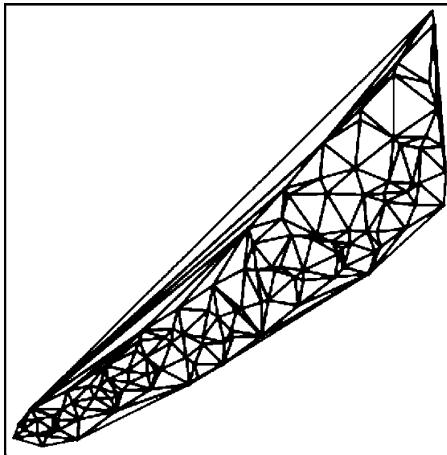


Figure 5.17. Delaunay triangulation

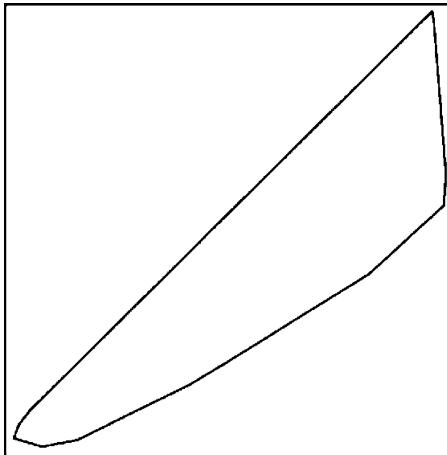


Figure 5.18. Convex hull

- The *convex hull* of a set of points X is the intersection of all convex sets containing X .

There are several algorithms for computing the convex hull. Since the convex hull consists of the outer edges of the Delaunay triangulation, we can use an algorithm for the Voronoi/Delaunay problem and then pick the outer edges. Its computation thus can be $O(n \log n)$.

Nonconvex Hull

A *nonconvex hull* is a hull that is not a convex hull. This class includes simple shapes like a *star convex* or *monotone convex* hull, but it also includes some space-filling, snaky objects and some that have disjoint parts. In short, we are interested in a gen-

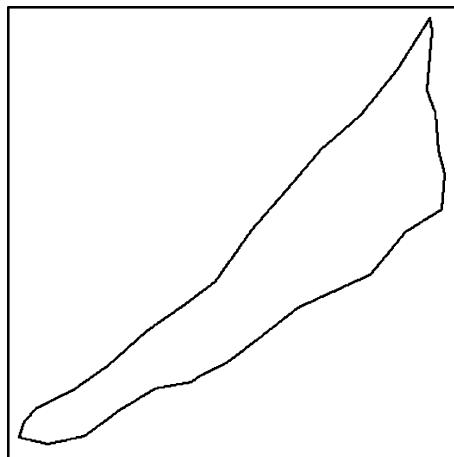


Figure 5.19. Alpha shape

eral class of nonconvex shapes. Some of these shapes are *complexes* (collections of *simplexes*). We take the hull of these shapes to be the collection of exterior edges of these complexes.

- In an *alpha-shape graph*, an edge exists between any pair of points that can be touched by an open disk $D(\alpha)$ containing no points.

Complexes

There are several subsets of the Delaunay triangulation that are complexes useful for characterizing the density of points, shape, and other aspects.

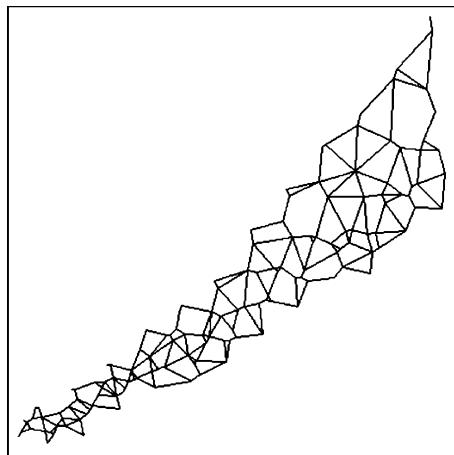


Figure 5.20. Gabriel graph

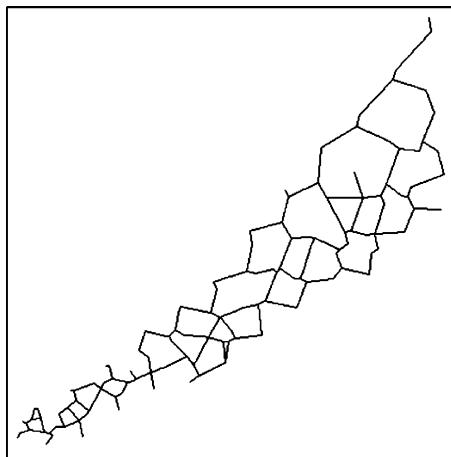


Figure 5.21. Relative neighborhood graph

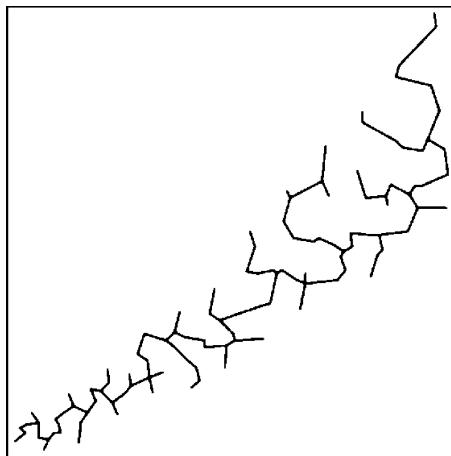


Figure 5.22. Minimum spanning tree

- In a *Gabriel graph*, an edge exists between any pair of points that have a D_2 containing no points.
- In a *relative neighborhood graph*, an edge exists between any pair of points p and q for which r is the distance between p and q and the intersection of $D(p, r)$ and $D(q, r)$ contains no points. This intersection region is called a *lune*.
- A *beta skeleton graph* is a compromise between the Gabriel and relative neighborhood graphs. It uses a lune whose size is determined by a parameter β . If $\beta = 1$, the beta skeleton graph is a Gabriel graph. If $\beta = 2$, the beta skeleton graph is a relative neighborhood graph.
- A *minimum spanning tree* is an acyclical subset of a Gabriel graph.

Disk Inclusion

5.4.2

Several proximity graphs are defined by disk inclusion. That is, edges exist in these graphs when predefined disks contain pairs of points. These graphs are not generally subsets of the Delaunay triangulation.

- In a *k-nearest-neighbor graph* (KNN), a directed edge exists between a point p and a point q if $d(p, q)$ is among the k smallest distances in the set $\{d(p, j) \mid 1 \leq j \leq n, j \neq p\}$. Most applications restrict KNN to a simple graph by removing self loops and edge weights. If $k = 1$, this graph is a subset of the MST. If $k > 1$, this graph may not be planar. Figure 5.23 shows a nearest-neighbor graph for

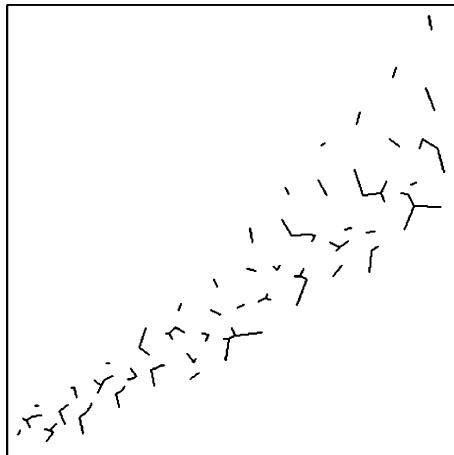


Figure 5.23. Nearest-neighbor graph

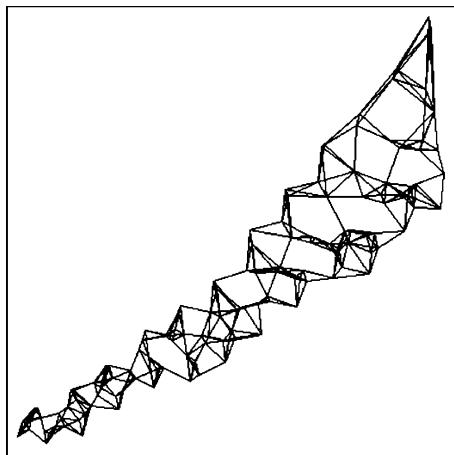


Figure 5.24. 5-Nearest-neighbor graph

the airline data. Figure 5.24 shows a 5-nearest-neighbor graph on the same set of points.

- In a *distance graph*, an edge exists between any pair of points that both lie in a $D(r)$. The radius r defines the size of the neighborhood. This graph is not always planar and is therefore not a subset of the Delaunay.
- In a *sphere-of-influence graph*, an edge exists between a point p and a point q if $d(p, q) \leq d_{nn}(p) + d_{nn}(q)$, where $d_{nn}(\cdot)$ is the nearest-neighbor distance for a point.

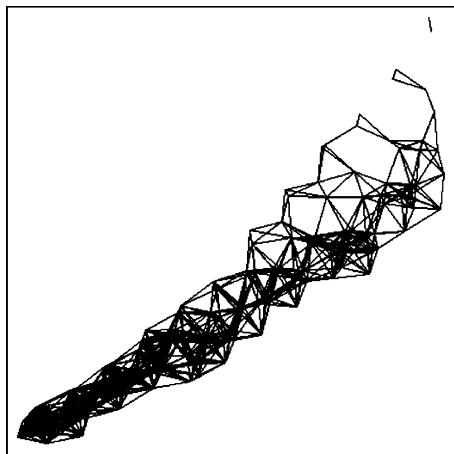


Figure 5.25. Distance graph

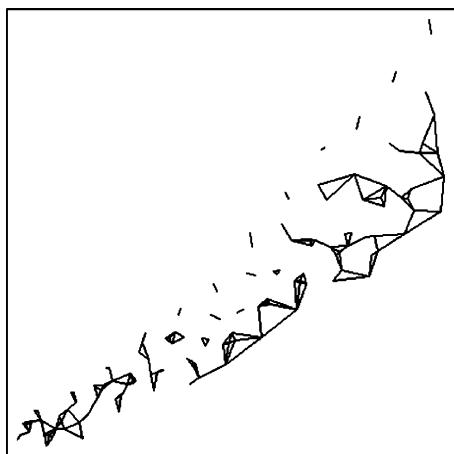


Figure 5.26. Sphere-of-influence graph

Graph-theoretic Analytics

Some graph-analytic procedures naturally lend themselves to visualization or are based on geometric graphs. We discuss a few in this section.

Scagnostics

5.5.1

A scatterplot matrix, variously called a SPLOM or casement plot or draftman's plot, is a (usually) symmetric matrix of pairwise scatterplots. An easy way to conceptualize a symmetric SPLOM is to think of a covariance matrix of p variables and imagine that each off-diagonal cell consists of a scatterplot of n cases rather than a scalar number representing a single covariance. This display was first published by John Hartigan (1975) and was popularized by Tukey and his associates at Bell Laboratories.

Large scatterplot matrices become unwieldy when there are many variables. First of all, the visual resolution of the display is limited when there are many cells. This defect can be ameliorated by pan and zoom controls. More critical, however, is the multiplicity problem in visual exploration. Looking for patterns in $p(p - 1)/2$ scatterplots is impractical for more than 25 variables. This problem is what prompted the Tukeys' solution.

The Tukeys reduced an $O(p^2)$ visual task to an $O(k^2)$ visual task, where k is a small number of measures of the distribution of a 2-D scatter of points. These measures included the area of the peeled convex hull of the 2-D point scatters, the perimeter length of this hull, the area of closed 2-D kernel density isolevel contours, the perimeter length of these contours, the convexity of these contours, a modality measure of the 2-D kernel densities, a nonlinearity measure based on principal curves fitted to the 2-D scatterplots, the median nearest-neighbor distance between points, and several others. By using these measures, the Tukeys aimed to detect anomalies in density, distributional shape, trend, and other features in 2-D point scatters.

After calculating these measures, the Tukeys constructed a scatterplot matrix of the measures themselves, in which each point in the scagnostic SPLOM represented a scatterplot cell in the original data SPLOM. With brushing and linking tools, unusual scatterplots could be identified from outliers in the scagnostic SPLOM.

Wilkinson et al. (2005) extended this procedure using proximity graphs. This extension improved scalability, because the graph calculations are $O(n \log n)$, and allowed the method to be applied to categorical and continuous variables. Wilkinson et al. (2005) developed nine scagnostics measures: Outlying, Skewed, Clumpy, Convex, Skinny, Striated, Stringy, Straight and Monotonic.

Figure 5.27 shows the output of the program developed in Wilkinson et al. (2005). The dataset used in the example is the Boston housing data cited in Breiman et al. (1984). The left SPLOM shows the data. The larger scagnostics SPLOM in the middle of the figure shows the distribution of the nine scagnostics. One point is highlighted. This point is an especially large value on the Outlying scagnostic statistic. Its corresponding scatterplot is shown in the upper-right plot superimposed on the scagnostics SPLOM. This plot involves a dummy variable for whether a tract bounds

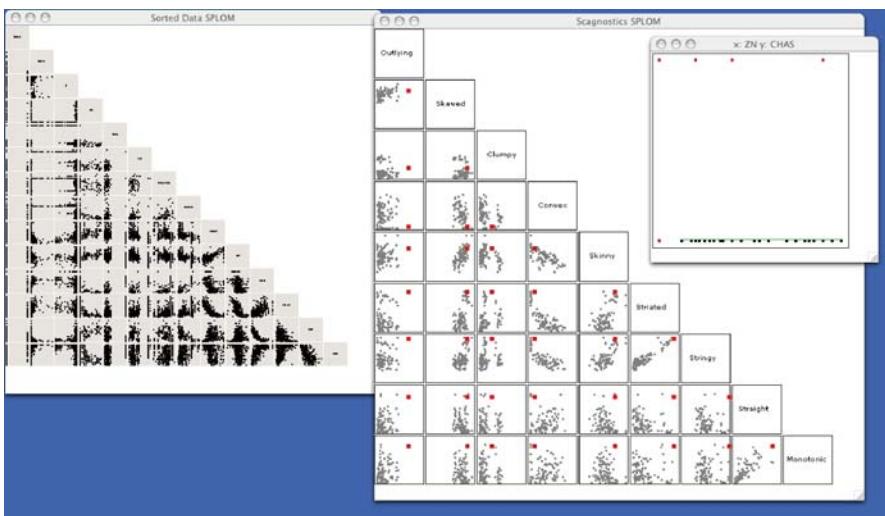


Figure 5.27. Scagnostics

the Charles River (CHAS) and proportion of residential land zoned for lots over 25 000 ft². (ZN). The scagnostic Outlying measure flagged the few cases that bounded the Charles River. The locations of this scatterplot point in the other scagnostics SPLOM characterize the plot as relatively skewed, skinny, striated, and stringy, but not convex.

5.5.2 Sequence Analysis

A sequence is a list of objects, e.g. $\langle x, y, z \rangle$. The ordering of the list is given by an order relation. In many applications of sequence analysis, objects are represented by tokens and sequences are represented by strings of tokens. In biosequencing, for example, the letters A, C, T and G are used to represent the four bases in a DNA strand.

Suppose we are given a length n string of tokens and want to find the most frequently occurring substrings of length m in the string ($m \ll n$). A simple (not especially fast) algorithm to do this involves generating candidate substrings and testing them against the target string. We begin with strings of length 1, each comprised of a different token. Then we build candidate subsequences of length 2. We count the frequency of each of these subsequences in the target string. Using any of these length 2 subsequences with a count greater than zero, we build candidate subsequences of length 3. We continue the generate-and-test process until we have tested the candidates of length m or until all counts are zero. This stepwise procedure traverses a subset of the branches of the tree of all possible subsequences so we do not have as many tests to perform.

Embedding a sequence analysis in a graph layout often gives us a simple way to visualize these subsequences. The layout may be based on known coordinates (as in geographic problems) or on an empirical layout using adjacency in the sequence list

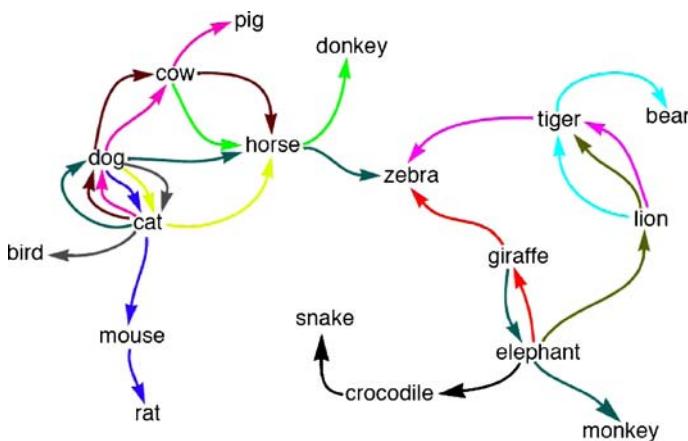


Figure 5.28. Animal name sequences

as edge information. Figure 5.28 shows an example using the same data represented in Fig. 5.13. We have superimposed the sequences using arrows.

Comparing Sequences

Suppose we have two sequences of characters or objects and we wish to compare them. If the sequences are of length n , we can construct an n by n table of zeroes and place a 1 in a diagonal cell if the value in each sequence at the corresponding position is the same. We would have an identity matrix if both sequences were identical and we can plot this matrix as a square array of pixels. With real data, however, we are more likely to encounter matching runs of subsequences that occur in different locations in each sequence. Consequently, we more often see subsequences as runs off the diagonal.

Figure 5.29 (Altschul et al. 2001) shows an example of this type of plot. Subsequences appear in the plot as diagonal runs from upper left to lower right. The longer the diagonal bars, the longer the matching subsequences.

Critical Paths

Suppose we have a directed acyclic graph (DAG) where the vertices represent tasks and an edge (u, v) implies that task u must be completed before task v . How do we schedule tasks to minimize overall time to completion? This job-scheduling problem has many variants. One customary variant is to weight the edges by the time it takes to complete tasks. We will mention two aspects of the problem that involve graphing. First, how do we lay out a graph of the project? We use the layout for a directed graph and flip the graph to a horizontal orientation. The result of our efforts is called a CPM (critical path method) graph. Second, how do we identify and color the critical path? Identifying the critical path is easy if the edges are not weighted. We simply do a breadth-first search of the DAG and keep a running tally of the path length. Finding the shortest path through a weighted graph requires dynamic programming.

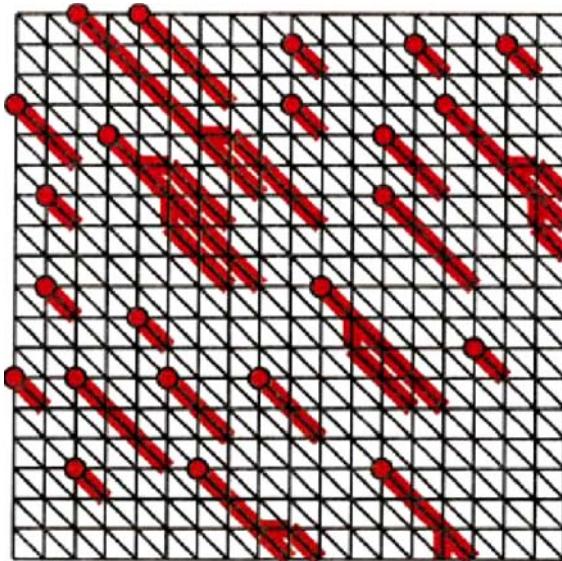


Figure 5.29. Comparing two sequences (courtesy Steven Altschul)

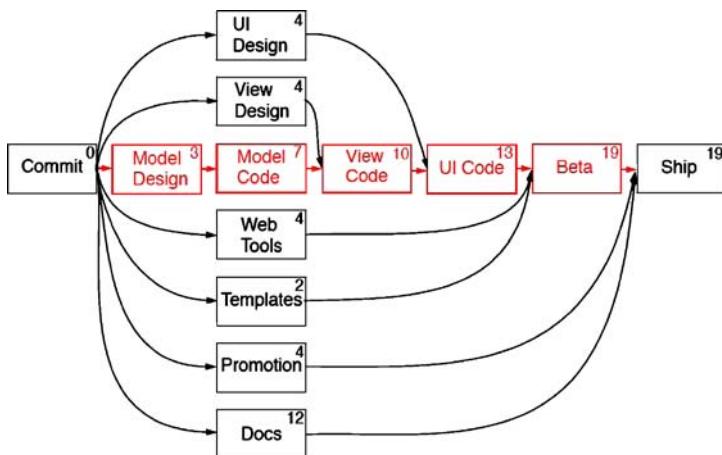


Figure 5.30. CPM chart

Graph layouts of large projects can become messy. Even without edge crossings, a large CPM graph can be difficult to interpret. An alternative to this display is called a Gantt chart. The horizontal axis measures time. The length of a bar represents the duration of a task. The vertical axis separates the tasks. The coloring categorizes tasks. The original form of the chart did not have the benefit of the graph theory behind CPM, but modern incarnations have blended the bars of the Gantt chart with the information on the critical path. Most computer project management packages com-

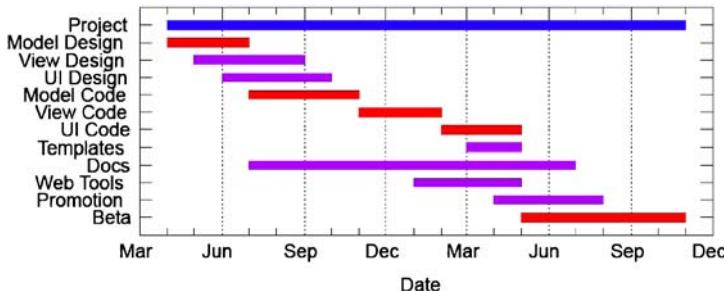


Figure 5.31. Gantt chart

pute the critical path with graph-theoretic algorithms and display the results in some variety of the Gantt chart.

Graph Matching

5.5.3

Given two graphs, how do we determine if there is an isomorphism between them? And if they are not isomorphic, can we identify isomorphic subgraphs or compute an overall measure of concordance? These questions have many answers; we will cover only a few.

Matching graphs has many applications in biology, chemistry, image processing, computer vision, and search engines. To the extent that a body of knowledge can be represented as a graph (a set of vertices and relations among them), graph matching is a core application. It provides, for example, the foundation for searching a database of graphs for a particular graph. If images and other material can be represented as graphs, then graph matching provides a powerful indexing mechanism for large databases of disparate materials. Indeed, since a relational table can be represented as a graph, matching can be used to identify congruent tables of primitives in a database. Given the topic of this chapter, however, we will focus on matching 2D geometric graphs.

Exact Graph Matching

Exact graph matching consists of identifying the isomorphism between two graphs. This amounts to finding (1) a vertex in one graph for every vertex in the other (and vice versa) and (2) an edge in one graph for every edge in the other (and vice versa). If both graphs are connected, then the second condition suffices to establish the isomorphism. Because this is a standard sorting-and-searching problem, it has polynomial complexity.

The problem is more general, however, because we are usually interested in finding isomorphisms under a permutation transformation. That is, we seek a vertex relabeling of G_1 such that an isomorphism between G_1 and G_2 exists after the relabeling. This more general matching problem has unknown complexity. For planar graphs, however, Hopcroft and Wong (1974) prove linear time complexity. Skiena (1998) and Shasha et al. (2002) discuss this topic further and review software for graph matching.

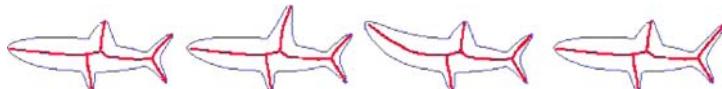


Figure 5.32. Medial axes (courtesy Thomas Sebastian, Philip Klein and Benjamin Kimia)

Approximate Graph Matching

Approximate graph matching consists of maximizing an index of concordance between two graphs under relabeling. Many indices have been proposed. Early approaches computed simple graph-theoretic measures and used these to compute distance or correlation coefficients. The *cophenetic* correlation, for example, is a Pearson correlation between the entries of a distance matrix and the corresponding ultrametric distances derived from a hierarchical clustering tree. This approach implies a matching index based on correlating ultrametric distances from two different trees (possibly after relabeling).

More recent approaches use other measures to derive concordance measures. The most famous example is the Google search engine (Brin and Page 1998), which uses a graph spectral measure to assess similarity. In the field of shape recognition, proximity graphs have been constructed from polygons by using disks similar to those we discussed in the previous section. Klein et al. (2001), for example, developed a shape-matching procedure using a derivative of the *medial axis*. The medial axis of a polygon is the locus of the centers of maximal circles that touch the polygon boundary more than once. Figure 5.32 shows an example.

Klein et al. (2001) used an *edit distance* measure to evaluate matching of medial axis graphs. Edit distance is the number of elementary operations needed to transform one graph into another. In the simple case, there are two editing operators: delete an edge, and relabel an edge. By subjecting the topology of the medial axis representations of shape to a specific edit distance measure, Klein et al. (2001) were able to characterize 2D projections of 3D shapes with a high degree of accuracy, regardless of orientation or scale. Torsello (2004) extended these methods.

Any proximity graph can be applied to the shape-recognition problem using edit distance or measures of similarity. Gandhi (2002), for example, measured the shape of leaves by recording turning angles at small steps along their perimeters. This measure transformed a shape into a single time series. Gandhi (2002) then used *dynamic time warping* (Sakoe and Chiba 1978) to compute a distance measure between leaf shapes.

Conclusion

This chapter has covered only a fraction of the visualization applications of graph theory. Graph-theoretic visualization is a rapidly developing field because only in the last few decades have the connections between data representation and graph theory been made explicit. Tukey and Tukey (1985) anticipated the role graph theory would play in visualization and John Tukey was especially interested in convex hulls, mini-

mum spanning trees, and other graphs for characterizing high-dimensional data. But as Tukey said many times, more powerful computing environments would be needed to realize the power of these methods. That time has arrived.

References

- Altschul, S., Bunday, R., Olsen, R. and Hwa, T. (2001). The estimation of statistical parameters for local alignment score distributions., *Nucleic Acids Research* 29:251–261.
- Battista, G.D., Eades, P., Tamassia, R. and Tollis, I. (1994). Algorithms for drawing graphs: an annotated bibliography, *Computational Geometry: Theory and Applications* 4:235–282.
- Battista, G.D., Eades, P., Tamassia, T. and Tollis, I. (1999). *Graph Drawing: Algorithms for the Visualization of Graphs*, Prentice-Hall, Upper Saddle River, NJ.
- Bederson, B., Schneiderman, B. and Wattenberg, M. (2002). Ordered and quantum treemaps: Making effective use of 2d space to display hierarchies, *ACM Transactions on Graphics (TOG)* 21(4):833–854.
- Box, G. E.P. and Jenkins, G.M. (1976). *Time Series Analysis: Forecasting and Control (rev. ed.)*, Holden-Day, Oakland, CA.
- Breiman, L., Friedman, J., Olshen, R. and Stone, C. (1984). *Classification and Regression Trees*, Wadsworth, Belmont, CA.
- Brin, S. and Page, L. (1998). The anatomy of a large-scale hypertextual Web search engine, *Computer Networks and ISDN Systems* 30(1–7):107–117.
- Carr, D.B., Littlefield, R.J., Nicholson, W.L. and Littlefield, J.S. (1987). Scatterplot matrix techniques for large n, *Journal of the American Statistical Association* 82:424–436.
- Dirschel, P. (1991). Klassifikationsbaumgrundlagen und -neuerungen, in W. Flischer, M. Nagel and R. Ostermann (eds), *Interaktive Datenanalyse mit ISP*, Essen, pp. 15–30.
- Gandhi, A. (2002). *Content-based image retrieval: plant species identification*, PhD thesis, Oregon State University.
- Hartigan, J.A. (1975). Printer graphics for clustering, *Journal of Statistical Computation and Simulation* 4:187–213.
- Hopcroft, J.E. and Wong, J.K. (1974). Linear time algorithm for isomorphism of planar graphs, *STOC: ACM Symposium on Theory of Computing (STOC)*.
- Johnson, B. and Schneiderman, B. (1991). Treemaps: A space-filling approach to the visualization of hierarchical information structures, *Proceedings of the IEEE Information Visualization '91*, pp. 275–282.
- Klein, P., Sebastian, T. and Kimia, B. (2001). Shape matching using edit-distance: an implementation, *Twelfth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, Washington, D.C., pp. 781–790.
- Kleiner, B. and Hartigan, J. (1981). Representing points in many dimensions by trees and castles, *Journal of the American Statistical Association* 76:260–269.
- Kobsa, A. (2004). User experiments with tree visualization systems, *Proceedings of the IEEE Information Visualization 2004*, pp. 9–16.

- Kruja, E., Marks, J., Blair, A. and Waters, R. (2001). A short note on the history of graph drawing, *Graph Drawing : 9th International Symposium, GD 2001, Lecture Notes in Computer Science*, Vol. 2265, Heidelberg, pp. 272–278.
- Lamping, J., Rao, R. and Pirolli, P. (1995). A focus+context technique based on hyperbolic geometry for visualizing large hierarchies, *Human Factors in Computing Systems: CHI 95 Conference Proceedings*, pp. 401–408.
- Lausen, B., Sauerbrei, W. and Schumacher, M. (1994). Classification and regression trees (CART) used for the exploration of prognostic factors measured on different scales, in P. Dirschedl and R. Ostermann (eds), *Computational Statistics*, Heidelberg, pp. 483–496.
- Marchette, D. (2004). *Random Graphs for Statistical Pattern Recognition*, John Wiley & Sons, New York.
- Minnotte, M. and Scott, D. (1993). The mode tree: A tool for visualization of nonparametric density features, *Journal of Computational and Graphical Statistics* 2:51–68.
- Phan, D., Yeh, R., Hanrahan, P. and Winograd, T. (2005). Flow map layout, *Proceedings of the IEEE Information Visualization 2005*, pp. 219–224.
- Sakoe, H. and Chiba, S. (1978). Dynamic programming algorithm optimization for spoken word recognition, *IEEE Transactions on Acoustics, Speech, and Signal Processing* 26:43–49.
- Shasha, D., Wang, J.T.-L. and Giugno, R. (2002). Algorithmics and applications of tree and graph searching, *Symposium on Principles of Database Systems*, pp. 39–52.
- Skiena, S. (1998). *The Algorithm Design Manual*, Springer-Verlag, New York.
- Torsello, A. (2004). *Matching Hierarchical Structures for Shape Recognition*, PhD thesis, University of York. citeseer.ist.psu.edu/torsello04matching.html.
- Tukey, J.W. and Tukey, P. (1985). Computer graphics and exploratory data analysis: An introduction, National Computer Graphics Association, Fairfax, VA, United States.
- Urbanek, S. (2003). Many faces of a tree, *Computing Science and Statistics: Proceedings of the 35th Symposium on the Interface*.
- Vach, W. (1995). Classification trees, *Computational Statistics* 10:9–14.
- White, R., Eisen, J., TL, T.K. and Fernald, R. (1998). Second gene for gonadotropin-releasing hormone in humans, *Proceedings of the National Academy of Sciences*, Vol. 95, pp. 305–309.
- Wilkinson, L. (1999). *The Grammar of Graphics*, Springer-Verlag, New York.
- Wilkinson, L. (2005). *The Grammar of Graphics* (2nd edn.), Springer-Verlag, New York.
- Wilkinson, L., Anand, A. and Grossman, R. (2005). Graph-theoretic scagnostics, *Proceedings of the IEEE Information Visualization 2005*, pp. 157–164.
- Wills, G.J. (1999). NicheWorks – interactive visualization of very large graphs, *Journal of Computational and Graphical Statistics* 8(2):190–212.

High-dimensional Data Visualization

II.6

Martin Theus

6.1	<i>Introduction</i>	152
6.2	<i>Mosaic Plots</i>	153
	Associations in High-dimensional Data	153
	Response Models.....	155
	Models.....	156
6.3	<i>Trellis Displays</i>	156
	Definition	157
	Trellis Display vs. Mosaic Plots	158
	Trellis Displays and Interactivity.....	161
	Visualization of Models	162
6.4	<i>Parallel Coordinate Plots</i>	164
	Geometrical Aspects vs. Data Analysis Aspects	164
	Limits	166
	Sorting and Scaling Issues	169
	Wrap-up.....	171
6.5	<i>Projection Pursuit and the Grand Tour</i>	172
	Grand Tour vs. Parallel Coordinate Plots	174
6.6	<i>Recommendations</i>	175

One of the biggest challenges in data visualization is to find general representations of data that can display the multivariate structure of more than two variables. Several graphic types like mosaicplots, parallel coordinate plots, trellis displays, and the grand tour have been developed over the course of the last three decades. Each of these plots is introduced in a specific chapter of this handbook.

This chapter will concentrate on investigating the strengths and weaknesses of these plots and techniques and contrast them in the light of data analysis problems.

One very important issue is the aspect of interactivity. Except for trellis displays, all the above plots need interactive features to rise to their full power. Some, like the grand tour, are only defined by using dynamic graphics.

6.1

Introduction

It is sometimes hard to resist the problem that is captured in the phrase “if all you have is a hammer, every problem looks like a nail.” This obviously also holds true for the use of graphics. A grand tour expert will most likely include a categorical variable in the high-dimensional scatterplot, whereas an expert on mosaicplots probably will try to fit a data problem as far as possible into a categorical framework.

This chapter will focus on the appropriate use of the different plots for high-dimensional data analysis problems and contrast them by emphasizing their strengths and weaknesses.

Data visualization can roughly be categorized into two applications:

1. Exploration

In the exploration phase, the data analyst will use many graphics that are mostly unsuitable for presentation purposes yet may reveal very interesting and important features. The amount of interaction needed during exploration is very high. Plots must be created fast and modifications like sorting or rescaling should happen instantaneously so as not to interrupt the line of thought of the analyst.

2. Presentation

Once the key findings in a data set have been explored, these findings must be presented to a broader audience interested in the data set. These graphics often cannot be interactive but must be suitable for printed reproduction. Furthermore, some of the graphics for high-dimensional data are all but trivial to read without prior training, and thus probably not well suited for presentation purposes – especially if the audience is not well trained in statistics.

Obviously, the amount of interactivity used is the major dimension to discriminate between exploratory graphics and presentation graphics.

Interactive linked highlighting, as described by Wills (2008, Chapter II.9 same volume), is one of the keys to the right use of graphics for high-dimensional data. Linking across different graphs can increase the dimensionality beyond the number of dimensions captured in a single multivariate graphic. Thus, the analyst can choose

the most appropriate graphics for certain variables of the data set; linking will preserve the multivariate context.

Although much care has been taken to ensure the best reproduction quality of all graphics in this chapter, the reader may note that the printed reproduction in black and white lacks clarity for some figures. Please refer to the book's website for an electronic version that offers full quality.

Mosaic Plots

6.2

Mosaic plots (Hartigan and Kleiner, 1981; Friendly, 1994; Hofmann, 2000) are probably the multivariate plots that require the most training for a data analyst. On the other hand, mosaicplots are extremely versatile when all possible interaction and variations are employed, as described by Hofmann (2008, Chapter III.13 same volume).

This section will explore typical uses of mosaicplots in many dimensions and move on to trellis displays.

Associations in High-dimensional Data

6.2.1

Meyer et al. (2008, Chapter III.12 same volume) already introduced techniques for visualizing association structures of categorical variables using mosaicplots. Obviously, the kind of interactions we look at in high-dimensional problems is usually more complex. Although statistical theory for categorical data often assumes that all variables are of equal importance, this may not be the case with real problems. Using the right order of the variables, mosaicplots can take the different roles of the variables into account.

Example: Detergent data

For an illustration of mosaicplots and their applications, we chose to look at the 4-D problem of the detergent data set (cf. Cox and Snell, 1991). In this data set we look at the following four variables:

1. **Water softness**
(soft, medium, hard)
2. **Temperature**
(low, high)
3. **M-user** (person used brand M before study)
(yes, no)
4. **Preference** (brand person prefers after test)
(X, M)

The major interest of the study is to find out whether or not preference for a detergent is influenced by the brand someone uses. Looking at the interaction of *M-user* and *Preference* will tell us that there is an interaction, but unrelated to the other two

variables. Looking at the variables *Water Softness* and *Temperature* we will find something that is to be expected: harder water needs warmer temperatures for the same washing result and a fixed amount of detergent.

Mosaic plots allow the inspection of the interaction of *M-user* and *Preference* conditioned for each combination of *Water Softness* and *Temperature*, resulting in a plot that includes the variables in the order in which they are listed above. Figure 6.1 shows the increasing interaction of *M-user* and *Preference* for harder water and higher temperatures.

Several recommendations can be given for the construction of high-dimensional classical mosaicplots:

- The first two and the last two variables in a mosaicplot can be investigated most efficiently regarding their association. Thus the interaction of interest should be put into the last two positions of the plot. Variables that condition an effect should be the first in the plot.
- To avoid unnecessary clutter in a mosaicplot of equally important variables, put variables with only a few categories first.
- If combinations of cells are empty (this is quite common for high-dimensional data due to the curse of dimensionality), seek variables that create empty cells at high levels in the plot to reduce the number of cells to be plotted (empty cells at a higher level are not divided any further, thus gathering many potential cells into one).
- If the last variable in the plot is a binary factor, one can reduce the number of cells by linking the last variable via highlighting. This is the usual way to handle categorical response models.

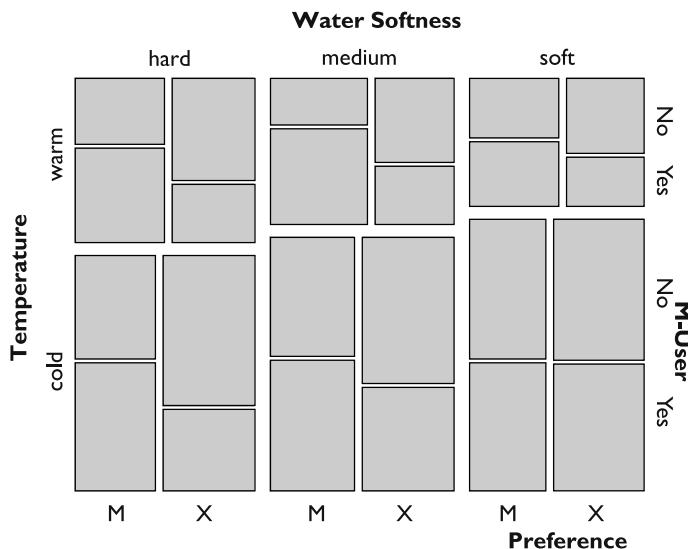


Figure 6.1. The interaction of *M-user* and *Preference* increases for harder water and higher temperatures

- Subsets of variables may reveal features far more clearly than using all variables at once. In interactive mosaicplots one can add/drop or change variables displayed in a plot. This is very efficient when looking for potential interactions between variables.

Response Models

6.2.2

In many data sets there is a single dependent categorical outcome and several categorical influencing factors. The best graphical representation of such a situation is to put all influencing factors in a mosaicplot and link the dependent variable with a barchart. This setup is shown in Fig. 6.2 for the Caesarean data set (cf. Fahrmeir and Tutz, 1994). The data set consists of three influencing factors – *Antibiotics*, *Risk Factor*, and *Planned* and one dependent categorical outcome, *Infection*, for 251 caesarean births. The question of interest is to find out which factors, or combination of factors, have a higher probability of leading to an infection.

At this point it is important to rethink what the highlighted areas in the mosaicplot actually show us. Let us look at the cases where no caesarean was planned, a risk factor was present, and no antibiotics were administered (the lower left cell in Fig. 6.2, which is highlighted to a high degree). In this combination, 23 of the 26 cases got an infection, making almost 88.5 %. That is

$$P(\text{Infection} \mid \overline{\text{Antibiotics}} \wedge \overline{\text{Risk Factor}} \wedge \overline{\text{Planned}}) = 0.8846.$$

But there is more we can learn from the plot. The infection probability is highest for cases with risk factors and no antibiotics administered. There is also one oddity

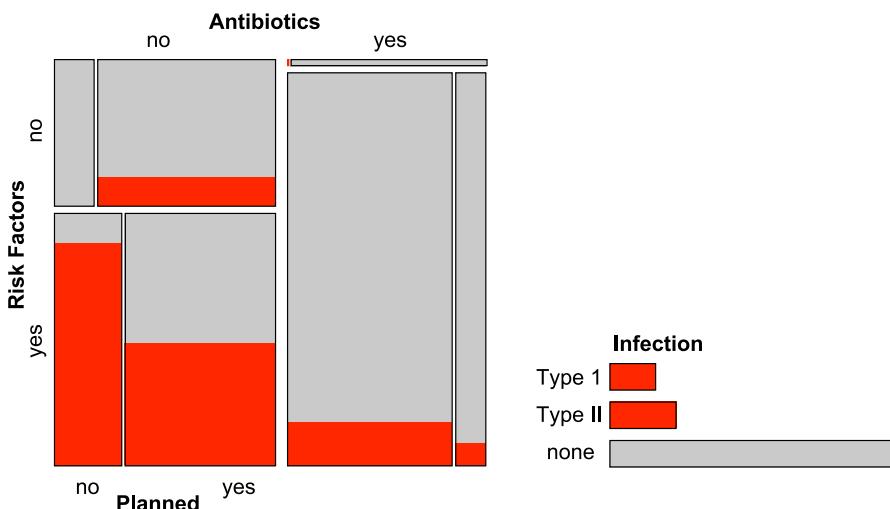


Figure 6.2. The categorical response model for the *caesarean birth* data is visualized using a mosaicplot for the influencing factors and a *barchart* for the response variable. Infection cases have been highlighted

in the data. Whereas the fact of a planned caesarean reduces the infection risk by around half, we do not have a single infection case for unplanned caesareans without risk factors and antibiotics – although at least three would be expected. Note that the chosen order is crucial for seeing this feature most easily. All these results can be investigated by looking at Fig. 6.2 but are harder to find by using classical models – nonetheless, they should be used to check significance.

6.2.3 Models

Meyer et al. (2008, Chapter III.12 same volume) presents a method for displaying association models in mosaicplots. One alternative to looking at log-linear models with mosaicplots is to plot the expected values instead of the observed values. This also permits the plotting of information for empty cells, which are invisible in the raw data but do exist in the modeled data. In general, a mosaicplot can visualize any continuous variable for crossings of categorical data, be it counts, expected values of a model, or any other positive value. Figure 6.3 shows the data from Fig. 6.1 with the two interactions *Water Softness* and *Temperature* and *M-user* and *Preference* included. Remaining residuals are coded in red (negative) and blue (positive). The feature we easily found in Fig. 6.1 – an increasing interaction between *M-user* and *Preference* – would call for a four-way interaction or at least some nonhierarchical model. Neither model can be interpreted easily or communicated to nonstatisticians. Furthermore, the log-linear model including the two two-way interactions has a *p*-value far greater than 0.05, suggesting that the model captures all “significant” structure. A more de-

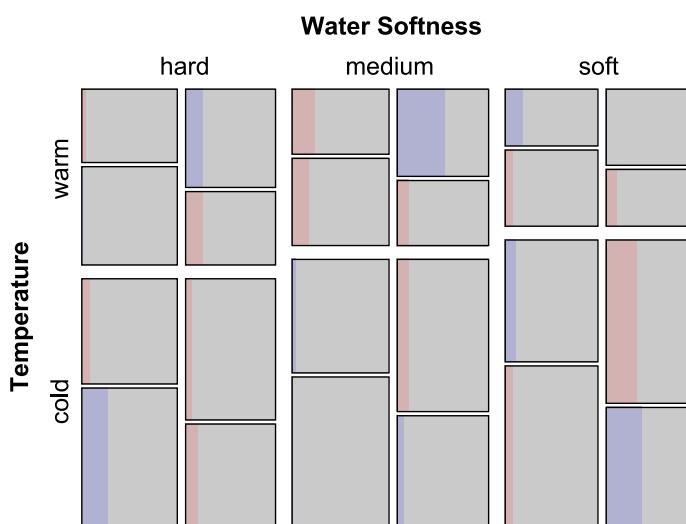


Figure 6.3. A model of the Detergent data with the interactions of *Water Softness* and *Temperature* and *M-user* and *Preference* included. The residuals are highlighted in red (darker shade) and blue (lighter shade). A very faded color indicates a high *p*-value

tailed discussion on log-linear models and mosaicplots can be found in Theus and Lauer (1999).

Trellis Displays

6.3

Trellis displays (called *Lattice Graphics* within the R package) also use conditioning to plot high-dimensional data. But whereas mosaicplots use a recursive layout, trellis displays use a gridlike structure to plot the data conditioned on certain subgroups.

Definition

6.3.1

Trellis displays were introduced by Becker et al. (1996) as a means to visualize multivariate data (see also Theus, 1999). Trellis displays use a latticelike arrangement to place plots onto so-called panels. Each plot in a trellis display is conditioned upon at least one other variable. To make plots comparable across rows and columns, the same scales are used in all the panel plots.

The simplest example of a trellis display is probably a boxplot y by x . Figure 6.4 shows a boxplot of the gas mileage of cars conditioned on the type of car. Results can easily be compared between car types since the scale does not change when visually traversing the different categories. Even a further binary variable can be introduced when highlighting is used, which would be the most effective way to add a third (binary) variable to the plot.

In principle, a single trellis display can hold up to seven variables at a time. Naturally five out of the seven variables need to be categorical, and two can be continuous. At the core of a trellis display we find the *panel plot*. The up to two variables plotted in the panel plot are called *axis variables*. (The current *Lattice Graphics* implementation in R does actually offer higher-dimensional plots like parallel coordinate

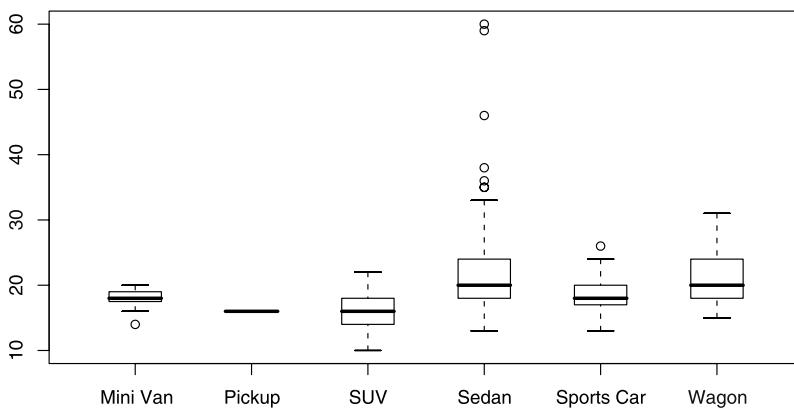


Figure 6.4. A boxplot y by x is a simple form of a trellis display

plots as panel plots, which is only a technical detail and not relevant for data analysis purposes). In principle the panel plot can be any arbitrary statistical graphic, but usually nothing more complex than a scatterplot is chosen. All panel plots share the same scale. Up to three categorical variables can be used as *conditioning variables* to form rows, columns, and pages of the trellis display. To annotate the conditioning categories of each panel plot, the so-called *strip labels* are plotted atop each panel plot, listing the corresponding category names. The two remaining variables – the so-called *adjunct variables* – can be coded using different glyphs and colors (if the panel plot is a glyph-based plot).

Trellis displays introduce the concept of *shingles*. Shingling is the process of dividing a continuous variable into (possibly overlapping) intervals in order to convert this continuous variable into a discrete variable. Shingling is quite different to conditioning with categorical variables. Overlapping shingles/intervals leads to multiple representations of data within a trellis display, which is not the case for categorical variables. Furthermore, it is hard to judge which intervals/cases have been chosen to build a shingle. Trellis displays show the interval of a shingle using an interval of the strip label. This is a solution which does not waste plotting space, but the information on the intervals is hard to read from the strip label. Nonetheless, there is a valid motivation for shingling, which is illustrated in Sect. 6.3.3.

In Fig. 6.4 we find one conditioning variable (*Car Type*) and one axis variable (*Gas Mileage*). The panel plot is a boxplot. Strip labels have been omitted as the categories can be annotated traditionally.

An example of a more complex trellis display can be found in Fig. 6.5. For the same cars data set as in Fig. 6.4, the scatterplot of *MPG* vs. *Weight* is plotted. Thus the panel plot is a scatterplot. The axis variables are *MPG* and *Weight*. The grid is set up by the two conditioning variables *Car Type* along *x* and *Drive* along *y*. A fifth variable is included as adjunct variable. The *Number of Cylinders* is included by coloring the points of the scatterplots. The upper strip label shows the category of *Drive*, the lower strip label that of *Car Type*. In Fig. 6.5 we find a common problem of trellis displays. Although the data set has almost 400 observations, 3 of the 18 panels are empty, and 3 panels have fewer than 4 observations.

6.3.2 Trellis Display vs. Mosaic Plots

Trellis displays and mosaicplots do not have very much in common. This can be seen when comparing Figs. 6.1 and 6.6. Obviously the panel plot is not a 2-D mosaicplot, which makes the comparison a bit difficult. On the other hand, the current implementations of trellis displays in R do not offer mosaicplots as panel plots, either.

In Fig. 6.6 the interaction structure is far harder to perceive than in the original mosaicplot. In a mosaicplot the presence of independence can be seen by a straight crossing of the dividing gaps of the categories (in Fig. 6.1 the user preference and the prior usage of product M can be regarded as independent for soft water and low temperatures; see lower right panel in the figure). But what does independence look like in the conditioned barchart representation of the trellis display of Fig. 6.6? Two variables in the panel plot are independent iff the ratios of all corresponding pairs of

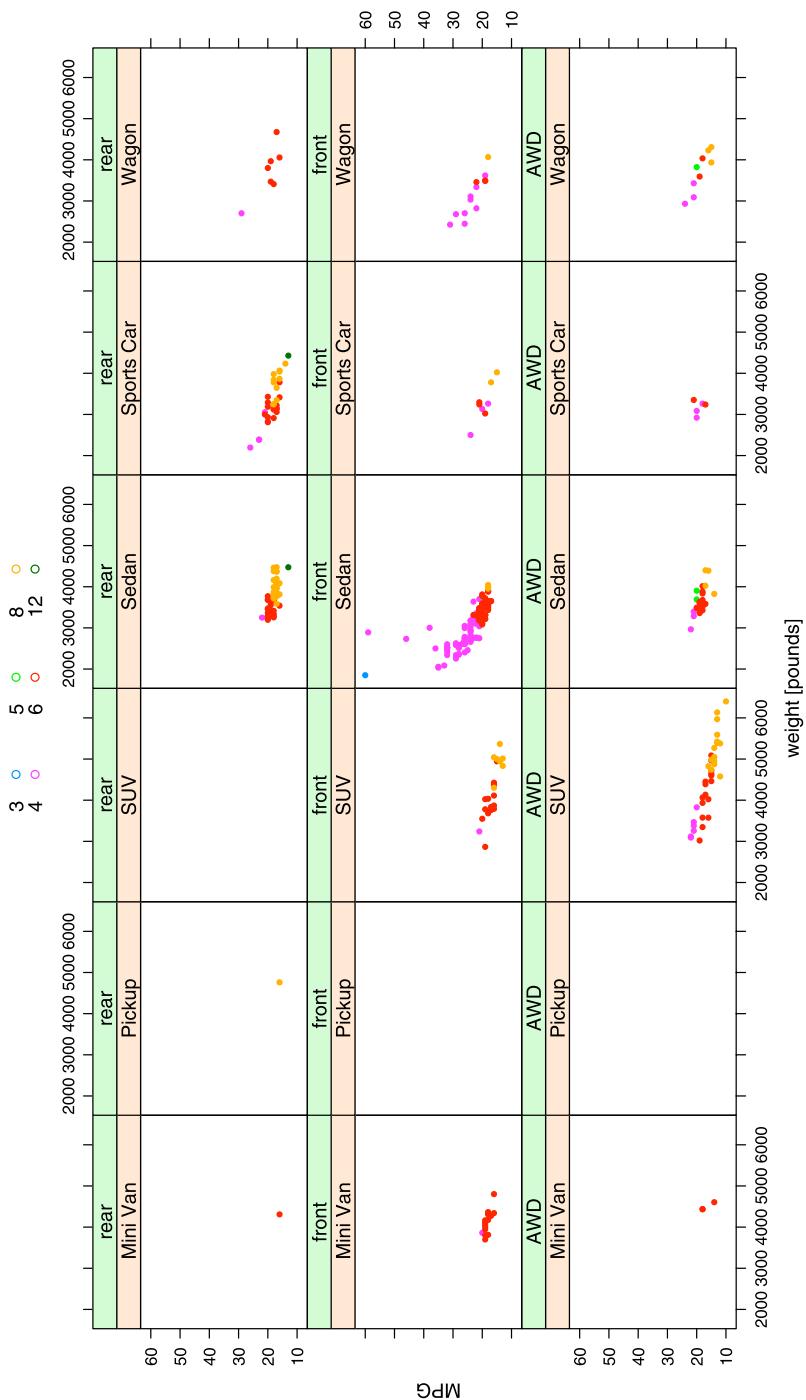


Figure 6.5. [This figure also appears in the color insert.] A trellis display incorporating five variables of the cars data set

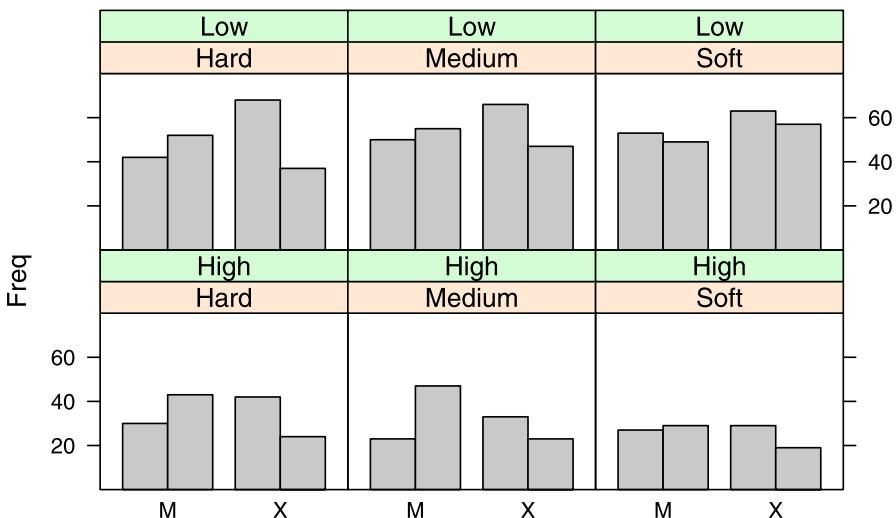


Figure 6.6. A trellis display of the detergent data from Figs. 6.1 and 6.3

levels of two variables are equal, i.e., the two barcharts are identical except for a scaling factor. Thus the only independence can be found in the panel for low temperature and soft water. Obviously it is hard to compare the ratios for more than just two levels per variable and for large absolute differences in the cell counts. Furthermore, it is even harder to quantify and compare interactions. This is because it is nontrivial to simultaneously judge the influence of differences in ratio and absolute cell sizes.

Nonetheless, there exist variations of mosaicplots (see Hofmann, 2008, Chapter III.13 same volume) that use an equal-sized grid to plot the data. Mosaic-plot vari-

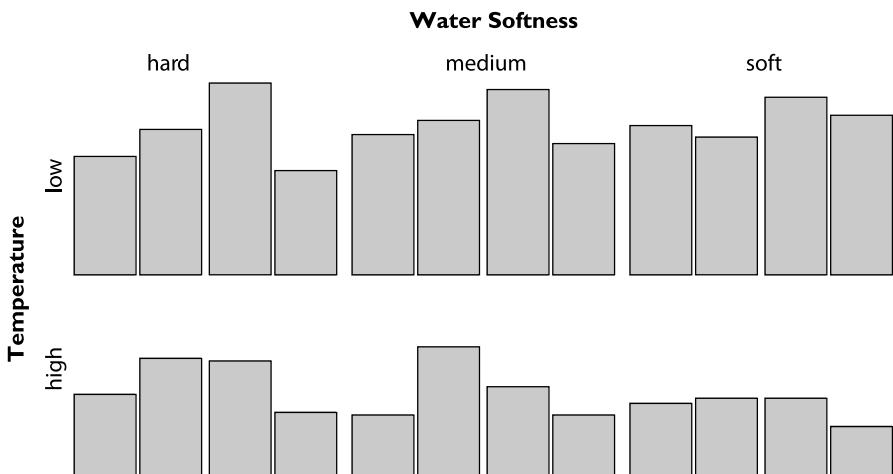


Figure 6.7. A mosaicplot in multiple multiple-barchart variation of the detergent data set that conforms exactly to the representation used in the trellis display of Fig. 6.6

ations using an equal-sized grid to plot data are *same bin size, fluctuation diagrams*, and *multiple barcharts*.

Figure 6.7 shows a mosaicplot in multiple multiple-barchart view with splitting directions x, y, x, x . The way the information is plotted is exactly the same as in Figs. 6.6 and 6.7. Flexible implementations of mosaicplots offering these variations can be found in Mondrian (Theus, 2002) and MANET (Unwin et al., 1996).

Trellis Displays and Interactivity

6.3.3

The conditional framework in a trellis display can be regarded as static snapshots of interactive statistical graphics. The single view in a panel of a trellis display can also be thought of as the highlighted part of the graphics of the panel plot for the conditioned subgroup. This can be best illustrated by looking at the cars data set again. Figure 6.8 shows a screenshot of an interactive session. Selecting a specific subgroup in a barchart or mosaicplot is one interaction. Another interaction would be brushing. Brushing a plot means to steadily move a brush, i.e., an indicator for the selection region, along one or two axes of a plot. The selected interval from the brush can be seen as an interval of a shingle variable. When a continuous variable is subdivided into, e.g., five intervals, this corresponds to five snapshots of the continuous brushing process from the minimum to the maximum of that variable. For the same scatterplot shown in Fig. 6.8, Fig. 6.9 shows a snapshot of a brush selecting the lowest values of the conditioning variables *Engine Size* and *Horsepower*. Now the motivation of shingle variables is more obvious, as they relate directly to this interactive technique. Brushing with linked highlighting is certainly far more flexible than the static view in a trellis display. On the other hand, the trellis display can easily be reproduced in printed form, which is impossible for the interactive process of brushing.

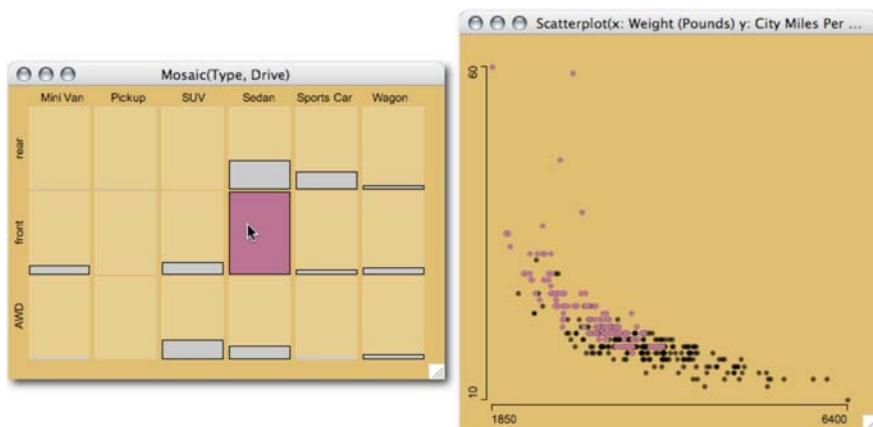


Figure 6.8. Selecting the group of front-wheel-drive sedans in the mosaicplot in multiple-barchart view (left), one gets the corresponding panel plot (scatterplot on right) from Fig. 6.5 in the highlighted subgroup

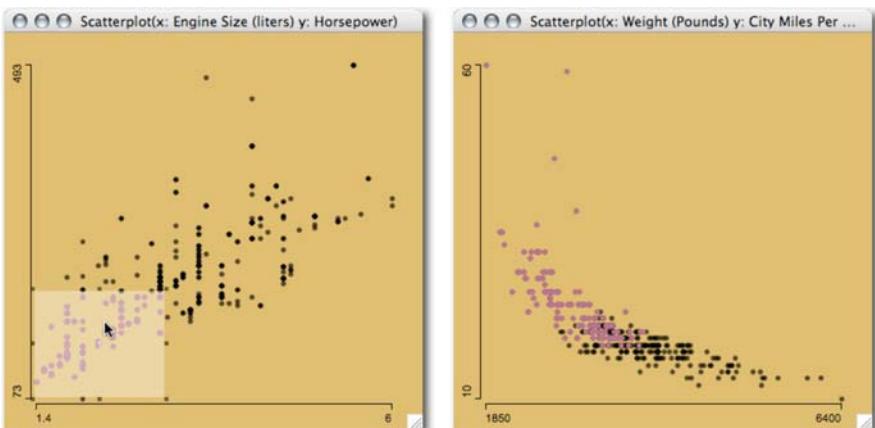


Figure 6.9. Brushing in the conditioning scatterplot (*left*), one gets the panel plot (*scatterplot on *right**) from Fig. 6.5 in the highlighted subgroup

6.3.4

Visualization of Models

The biggest advantage of trellis displays is the common scale among all plot panels. This allows an effective comparison of the panel plots between rows, columns, and pages, depending on the number of conditioning variables and the type of panel plot. Trellis displays are most powerful when used for model diagnostics. In model diagnostics one is most interested in understanding for what data the model fits well and for which cases it does not.

In a trellis display the panel plot can incorporate model information like fitted curves or confidence intervals conditioned for exactly the subgroup shown in the panels. For each panel, the fit and its quality can then be investigated along with the raw data. Figure 6.10 shows the same plot as in Fig. 6.5 except for the adjunct variable. Each scatterplot has a lowess smoother superimposed. One problem with trellis displays is the fact that it is hard to judge the number of cases in a panel plot. For example, in Fig. 6.10 it would be desirable to have confidence bands for the scatterplot smoother in order to be able to judge the variability of the estimate across panels.

Wrap-up

As can be seen from the examples in this section, trellis displays are most useful for continuous axis variables, categorical conditioning variables, and categorical adjunct variables. Shingling might be appropriate under certain circumstances, but it should generally be avoided to ease interpretability.

The major advantage of trellis displays over other multivariate visualization techniques is the flat learning curve of such a display and the possibilities of static reproduction as current trellis display implementations do not offer any interactions. Trellis displays also offer the possibility of easily adding model information to the plots.

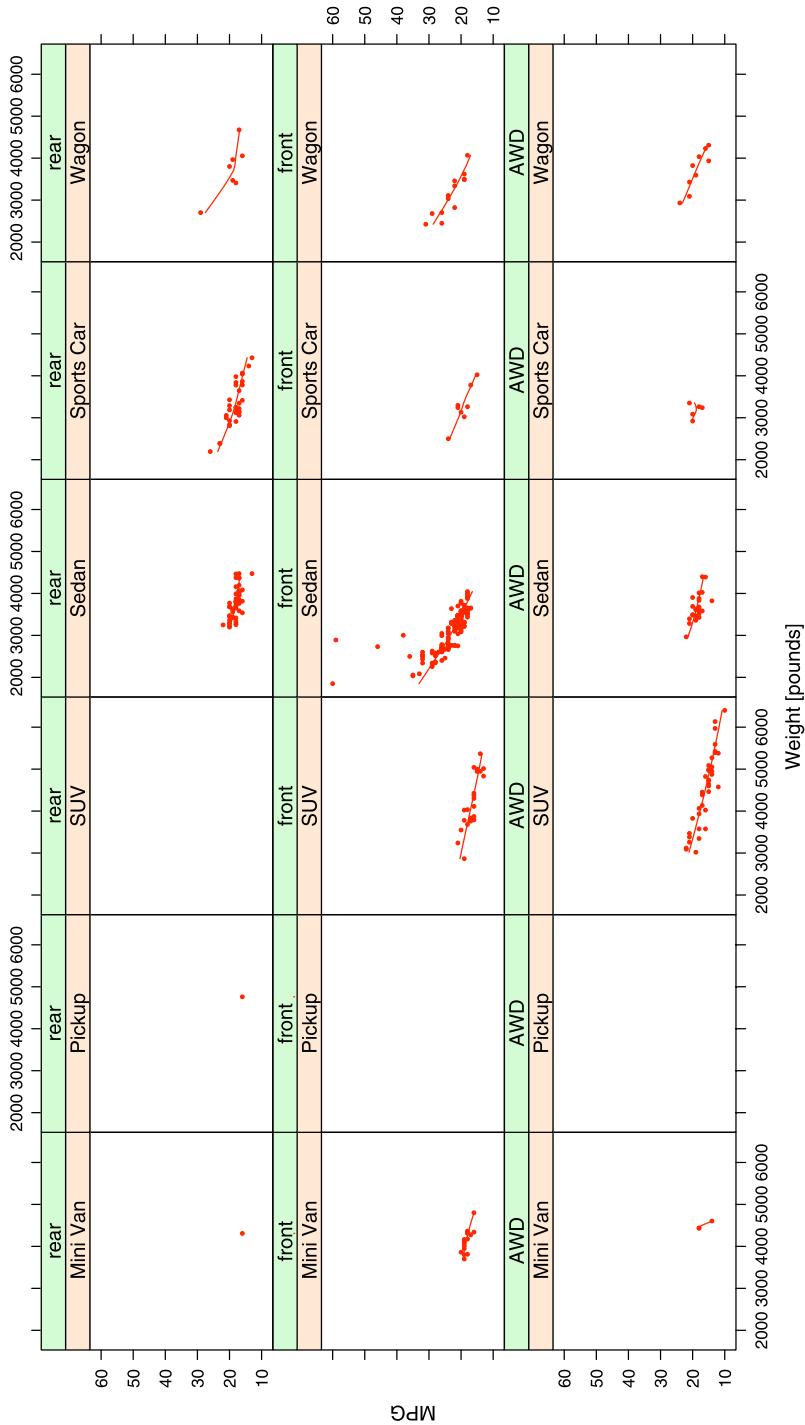


Figure 6.10. The same trellis display as in Fig. 6.5 with an additional lowess smoother superimposed

Nevertheless, interactive linked graphics are usually more flexible in exploratory data analysis applications. Linking the panel plot to barcharts or mosaicplots of the conditioning variables and/or adjunct variables or brushing over a shingle variable is more flexible, though these techniques lack the global overview and the possibility of static reproduction.

6.4

Parallel Coordinate Plots

Parallel coordinate plots, as described by Inselberg (2008, Chapter III.14 same volume), escape the dimensionality of two or three dimensions and can accommodate many variables at a time by plotting the coordinate axes in parallel. They were introduced by Inselberg (1985) and discussed in the context of data analysis by Wegman (1990).

6.4.1

Geometrical Aspects vs. Data Analysis Aspects

Whereas in Inselberg (2008, Chapter III.14 same volume), the geometrical properties of parallel coordinate plots are emphasized to visualize properties of high-dimensional data-mining and classification methods, this section will investigate the main use of parallel coordinate plots in data analysis applications. The most interesting aspects in using parallel coordinate plots are the investigation of groups/clusters, outliers, and structures over many variables at a time. Three main uses of parallel coordinate plots in exploratory data analysis can be identified as the following:

— Overview

No other statistical graphic can plot so much information (cases and variables) at a time. Thus parallel coordinate plots are an ideal tool to get a first overview of a data set. Figure 6.11 shows a parallel coordinate plot of almost 400 cars with 10 variables. All axes have been scaled to *min-max*. Several features, like a few very expensive cars, three very fuel-efficient cars, and the negative correlation between car size and gas mileage, are immediately apparent.

— Profiles

Despite the overview functionality, parallel coordinate plots can be used to visualize the profile of a single case via highlighting. Profiles are not only restricted to single cases but can be plotted for a whole group, to compare the profile of that group with the rest of the data.

Using parallel coordinate plots to profile cases is especially efficient when the coordinate axes have an order like time.

Figure 6.12 shows an example of a single profile highlighted – in this case, the most fuel-efficient car.

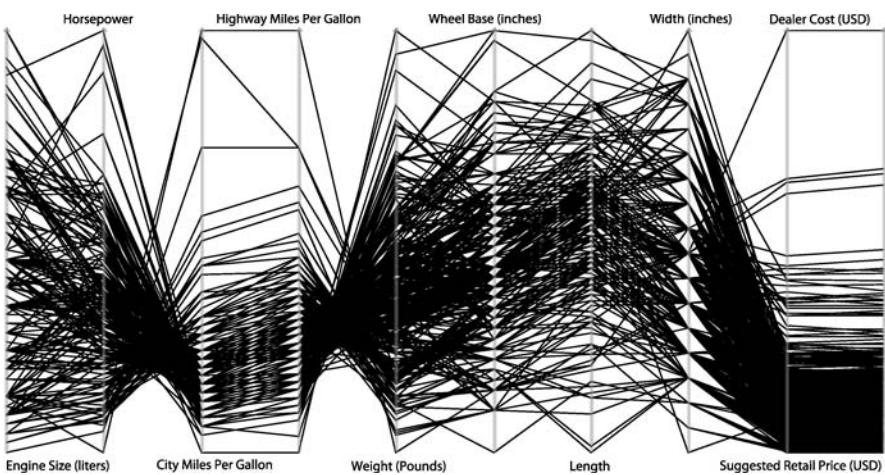


Figure 6.11. Parallel coordinate plot for 10 variables on almost 400 cars

— Monitor

When working on subsets of a data set parallel coordinate plots can help to relate features of a specific subset to the rest of the data set. For instance, when looking at the result of a multidimensional scaling procedure, parallel coordinate plots can help to find the major axes, which influence the configuration of the MDS. Figure 6.13 shows a 2-D MDS along with the corresponding parallel coordinate plot. Querying the leftmost cases in the MDS shows that these cars are all hybrid cars with very high gas mileage. The top right cases in the MDS correspond to heavy cars like pickups and SUVs. Obviously, similar results could have been found with biplots.



Figure 6.12. PCP from Fig. 6.11 with the most fuel-efficient car (Honda Insight 2dr.) highlighted

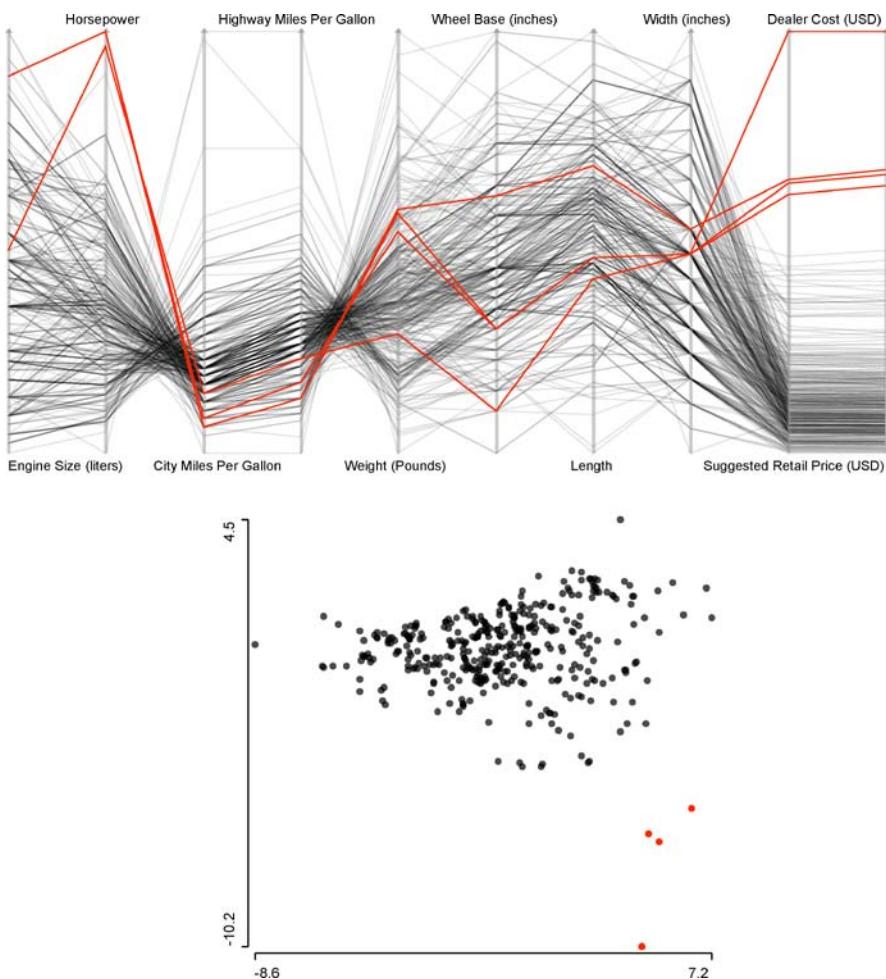


Figure 6.13. MDS (bottom) and parallel coordinate plot (top) of the cars data set. The four cars in the lower right of the MDS are highlighted and happen to be the most expensive and most powerful cars in the data set

6.4.2 Limits

Parallel coordinates are often overrated with respect to the insight they provide into multivariate features of a data set. Obviously scatterplots are superior for investigating 2-D features, but scatterplot matrices (SPLOMs) need far more space to plot the same information as PCPs. Even the detection of multivariate outliers is not something that can usually be directly aided by parallel coordinates. Detecting features in parallel coordinates that are **not** visible in a 1-D or 2-D plot is rare. On the other hand, parallel coordinate plots are extremely useful for interpreting the findings of

multivariate procedures like outlier detection, clustering, or classification in a highly multivariate context.

Although parallel coordinate plots can handle many variables at a time, their rendering limits are reached very soon when plotting more than only a few hundreds of lines. This is due to overplotting, which is far worse than with scatterplots, since parallel coordinate plots only use one dimension to plot the information and the glyph used (a line) prints far more ink than the glyphs in a scatterplot (points). One solution to coping with overplotting is to use α -blending. When α -blending is used, each polygon is plotted with only $\alpha\%$ opacity, i.e., $(1 - \alpha)\%$ transparency. With smaller α values, areas of high line density are more visible and hence are better contrasted to areas with a small density.

Figures 6.11 to 6.13 use α -blending to make the plots better readable or to emphasize the highlighted cases. In the cars data set we only look at fewer than 400 cases, and one can imagine how severe the overplotting will get once thousands of polylines are plotted.

Figures 6.14 and 6.15 show two examples of how useful α -blending can be. The so-called “Pollen” data used in Fig. 6.14 come from an ASA data competition in the late 80s. The data are completely artificial and have the word “E U R E K A” woven into the center of the simulated normal distributions. The almost 4000 cases in five dimensions produce a solid black band without any α -blending applied. Going down to an alpha value of as little as 0.005 will reveal a more solid thin line in the center of the data. Zooming in on these cases will find the word “Eureka,” which just increases the simulated density in the center enough to be visible.

The data in Fig. 6.15 are real data from Forina et al. (1983) on the fatty acid content of Italian olive oil samples from nine regions. The three graphics show the same plot of all eight fatty acids with α -values of 0.5, 0.1, and 0.05. Depending on the amount of α -blending applied, the group structure of some of the nine regions is more or less visible.

Note that it is hard to give general advice on how much α -blending should be applied because the rendering system and the actual size of the plot may change its

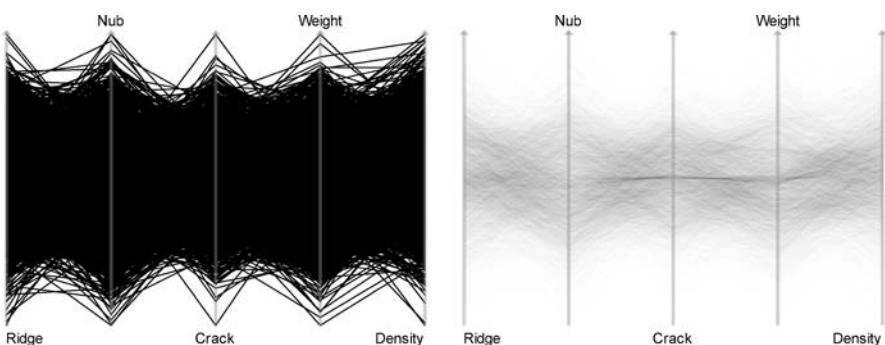


Figure 6.14. The “Pollen” data with $\alpha = 1$ (left) and $\alpha = 0.005$ (right)

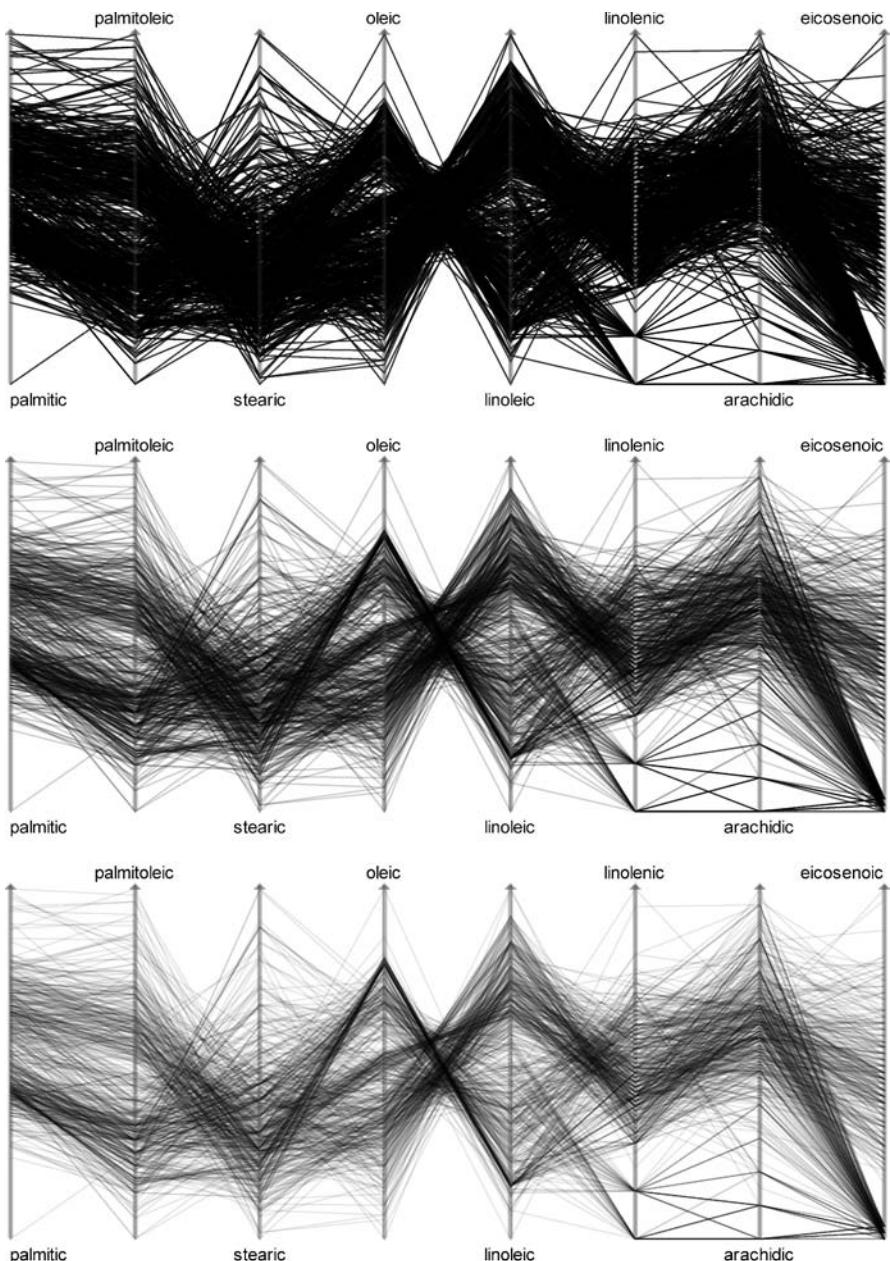


Figure 6.15. The “Olive Oils” data with $\alpha = 0.5$ (top), $\alpha = 0.1$ (middle), and $\alpha = 0.05$ (bottom)

appearance substantially. As with many exploratory techniques, the user should experiment with different settings until he or she feels comfortable enough with the insight gained.

Sorting and Scaling Issues

6.4.3

Parallel coordinate plots are especially useful for variables which either have an order such as time or all share a common scale. In these cases, scaling and sorting issues are very important for a successful exploration of the data set.

Sorting

Sorting in parallel coordinate plots is crucial for the interpretation of the plots, as interesting patterns are usually revealed at neighboring variables. In a parallel coordinate plot of k variables, only $k - 1$ adjacencies can be investigated without reordering the plot. The default order of a parallel coordinate plot is usually the sequence in which the variables are passed to the plotting routine, in most cases the sequence in the data file itself. In many situations this order is more or less arbitrary. Fortunately, one only needs $\lfloor \frac{k+1}{2} \rfloor$ different orderings to see all adjacencies of k variables (see Wegman, 1990).

Whenever all, or at least groups of, variables share the same scale, it is even more helpful to be able to sort these variables according to some criterion. This can be statistics of the variables (either all cases or just a selected subgroup) like minimum, mean, range, or standard deviation, the result of a multivariate procedure, or even some external information. Sorting axes can reduce the visual clutter of a parallel coordinate plot substantially.

If data sets are not small, sorting options have to be provided both manually and automatically.

Scalings

Besides the default scaling, which is to plot all values over the full range of each axis between the minimum and the maximum of the variable, several other scalings are useful. The most important scaling option is to either individually scale the axes or to use a common scale over all axes. Other scaling options define the alignment of the values, which can be aligned at:

- The mean
- The median
- A specific case
- A specific value

For an aligned display, it is not obvious what the range of the data should be when an individual scale is chosen. For individual scales, a 3σ scaling is usually a good choice to map the data onto the plot area.

Alignments do not force a common scale for the variables. Common scaling and alignments are independent scaling options.

Figure 6.16 shows a parallel coordinate plot for the individual stage times of the 155 cyclists who finished the 2005 Tour de France bicycle race. In the upper plot we

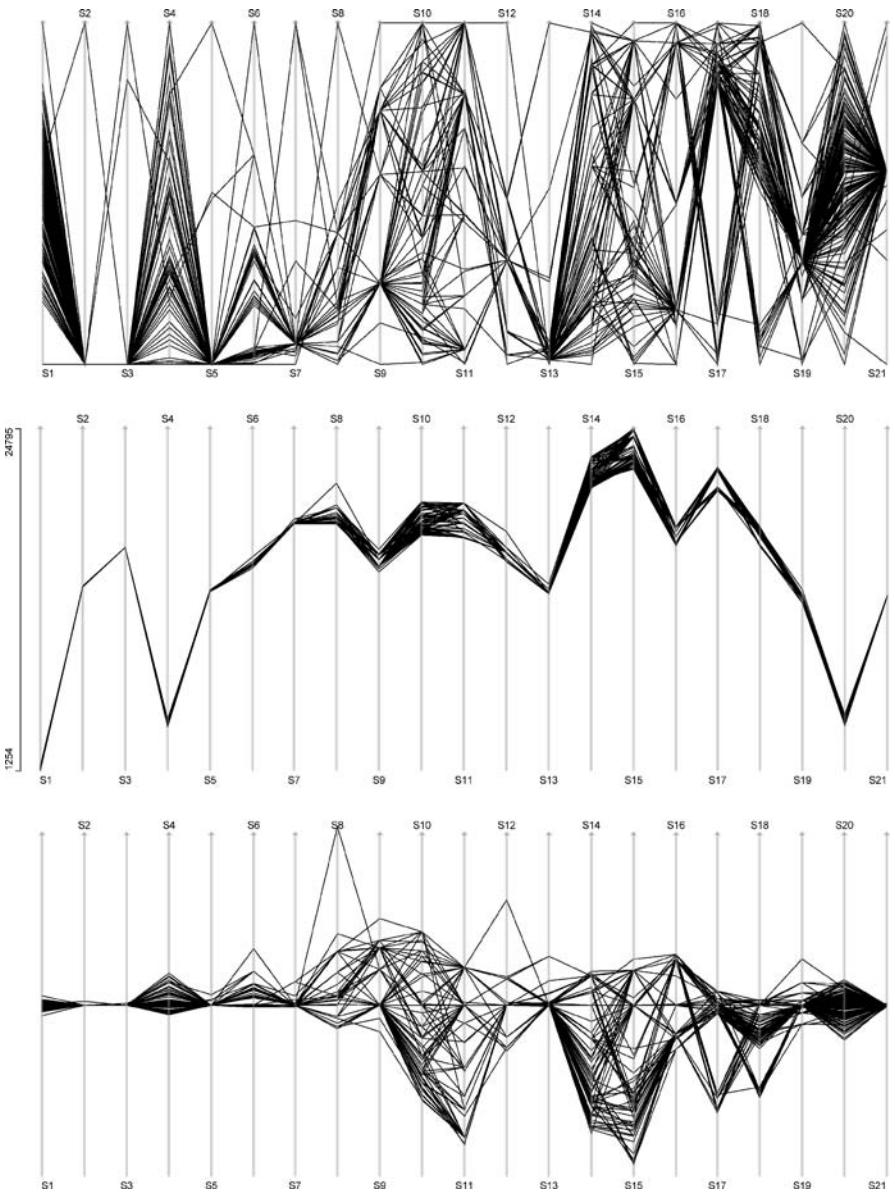


Figure 6.16. Three scaling options for the stage times in the Tour de France 2005: *Top*: all stages are scaled individually between minimum and maximum value of the stage (usual default for parallel coordinate plots) *Middle*: a common scale is used, i.e., the minimum/maximun time of all stages is used as the global minimum/maximun for all axes (this is the only scaling option where a global and common axis can be plotted) *Below*: common scale for all stages, but each stage is aligned at the median value of that stage, i.e., differences are comparable, locations not

see the default min-max scaling. Except for some local groups, not much can be seen from this plot. The middle plot shows the common scaling option for the same data. Now the times are comparable, but due to the differences in absolute time needed for a short time trial and a hard mountain stage, the spread between the first and the last cyclist is almost invisible for most of the stages. Again, except for some outliers, there is hardly anything to see in this representation. The lower plot in Fig. 6.16 shows the same data as the upper two plots, but now each axis is aligned at its median (the median has the nice interpretation of capturing the time of the peloton). Note that the axes still have the same scale, i.e., time differences are still comparable, but now are aligned at the individual medians. This display option clearly reveals the most information.

For a better description of the race as a whole, it is sensible to look at the cumulative times instead of the stage times. Figure 6.17, left, shows a parallel coordinate plot for the cumulative times for each of the 155 cyclists who completed the tour for the corresponding stage. The scaling is the typical default scale usually found in parallel coordinate plots, i.e., individually scaled between minimum and maximum of each axis. All drivers of the team “Discovery Channel” are selected. Although this scaling option gives the highest resolution for these data, it is desirable to have a common scale for all axes. A simple common scale won’t do the trick here, as the cumulative times keep growing, dwarfing the information of the early stages. Figure 6.17, right, uses a common scale, but additionally each axis is aligned at the median of each variable. (Time differences at early stages are not very interesting for the course of the race). Figure 6.17, right, now shows nicely how the field spreads from stage to stage and how the mountain stages (e.g., stages 14 to 16 are stages in the Alps) spread the field far more than flat stages. The drivers of the team “Discovery Channel” are also selected in this plot, showing how the team was separated during the course of the race, although most of the cyclists remained in good positions, supporting the later winner of the race.

The development of the race can be compared in Fig. 6.18, where the plot from Fig. 6.17, right, is shown along with two profile plots. The upper profile plot shows the cumulative category of the stage, which is the sum of 5 minus the category of a mountain in the stage. The peaks starting at stages 8 and 14 nicely indicate the mountain stages in the Pyrenees and the Alps. The lower profile plot gives the average speed of the winner of each stage. Obviously both profiles are negatively correlated.

Wrap-up

6.4.4

Parallel coordinate plots are not very useful “out of the box,” i.e., without features like α -blending and scaling options. The examples used in this chapter show how valuable these additions are in order to get a sensible insight into high-dimensional continuous data. Highlighting subgroups can give additional understanding of group structures and outliers.

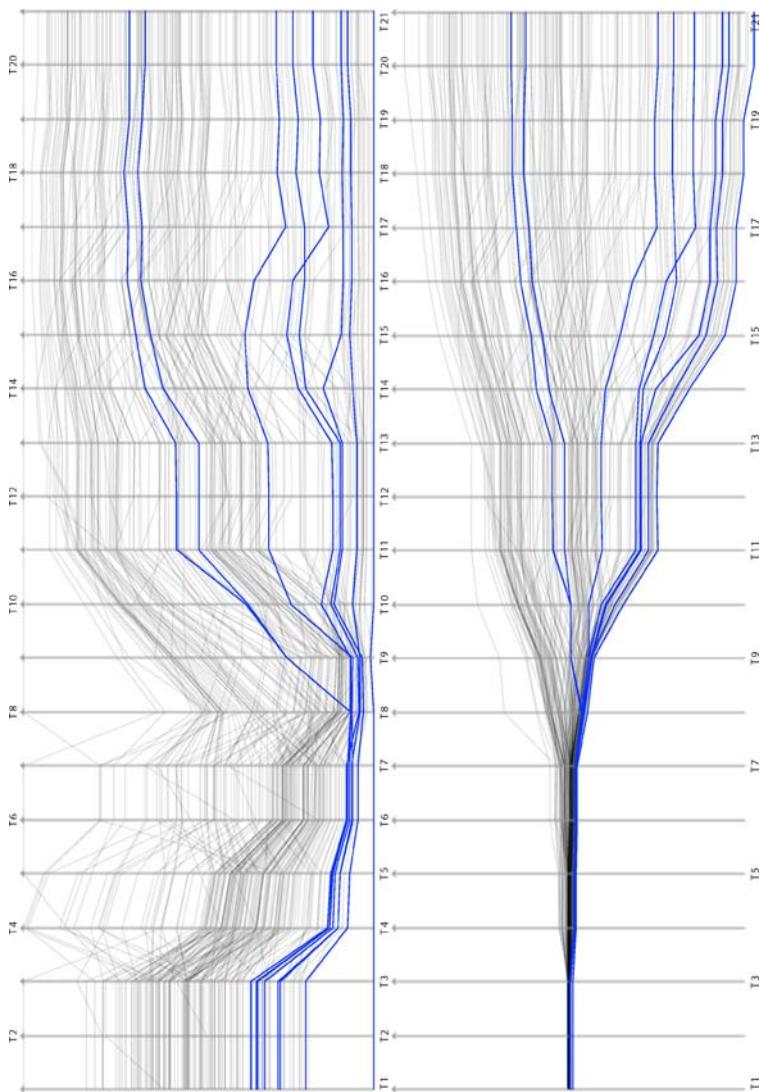


Figure 6.17. Results from 2005 Tour de France. *Left:* each axis shows the cumulative results for all 155 cyclists who finished the tour. *Right:* common scaling applied and axes aligned at medians of each stage. (The eight riders from team “Discovery Channel” are selected)

6.5 Projection Pursuit and the Grand Tour

The grand tour (see Buja and Asimov, 1986) is by definition (see Chen et al., 2008, Chapter III.2) a purely interactive technique. Its basic definition is:

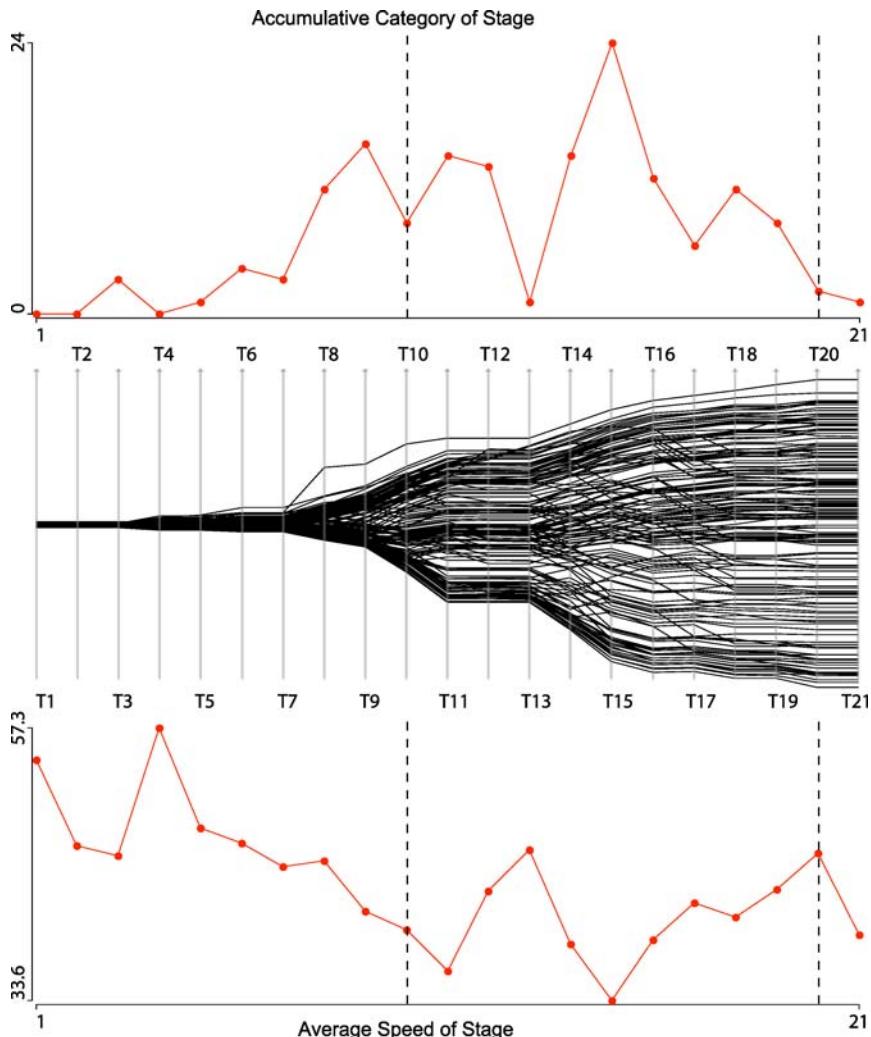


Figure 6.18. The same plot as Fig. 6.17, right, shown in the middle, with a profile plot of the cumulative category of the mountains in the stage (top) and the average speed (of the winner) of each stage (bottom)

A continuous 1-parameter family of d -dimensional projections of p -dimensional data that is dense in the set of all d -dimensional projections in \mathbb{R}^p . The parameter is usually thought of as time.

For a 3-D rotating plot, parameter p equals 3 and parameter d equals 2. In contrast to the 3-D rotating plot, the grand tour does not have classical rotational controls but uses successive randomly selected projections. Figure 6.19 shows an example of three successive planes P_1 , P_2 , and P_3 in three dimensions.

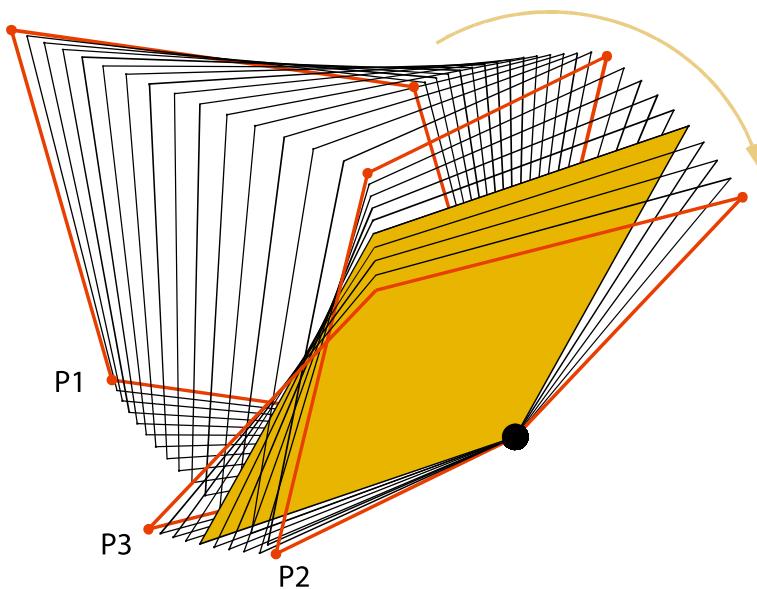


Figure 6.19. Example path of a grand tour

The planes between the randomly selected base planes are interpolated to get a smooth pseudorotation, which is comparable to a physical 3-D rotation. A more technical description of the grand tour can be found in Buja et al. (1999). Although the human eye is not very well trained to recognize rotations in more than three dimensions, the grand tour helps reveal structures like groups, gaps, and dependencies in high-dimensional data, which might be hard to find along the orthogonal projections.

Although projections are usually monitored using a scatterplot, any other plot like a histogram or parallel coordinate plot can be used to display the projected data (see, e.g., Wegman, 1991).

Projection pursuit is a means to get more guidance during the rotation process. A new projection plane is selected by optimizing a projection pursuit index, which measures a feature like point mass, holes, gaps, or other target structures in the data.

6.5.1 Grand Tour vs. Parallel Coordinate Plots

The grand tour is a highly exploratory tool, even more so than the other methods discussed in this chapter. Whereas in classical parallel coordinate plots the data are still projected to orthogonal axes, the grand tour permits one to look at arbitrary nonorthogonal projections of the data, which can reveal features invisible in orthogonal projections. Figure 6.20 shows an example of three projections of the cars data set using the grand tour inside ggobi (see Swayne et al., 2003). The cases are colored according to the number of cylinders. Each plot has the projection directions of the 11 variables added at the lower left of the plot. Obviously these static screenshots of

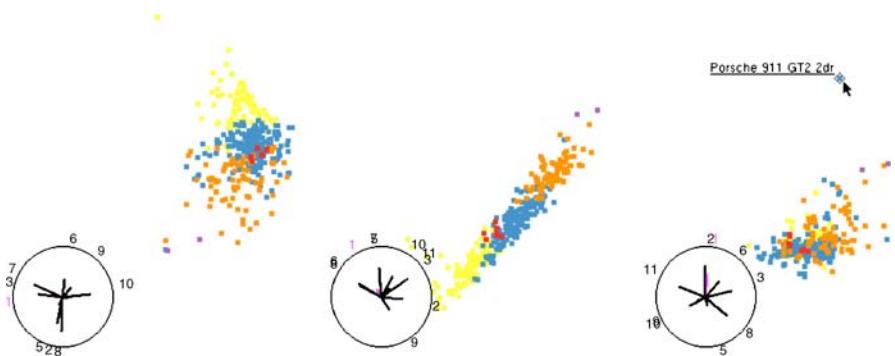


Figure 6.20. Three example screenshots of three different projections of the cars data set. The cases are colored according to the number of cylinders. The *rightmost plot* has the least discrimination of the groups but the strongest separation of an outlier, the “Porsche GT”

the projections are not very satisfactory unless they reveal a striking feature. Whereas the parallel coordinate plot of the same data (cf. Fig. 6.11) can at least show the univariate distributions along with some bivariate relationships, the grand tour fails to do so, and focuses solely on the multivariate features, which may be visible in certain projections.

The grand tour can help to identify the geometry of variables beyond the limits of the three dimensions of a simple rotating plot. Nevertheless, examples of structures in more than five dimensions are rare, even when using the grand tour. In these cases the fixed geometrical properties of parallel coordinate plots seem to be an advantage.

Monitoring the projected data in parallel coordinates instead of a simple scatterplot is a promising approach to investigating data beyond ten or even more dimensions. Unfortunately, only very few flexible implementations of the grand tour and projection pursuit exist, which limits the possibility of a successful application of these methods.

Recommendations

6.6

This chapter showed the application, strengths, and weaknesses of the most important high-dimensional plots in statistical data visualization. All plots have their specific field of application, where no other method delivers equivalent results. The fields of application are broader or narrower depending on the method. All four methods and techniques discussed in this chapter, i.e., mosaicplots, trellis displays, parallel coordinate plots, and the grand tour and projection pursuit, need a certain amount of training to be effective. Furthermore, training is required to learn the most effective use of the different methods for different tasks.

Some of the plots are optimized for presentation graphics (e.g., trellis displays), others, in contrast only make sense in a highly interactive and exploratory setting (e.g., grand tour).

The high-dimensional nature of the data problems for the discussed plots calls for interactive controls – be they rearrangements of levels and/or variables or different scalings of the variables or the classical linked highlighting – which put different visualization methods together into one framework, thus further increasing the dimensionality.

Figure 6.21 illustrates the situation. When analyzing high-dimensional data, one needs more than just one visualization technique. Depending on the scale of the variables (discrete or continuous) and the number of variables that should be visualized simultaneously, one or another technique is more powerful. All techniques – except for trellis displays – have in common that they only rise to their full power when interactive controls are provided. Selection and linking between the plots can bring the different methods together, which then gives even more insight. The results found in an exploration of the data may then be presented using static graphics. At this point, trellis displays are most useful to communicate the results in an easy way.

Finally, implementations of these visualization tools are needed in software. Right now, most of them are isolated features in a single software package. Trellis displays can only be found in the R package, or the corresponding commercial equivalent

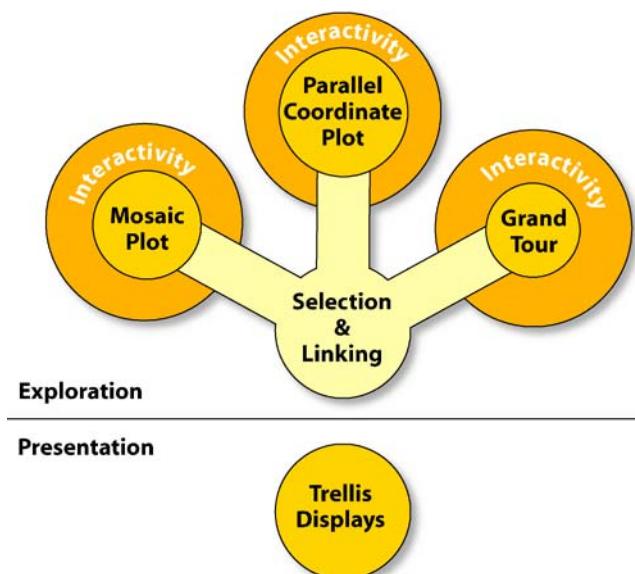


Figure 6.21. Diagram illustrating the importance of interactivity and linking of the high-dimensional visualization tools in statistical graphics

S-Plus, and grand tour and projection pursuit only in the *ggobi* package. It would be desirable to have a universal tool that could integrate all of the methods in a highly interactive and tightly linked way.

References

- Becker, R., Cleveland, W. and Shyu, M. (1996). The visual design and control of trellis displays, *Journal of Computational and Graphical Statistics* 6(1):123–155.
- Buja, A. and Asimov, D. (1986). Grand tour methods: An outline, *17th Symposium on the Interface of Computer Science and Statistics*, pp. 63–67.
- Buja, A., Cook, D., Asimov, D. and Hurley, C.B. (1999). *Theory and computational methods for dynamic projections in high-dimensional data visualization*. A monograph on rendering techniques, mathematics, and algorithms for tours and related methods. <http://www.research.att.com/areas/stat/xgobi/papers/dynamic-projections.ps.gz>
- Chen, C.-H., Härdle, W. and Unwin, A. (eds) (2008). *Handbook of Data Visualization*, Springer, Heidelberg.
- Cox, D.R. and Snell, E.J. (1991). *Applied Statistics – Principles and Examples*, Chapman & Hall, London.
- Fahrmeir, L. and Tutz, G. (1994). *Multivariate Statistical Modelling Based on Generalized Linear Models*, Springer, New York.
- Forina, M., Armanino, C., Lanteri, S. and Tiscornia, E. (1983). Classification of olive oils from their fatty acid composition, in H. Martens and H. Russwurm (eds), *Food Research and Data Analysis*, Applied Science Publishers, London UK, pp. 189–214.
- Friendly (1994). Mosaic displays for multi-way contingency tables, *Journal of the American Statistical Association* 89:190–200.
- Hartigan, J.A. and Kleiner, B. (1981). Mosaics for contingency tables., *13th Symposium on the Interface*, Springer Verlag, New York, pp. 268–273.
- Hofmann, H. (2000). Exploring categorical data: interactive mosaic plots, *Metrika* 51(1):11–26.
- Hofmann, H. (2008). *Mosaicplots and their Variations*, in Chen et al. (2008).
- Inselberg, A. (1985). The Plane with Parallel Coordinates, *The Visual Computer* 1:69–91.
- Inselberg, A. (2008). *Parallel Coordinates: Visualization and Classification of High Dimensional Datasets*, in Chen et al. (2008).
- Meyer, D., Zeileis, A. and Hornik, K. (2008). *Visualizing Multi-way Contingency Tables*, in Chen et al. (2008).
- Swayne, D.F., Lang, D.T., Buja, A. and Cook, D. (2003). GGobi: evolving from XGobi into an extensible framework for interactive data visualization, *Computational Statistics and Data Analysis* 43(4):423–444.
- Theus, M. (1999). Trellis displays, in S. Kotz and C. Read (eds), *Encyclopedia of Statistical Science, Update Volume III*, Wiley, New York.
- Theus, M. (2002). Interactive Data Visualization using Mondrian, *Journal of Statistical Software* 7(11).

- Theus, M. and Lauer, S. (1999). Visualizing loglinear models, *Journal of Computational and Graphical Statistics* 8(3):396–412.
- Unwin, A., Hawkins, G., Hofmann, H. and Siegl, B. (1996). Interactive Graphics for Data Sets with Missing Values – MANET, *Journal of Computational and Graphical Statistics* 5(2):113–122.
- Wegman, E. (1990). Hyperdimensional Data Analysis Using Parallel Coordinates, *Journal of American Statistics Association* 85:664–675.
- Wegman, E. (1991). *The Grand Tour in k-Dimensions*, Technical Report 68, Center for Computational Statistics, George Mason University.
- Wills, G. (2008). *Linked Data Views*, in Chen et al. (2008).

Multivariate Data Glyphs: Principles and Practice

II.7

Matthew O. Ward

7.1	<i>Introduction</i>	180
7.2	<i>Data</i>	180
7.3	<i>Mappings</i>	181
7.4	<i>Examples of Existing Glyphs</i>	182
7.5	<i>Biases in Glyph Mappings</i>	183
7.6	<i>Ordering of Data Dimensions/Variables</i>	184
	Correlation-driven	185
	Symmetry-driven.....	185
	Data-driven	185
	User-driven	186
7.7	<i>Glyph Layout Options</i>	188
	Data-driven Placement	188
	Structure-driven Placement	189
7.8	<i>Evaluation</i>	191
7.9	<i>Summary</i>	195

Introduction

In the context of data visualization, a glyph is a visual representation of a piece of data where the attributes of a graphical entity are dictated by one or more attributes of a data record. For example, the width and height of a box could be determined by a student's score on the midterm and final exam for a course, while the box's color might indicate the gender of the student. The definition above is rather broad, as it can cover such visual elements as the markers in a scatterplot, the bars of a histogram, or even an entire line plot. However, a narrower definition would not be sufficient to capture the wide range of data visualization techniques that have been developed over the centuries that are termed glyphs.

Glyphs are one class of visualization techniques used for multivariate data. Their major strength, as compared to techniques such as parallel coordinates, scatterplot matrices, and stacked univariate plots, is that patterns involving more than two or three data dimensions can often be more readily perceived. Subsets of dimensions can form composite visual features that analysts can be trained to detect and classify, leading to a richer description of interrecord and intrarecord relationships than can be extracted using other techniques.

However, glyphs do have their limitations. They are generally restricted in terms of how accurately they can convey data due to their size and the limits of our visual perception system to measure different graphical attributes. There are also constraints on the number of data records that can be effectively visualized with glyphs; excessive data set size can result in significant occlusion or the need to reduce the size of each glyph, both of which make the detection of patterns difficult, if not impossible. Thus glyphs are primarily suitable for qualitative analysis of modest-sized data sets.

This paper describes the process of glyph generation – the mapping of data attributes to graphical attributes – and presents some of the perceptual issues that can differentiate effective from ineffective glyphs. Several important issues in the use of glyphs for communicating information and facilitating analysis are also discussed, including dimension order and glyph layout. Finally, some ideas for future directions for research on visualization using glyphs are presented.

Data

Glyphs are commonly used to visualize multivariate data sets. *Multivariate data*, also called multidimensional or n -dimensional data, consist of some number of items or records, n , each of which is defined by a d -vector of values. Such data can be viewed as a $d \times n$ matrix, where each row represents a data record and each column represents an observation, variable, or dimension. For the purpose of this paper, we will assume a data item is a vector of scalar numeric values. Categorical and other nonnumeric values can also be visualized using glyphs, though often only after conversion to numeric form (Rosario et al., 2004). Nonscalar values can also be incorporated by

linearizing the embedded vectors or tensors. We also assume that a data set consists of one or more of such data items/records and that for each position in the vector we can calculate a minimum and maximum value. This allows us to normalize the data to facilitate mapping to graphical attributes.

Variables/dimensions can be independent or dependent, which might imply that some ordering or grouping of dimensions could be beneficial. They can be of homogeneous type, such as a set of exam grades, or of mixed/heterogeneous types, such as most census data. This might suggest the use of a consistent mapping (e.g., all dimensions map to line lengths) or separation based on type so that each distinct group of related dimensions might control one type of mapping.

Mappings

7.3

Many authors have developed lists of graphical attributes to which data values can be mapped (Cleveland and McGill, 1984; Cleveland, 1993; Bertin, 1981). These include position (1-, 2-, or 3-D), size (length, area, or volume), shape, orientation, material (hue, saturation, intensity, texture, or opacity), line style (width, dashes, or tapers), and dynamics (speed of motion, direction of motion, rate of flashing).

Using these attributes, a wide range of possible mappings for data glyphs are possible. Mappings can be classified as follows:

- One-to-one mappings, where each data attribute maps to a distinct and different graphical attribute;
- One-to-many mappings, where redundant mappings are used to improve the accuracy and ease at which a user can interpret data values; and
- Many-to-one mappings, where several or all data attributes map to a common type of graphical attribute, separated in space, orientation, or other transformation.

One-to-one mappings are often designed in such a way as to take advantage of the user's domain knowledge, using intuitive pairings of data to graphical attribute to ease the learning process. Examples include mapping color to temperature and flow direction to line orientation. Redundant mappings can be useful in situations where the number of data dimensions is low and the desire is to reduce the possibility of misinterpretation. For example, one might map population to both size and color to ease analysis for color-impaired users and facilitate comparison of two populations with similar values. Many-to-one mappings are best used in situations where it is important to not only compare values of the same dimension for separate records, but also compare different dimensions for the same record. For example, mapping each dimension to the height of a vertical bar facilitates both intrarecord and interrecord comparison. In this paper, we focus primarily on one-to-one and many-to-one mappings, although many of the principles discussed can be applied to other mappings.

Examples of Existing Glyphs

The following list (from Ward, 2002) contains a subset of glyphs that have been described in the literature or are in common use. Some are customized to a particular application, such as visualizing fluid flow, while others are more general purpose. In a later section we examine many of these mappings and try to identify some of their strengths and weaknesses.

- Profiles (du Toit, 1986): height and color of bars (Fig. 7.1a).
- Stars (Siegel et al., 1972): length of evenly spaced rays emanating from center (Fig. 7.1b).
- Anderson/metroglyphs (Anderson, 1957; Gnanadesikan, 1977): length of rays (Fig. 7.1b).
- Stick figures (Pickett and Grinstein, 1988): length, angle, color of limbs (Fig. 7.1c).
- Trees (Kleiner and Hartigan, 1981): length, thickness, angles of branches; branch structure derived from analyzing relations between dimensions (Fig. 7.1c).
- Autoglyph (Beddow, 1990): color of boxes (Fig. 7.1d).
- Boxes (Hartigan, 1975): height, width, depth of first box; height of successive boxes (Fig. 7.1d).
- Hedgehogs (Klassen and Harrington, 1991): spikes on a vector field, with variation in orientation, thickness, and taper.
- Faces (Chernoff, 1973): size and position of eyes, nose, mouth; curvature of mouth; angle of eyebrows (Fig. 7.1e).
- Arrows (Wittenbrink et al., 1996): length, width, taper, and color of base and head (Fig. 7.1f).
- Polygons (Schroeder et al., 1991): conveying local deformation in a vector field via orientation and shape changes.
- Dashtubes (Fuhrmann and Groller, 1998): texture and opacity to convey vector field data.
- Weathervanes (Friedman et al., 1972): level in bulb, length of flags (Fig. 7.1f).
- Circular profiles (Mezzich and Worthington, 1978): distance from center to vertices at equal angles.
- Color glyphs (Levkowitz, 1991): colored lines across a box.
- Bugs (Chuah and Eick, 1998): wing shapes controlled by time series; length of head spikes (antennae); size and color of tail; size of body markings.
- Wheels (Chuah and Eick, 1998): time wheels create ring of time series plots, value controls distance from base ring; 3-D wheel maps time to height, variable value to radius.
- Boids (Kerlick, 1990): shape and orientation of primitives moving through a time-varying field.
- Procedural shapes (Rohrer et al., 1998; Ebert et al., 1999): blobby objects controlled by up to 14 dimensions.
- Glyphmaker (Ribarsky et al., 1994): user-controlled mappings.
- Icon Modeling Language (Post et al., 1995): attributes of a 2-D contour and the parameters that extrude it to 3-D and further transform/deform it.

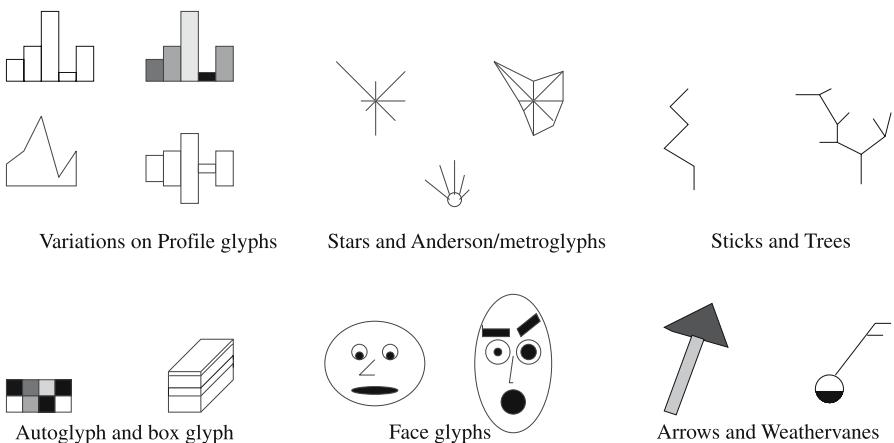


Figure 7.1. Examples of multivariate glyphs (from Ward, 2002)

The list above is ample evidence that a significant number of possible mappings exist, many of which have yet to be proposed or evaluated. The question then becomes determining which mapping will best suit the purpose of the task, the characteristics of the data, and the knowledge and perceptual abilities of the user. These issues are described in the sections below.

Biases in Glyph Mappings

7.5

One of the most common criticisms of data glyphs is that there is an implicit bias in most mappings, i.e., some attributes or relationships between attributes are easier to perceive than others. For example, in profile or star glyphs, relationships between adjacent dimensions are much easier to measure than those that are more separated, and in Chernoff faces, attributes such as the length of the mouth or nose are perceived more accurately than graphical attributes such as curvature or radius.

In this section I attempt to isolate and categorize some of these biases, using both results from prior studies on graphical perception as well as our own empirical studies. It is clear, however, that much more substantial work is needed in measuring and correcting for these biases when designing and utilizing glyphs in data analysis.

Perception-based bias Certain graphical attributes are easier to measure and compare visually than others. For example, Cleveland (1993) reports on experiments that show length along a common axis can be gauged more accurately than, say, angle, orientation, size, or color. Figure 7.2 shows the same data with three different mappings. Relationships are easier to see with the profile glyphs (length on a common base), followed by the star glyphs (length with different orientations). The pie glyph fares the worst, as the user is required to compare angles. Thus in mappings that are not many-to-one (i.e., those that employ a mixture of graphi-

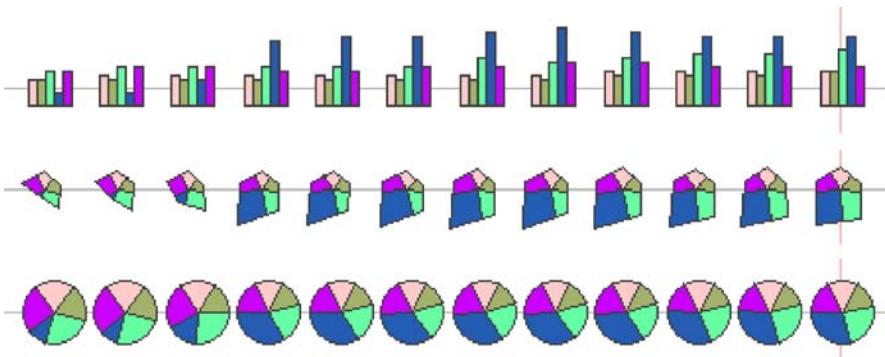


Figure 7.2. Profile, glyphs, and pie glyphs of a subset of data regarding five economic indicators, as generated with SpiralGlyphics (Ward and Lipchak, 2000). Features within and between glyphs are generally easier to compare with profile glyphs

cal attributes), there is an inherent difference in our ability to extract values from different data dimensions.

Proximity-based bias In most, if not all, glyphs, relationships between data dimensions mapped to adjacent features in a glyph are easier to perceive and remember than those mapped to nonadjacent features. To the best of my knowledge, no one has performed experiments to quantify the degree of this bias, although Chernoff and Rizvi (1975) reported as much as 25 % variance in results by rearranging data mappings within Chernoff faces. It is likely that the amount of bias will depend as well on the type of glyph used, as comparing lengths of bars with a common baseline will be easier than comparing the lengths of rays in a star glyph.

Grouping-based bias Graphical attributes that are not adjacent but may be semantically or perceptually grouped may result in the introduction of bias as well. For example, if we map two variables to the size of the ears in a face, the relationship between those variables may be easier to discern than, say, mapping one to the shape of the eye and the other to the size of the adjacent ear.

7.6 Ordering of Data Dimensions/Variables

Each dimension of a data set will map to a specific graphical attribute. By modifying the order of dimensions while preserving the type of mapping, we will generate alternate “views” of the data. However, barring symmetries, there are $N!$ different dimension orderings, and thus distinct views. An important issue in using glyphs is to ascertain which ordering(s) will be most supportive of the task at hand. In this section, I will present a number of dimension-ordering strategies that can be used to generate views that are more likely to provide more information than random ordering.

Correlation-driven

7.6.1

Many researchers have proposed using correlation and other similarity measures to order dimensions for improved visualization. Bertin's reorderable matrix (Bertin, 1981) showed that by rearranging the rows and columns of a tabular display, groups of related records and dimensions could be exposed. Ankerst et al. (1998) used cross-correlation and a heuristic search algorithm to rearrange dimensions for improved interpretability. Friendly and Kwan (2003) introduced the notion of effect ordering, where an ordering of graphical objects or their attributes would be decided based on the effect or trend that a viewer seeks to expose. In particular, they showed that by ordering the dimensions of a star glyph based on their angles in a biplot (basically each dimension is represented by a line whose angle is controlled by the first two eigenvectors), related dimensions would get grouped together. This is related to the method reported by Borg and Staufenbiel (1992), where they compared traditional snowflake and star glyphs with what they called factorial suns, which display each data point using the dimension orientations generated via the first two eigenvectors rather than uniformly spaced angles. Their experiments showed significant improvement by naive users in interpreting data sets.

Symmetry-driven

7.6.2

Gestalt principles indicate we have a preference for simple shapes, and we are good at seeing and remembering symmetry. In Peng et al. (2004), the shapes of star glyphs resulting from using different dimension orders were evaluated for two attributes: monotonicity (the direction of change is constant) and symmetry (similar ray lengths on opposite sides of the glyph). The ordering that maximized the number of simple and symmetric shapes was chosen as the best. User studies showed a strong preference for visualizations using the ordering optimized in this fashion. We conjecture that simple shapes are easier to recognize and facilitate the detection of minor shape variations; for example, shapes with only a small number of concavities and convexities might require less effort to visually process than shapes with many features. Also, if most shapes are simple, it is much more apparent which records correspond to outliers. More extensive formal evaluations are needed to validate these conjectures, however. See Fig. 7.3 for an example.

Data-driven

7.6.3

Another option is to base the order of the dimensions on the values of a single record (base), using an ascending or descending sorting of the values to specify the global dimension order. This can allow users to see similarities and differences between the base record and all other records. It is especially good for time-series data sets to show the evolution of dimensions and their relationships over time. For example, sorting the exchange rates of ten countries with the USA by their relative values in the first year of the time series exposes a number of interesting trends, anomalies, and periods of relative stability and instability (Fig. 7.4). In fact, the original order is nearly reversed at a point later in the time series (not shown).

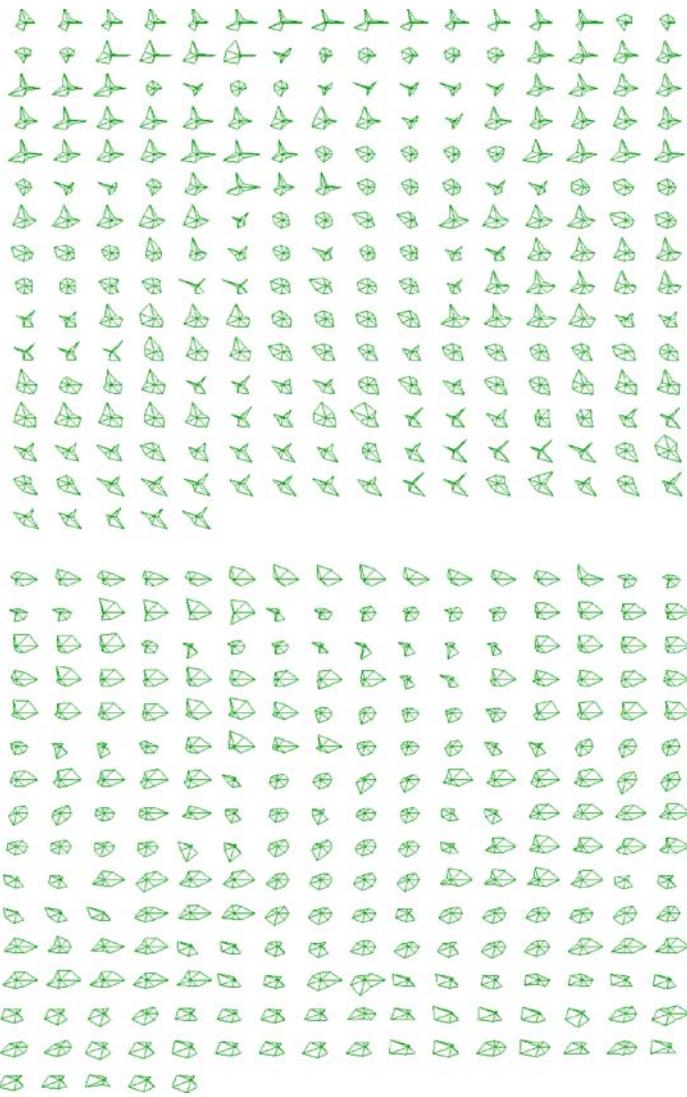


Figure 7.3. Star glyphs for a subset of the cars data set using a random dimension ordering and one based on shape analysis. More simple shapes can be seen in the second version than the first, which we believe can facilitate detection of groups of similar shapes as well as outliers

7.6.4 User-driven

As a final strategy, we can allow users to apply knowledge of the data set to order and group dimensions by many aspects, including derivative relations, semantic similarity, and importance. Derivative relations mean that the user is aware that one or more dimensions may simply be derived through combinations of other dimensions.



Figure 7.4. [This figure also appears in the color insert.] Exchange rate data using the original ordering of dimensions and then ordered by the first data record. Significant features in the ordered version, such as the sudden rise in value of one of the lower currencies during the third year and the progressive alignment of several of the inner currencies, are difficult to detect in the original ordering

These derivative relations might or might not get exposed in correlation-based ordering. Semantic similarities indicate dimensions that have related meanings within the domain; even if the values do not correlate well, users might logically group or order them to help in their analysis task. Finally, some dimensions are likely to have more importance than others for a given task, and thus ordering or assigning such dimensions to more visually prominent features of the glyph (or, in some cases, the features the user is likely to examine first, such as the leftmost bar of a profile glyph) will likely have a positive impact on task performance.

7.7

Glyph Layout Options

The position of glyphs can convey many attributes of data, including data values or structure (order, hierarchy), relationships, and derived attributes. In this section I will describe a taxonomy of glyph layout strategies, presented in detail in (Ward, 2002), based on the following considerations:

- Whether the placement will be data driven, e.g., based on two or more data dimensions, or structure driven, such as methods based on an explicit or implicit order or other relationship between data points.
- Whether overlaps between glyphs will be allowed. This can have a significant impact on the size of the data set that can be displayed, the size of the glyphs used, and the interpretability of the resulting images.
- The tradeoff between optimized screen utilization, such as found in space-filling algorithms, versus the use of white space to reinforce distances between data points.
- Whether the glyph positions can be adjusted after initial placement to improve visibility at the cost of distorting the computed position. Overlapping glyphs can be difficult to interpret, but any movement alters the accuracy of the visual depiction. We need to know, for the given domain, what the tradeoffs are between accuracy and clarity.

7.7.1

Data-driven Placement

Data-driven glyph placement, as the name implies, assumes a glyph will be positioned based entirely on some or all of the data values associated with the corresponding record. We differentiate two classes of such techniques based on whether the original data values are used directly or whether positions are derived via computations involving these data values. An example of the first would be the positioning of markers in a scatterplot using two dimensions (Fig. 7.5), while an example of the second would be to use PCA to generate the x and y coordinates of the resulting glyph (Fig. 7.6). More complex analysis has also been used in glyph placement. Several researchers (Globus et al., 1991; Helman and Hesselink, 1991) have proposed methods for placing glyphs at critical points within flow fields from fluid dynamics



Figure 7.5. Example of positioning glyphs according to two dimensions. In this case, the cars data set displayed with star glyphs using MPG and horsepower to specify position. Groupings of similar shapes and some outliers are visible

simulations. The advantages of using the direct method is that position has an easily interpreted meaning and can act to emphasize or even replace two of the data dimensions. The derived methods can add to the information content of the display and draw the user's attention to implicit relationships between data records.

Data-driven methods almost always result in some overlap between glyphs, which can lead to misinterpretation and undetected patterns. Many methods have been developed to address this problem via distorting the position information. Random jitter is commonly added to positions in plotting, especially for data that take on only a small number of possible values. Other methods use spring- or force-based methods to minimize or eliminate overlaps while also minimizing the displacement of glyphs from their original positions (Keim and Hermann, 1998). Woodruff et al. (1998) proposed a relocation algorithm that attempts to maintain constant density across the display. Since distortion introduces error into the visual presentation, it is best to allow users to control the amount of distortion applied by either setting the maximum displacement for an individual glyph or the average among all glyphs or by using animation to show the movement of glyphs from their original positions to their distorted positions.

Structure-driven Placement

7.7.2

Structure-driven glyph placement assumes the data have some implicit or explicit structural attribute that can be used to control the position. A common type of structure is an ordering relationship, such as in time-series or spatial data. Ordering can also be derived via one or more dimensions (Fig. 7.7). This is different, however, from data-driven placement algorithms in that the data values only define the ordering relationship, which is then used in generating the position. A related structure is a cyclic

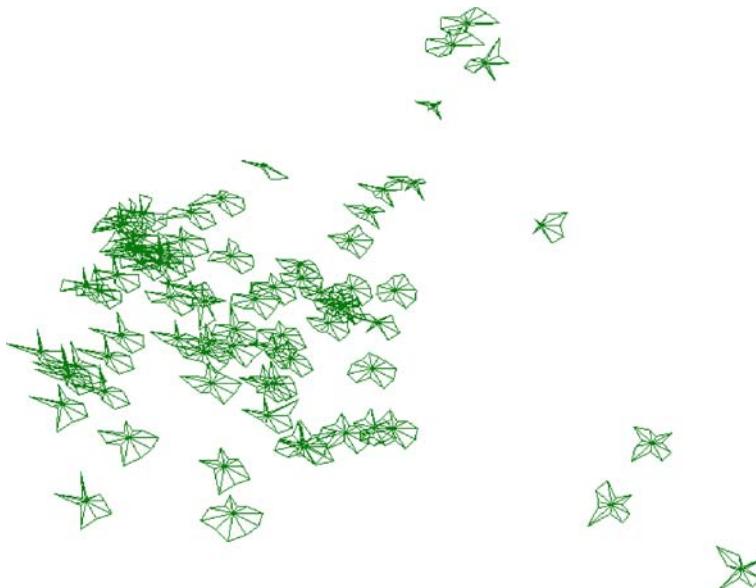


Figure 7.6. Example of positioning glyphs using derived dimensions. In this case, the breakfast cereal data set is shown using the first two principal components to generate the positions. Again, groups of similar shapes are clearly visible

relationship, where on top of the linear ordering is a cycle length, which implies that each glyph is related not only to the adjacent glyphs in the sequence but also the glyphs in the previous and following cycles. Examples of such cyclic placement are shown in Fig. 7.8.

Another type of structure that can be used for positioning is hierarchical or tree-based structures (Ward and Kein, 1997). These may be a fixed attribute of the data (e.g., a computer file system) or computed via, say, a hierarchical clustering algorithm. A wide range of options exist for computing the positions given such a hierarchical structure, as can be seen in the tree-drawing literature (Di Battista et al., 1999). Hierarchically structured glyphs allow easy access both to the raw data as well as aggregation information (Fig. 7.9). Finally, data records might have a network or graph-based structure, such as geospatial data or the web pages associated with an organization. Again, methods from the graph-drawing community can be used to generate the positions for glyphs.

Different structure-driven placement strategies will have different degrees of overlap; a grid layout of ordered data records can assure no overlaps, while tree and graph layouts for dense data sets can result in significant overlap. In cases of overlap, distortion methods are quite common, as structure may be easily preserved and visible even with significant movement of glyphs. Most nonlinear distortion (lens) techniques (Leung and Apperley, 1994) allow the user to view one region of the data space in greater detail than others, shifting the distribution of screen space to provide more

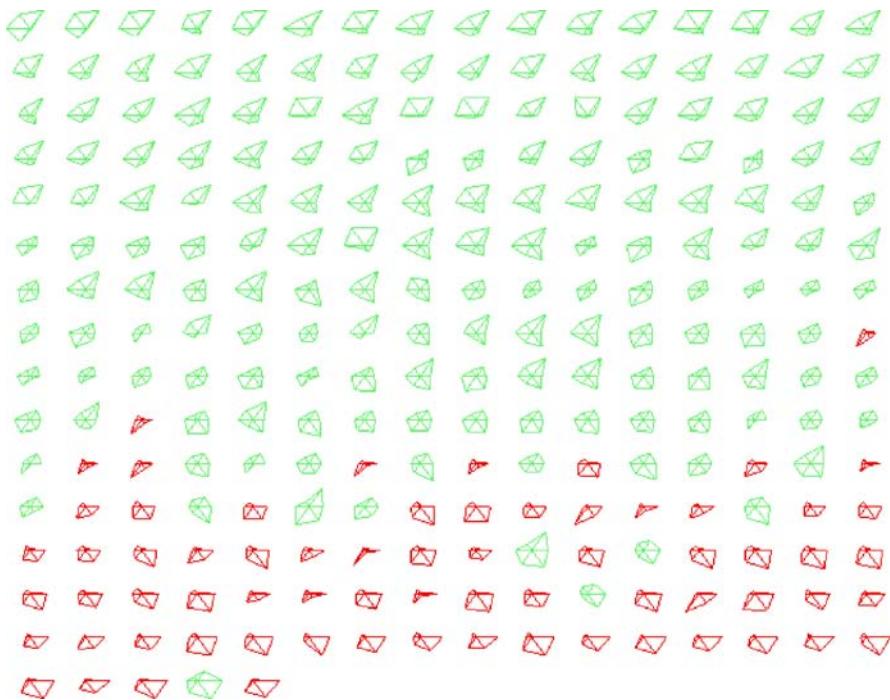


Figure 7.7. Example of ordering glyphs according to one dimension. In this case, the data set is a subset of the cars data, sorted by the MPG variable (*top to bottom* corresponds to low to high MPG). The highlighted (dark) glyphs represent four-cylinder cars. A clear grouping of shapes is visible. A few outliers can be seen near the *bottom* of the figure, which represent six-cylinder cars with good MPG

room for the user's focus region(s). This usually enables users to see subsets of the data without the problem of occlusion. Distortion can also be used to enhance separation of subsets of data into groups. Thus in an order-based layout, gaps between adjacent glyphs can be set proportional to a similarity metric. In a sense, this can be seen as a combination of structure and data-driven methods (Fig. 7.10).

Evaluation

7.8

Evaluation of the effectiveness of glyphs for information presentation and analysis can be performed in a number of different ways. In this section, I describe several such assessment processes, including:

- Evaluation based on ranking of human perceptual abilities for different graphical attributes;
- Evaluation based on the speed and accuracy of users performing specific tasks;
- Evaluation based on ease of detection of data features in the presence of occlusion and clutter; and

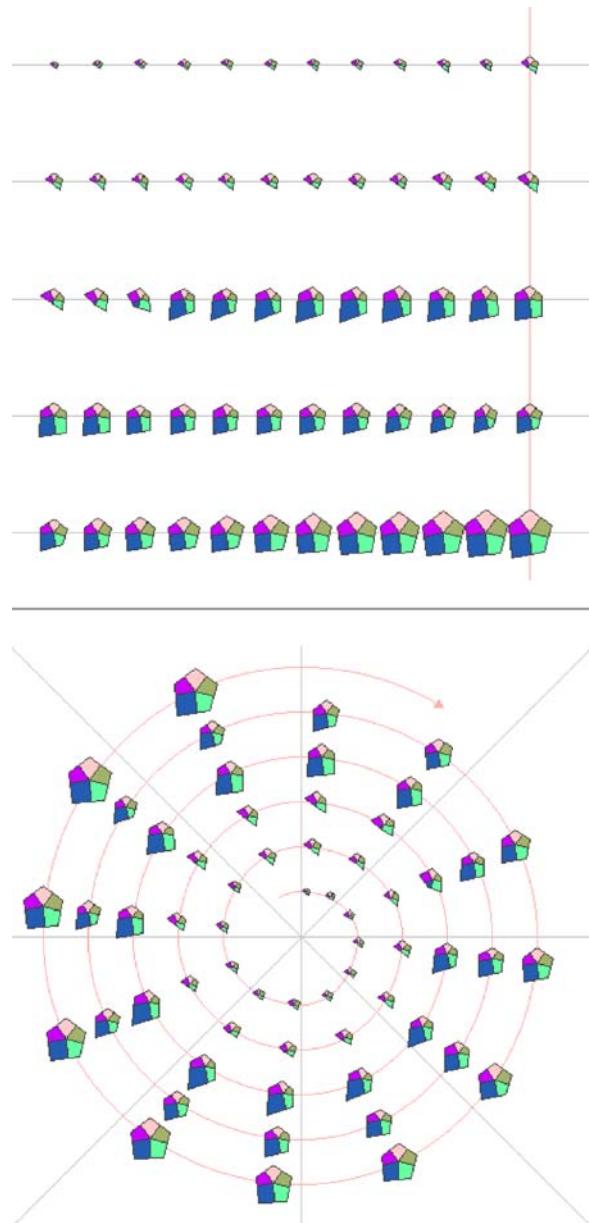


Figure 7.8. Examples of cyclic data glyph layouts, as generated by SpiralGlyphics (Lipchak and Ward, 1997; Ward and Lipchak, 2000). The data consist of five economic indicators over 5 years. In the first layout, each row constitutes a cycle, while in the second, each ring of the spiral is one cycle. While both allow the user to see both intracycle and intercycle variations in the data, the spiral more easily allows comparison of the end of one cycle and the beginning of the next. In addition, space appears to be more effectively utilized in the spiral layout for long cycles

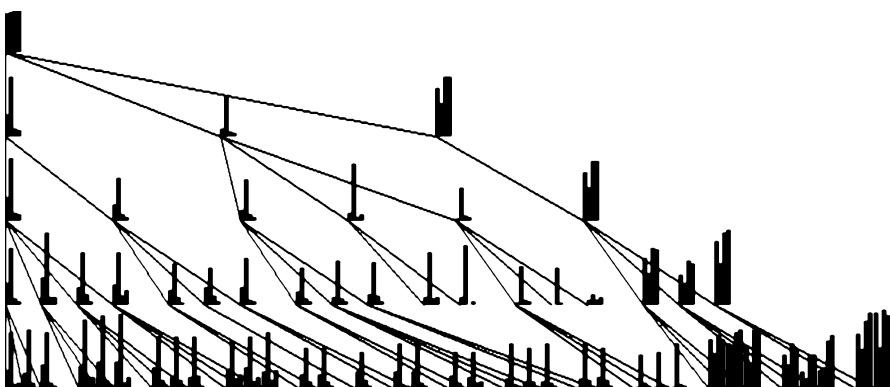


Figure 7.9. Profile glyphs of a hierarchically clustered subset of the Iris data. Nonterminal nodes are computed as the average values of their descendants. The clustering algorithm appears to have done a reasonable job, though a few outliers exist, such as the cluster associated with the fifth node in the third row

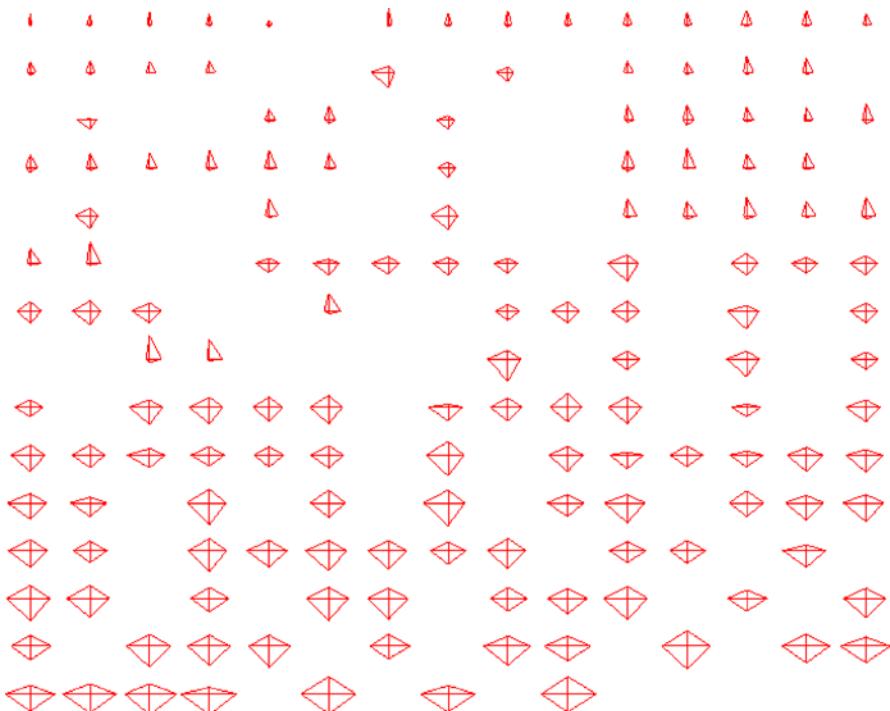


Figure 7.10. Star glyphs of Iris data set, ordered by one dimension and positioned with horizontal spacing proportional to the distance between adjacent data points. This nonoverlapping layout makes it easy to identify both clusters and large gaps in the N-D distance where the values for the ordering dimension are similar to each other

-
- Evaluation based on the scalability of techniques in terms of number of records and dimensions.

A large number of evaluation studies on glyphs and other forms of multivariate data analysis have been carried out over the years. Some of these have been rather ad hoc, or based just on the opinions and observations of the authors, while others have involved detailed and carefully orchestrated user studies.

Cluff et al. (1991) evaluated and categorized 12 methods of multivariate data presentation, including several forms of glyphs. Their evaluation criteria fell into three groups: objectives, information level and dimension capacity, and global criteria. Under the objectives group they considered accuracy, simplicity, clarity, appearance, and design. In the information level and dimensional capacity criterion, visualizations are classified as to what extent they retain the level of information present in the data. At the lowest level (elementary) is the ability to convey the individual data values, while at the intermediate level, relationships among subsets of the data can be seen. The highest level (overall) provides linkages between multiple relationships and allows users to understand the data sufficiently to solve real tasks. The global criteria group includes flexibility, interpretability, visual impact, time to mastery, and computational tractability. For each visualization method, each of these criteria was classified (subjectively by the authors) as 1 = does not sufficiently meet objective, 2 = meets objective satisfactorily, and 3 = meets objective in an excellent manner. While the results of the classifications lack statistical significance due to the small sample size, the categorization of evaluation criteria was far more extensive than in most other studies.

Lee et al. (2003) analyzed the effectiveness of two glyph techniques (Chernoff faces and star glyphs) and two spatial mappings, where each data record was simply represented by a marker whose position was based on similarity to other data records. Binary data was used, and 32 subjects were asked a range of questions regarding relationships between records (both local and global). Results showed that the subjects answered many of the questions more quickly and accurately, and with more confidence, using the spatial mappings. This confirmed the hypothesis of many researchers, which is that glyph interpretation can be quite slow for tasks that involve a significant scanning and comparison. However, questions regarding the values of particular data features could not readily be answered with the spatial mappings. The implication is that, given a known set of questions, it may be possible to assign positions of simple points to facilitate a task. For general tasks, however, a combination involving positioning of glyphs based on data relationships, as suggested in the glyph layout section of this paper, would likely be most effective.

As mentioned earlier, Borg and Staufenbiel (1992) compared snowflake and star glyphs with factorial suns, with the angles of the lines conveying relationships between dimensions. In their experiment, they used the classification of 44 prototypical psychiatric patients across 17 attributes into 4 categories, as determined by 11 experts. Then, using each of the 3 glyph types, they generated drawings of each of the 44 cases. Thirty beginning psychology students were asked to group the drawings into 4 categories based on shape similarity. They then studied the frequency with which drawings of the same category (according to the experts) were grouped together by the

students. The results showed a significantly greater success rate with factorial suns as compared to snowflakes and star glyphs.

Several evaluations of Chernoff faces have been reported since their introduction. One interesting set of experiments was reported in Morris et al. (1999), who focused on studying the effectiveness and preattentiveness of different facial features. Subjects were shown images containing varying numbers of faces, and they were asked to determine if a face with a designated feature existed. The amount of time required to complete each task was measured and analyzed. Not surprisingly, the amount of time needed was proportional to the number of glyphs on the screen. However, the authors determined that it is not likely that preattentive processing was involved, as tests done with short duration, even with small numbers of glyphs, yielded poor results. Their conclusion was that, because glyph analysis with Chernoff faces was being done sequentially, they are unlikely to provide any advantage over other types of multivariate glyphs.

A study that placed Chernoff faces ahead of several other glyph types was reported by Wilkinson (1982). In this study, subjects were asked to sort sets of glyphs from most similar to least similar. The glyphs used were Chernoff faces (Chernoff, 1973), Blobs (Andrews, 1972), castles (Kleiner and Hartigan, 1981), and stars (Siegel et al., 1972). The results were that faces produced results with the best goodness of fit to the real distances, followed by stars, castles, and blobs. The author felt that the memorability of the faces helped users to better perform this type of task.

Summary

7.9

Glyphs are a popular, but insufficiently studied, class of techniques for the visualization of data. In this paper, we've discussed the process and issues of glyph formation and layout, including the identification of problems of bias due to perceptual limitations and dimension ordering. We also presented techniques for evaluating the effectiveness of glyphs as a visualization method and some results obtained from evaluation.

Many avenues exist for future development and application of glyphs for data and information visualization. There is a continuing need for glyph designs that minimize bias while maximizing the accuracy of communicating data values. While the majority of recent designs have been tailored to particular domains and tasks, we believe there is still room for work on general-purpose glyph designs. Scalability is also a big issue, as most glyph methods in use are limited either by the number of data records or data dimensions that can be easily accommodated. Given the growth in the size and dimensionality of common data sets, novel mechanisms are needed to enable users to explore larger and larger amounts of data. Work on aggregation glyphs (Yang et al., 2002) or other multiresolution strategies may be the key to the problem of scale. Finally, rigorous evaluation is essential to help identify the strengths and weaknesses of each proposed glyph in terms of user perception, analysis tasks, and data characteristics. Most efforts at evaluation to date have been ad hoc or very limited in scope.

Acknowledgement. The author would like to thank Jing Yang, Ben Lipchak, and Wei Peng for their help in designing and implementing the software systems used to generate the figures in this paper (XmdvTool and SpiralGlyphics). This work was supported by NSF grants IIS-9732897 and IIS-0119276.

References

- Anderson, E. (1957). A semigraphical method for the analysis of complex problems. *Proc Natl Acad Sci USA*, 13:923–927.
- Andrews, D.F. (1972). Plots of high dimensional data. *Biometrics*, 28:125–136.
- Ankerst, M., Berchtold, S. and Keim, D. (1998). Similarity clustering of dimensions for an enhanced visualization of multidimensional data. In: *Proceedings of 1998 IEEE symposium on information visualization*, pp. 52–60. IEEE Computer Society Press, Los Alamitos, CA.
- Beddow, J. (1990). Shape coding of multidimensional data on a microcomputer display. In: *Proceedings of 1990 IEEE conference on visualization*, pp. 238–246. IEEE Computer Society Press, Los Alamitos, CA.
- Bertin, J. (1981). *Semiology of graphics*. University of Wisconsin Press, Madison, WI.
- Borg, I. and Staufenbiel, T. (1992). Performance of snow flakes, suns, and factorial suns in the graphical representation of multivariate data. *Multivariate Behav Res*, 27:43–45.
- Chernoff, H. (1973). The use of faces to represent points in k-dimensional space graphically. *J Am Stat Assoc*, 68:361–368.
- Chernoff, H. and Rizvi, M.H. (1975). Effect on classification error of random permutations of features in representing multivariate data by faces. *J Am Stat Assoc*, 70:548–554.
- Chuah, M., Eick, S. (1998). Information rich glyphs for software management data. *IEEE Comput Graph Appl*, 18:24–29.
- Cleveland, W. and McGill, R. (1984). Graphical perception: theory, experimentation and application to the development of graphical methods. *J Am Stat Assoc*, 79:531–554.
- Cleveland, W. (1993). *Visualizing Data*. Hobart, Summit, NJ.
- Cluff, E., Burton, R.P. and Barrett, W.A. (1991). A survey and characterization of multidimensional presentation techniques. *J Imag Technol*, 17:142–153.
- Di Battista, G., Eades, P., Tamassia, R. and Tollis, I. (1999). *Graph Drawing: Algorithms for the Visualization of Graphs*. Prentice-Hall, Upper Saddle River, NJ.
- du Toit, S., Steyn, A. and Stumpf, R. (1986). *Graphical Exploratory Data Analysis*. Springer, Berlin Heidelberg New York.
- Ebert, D., Rohrer, R., Shaw, C., Panda, P., Kukla, J. and Roberts, D. (1999). Procedural shape generation for multi-dimensional data visualization. In: *Proceedings of Data Visualization '99*, pp. 3–12. Springer, Berlin Heidelberg New York.
- Friedman, J., Farrell, E., Goldwyn, R., Miller, M. and Sigel, J. (1972). A graphic way of describing changing multivariate patterns. In: *Proceeding of the 6th Interface Symposium on Computer Science and Statistics*, pp. 56-59.

- Friendly, M. and Kwan, E. (2003). Effect ordering for data displays. *Comput Stat Data Anal*, 43:509–539.
- Fuhrmann, A. and Groller, E. (1998). Real-time techniques for 3D flow visualization. In: *Proceedings of 1998 IEEE conference on visualization*, pp. 305–312. IEEE Computer Society Press, Los Alamitos, CA.
- Globus, A., Levit, C. and Lasinski, T. (1991). A tool for visualizing the topology of three-dimensional vector fields. In: *Proceedings of 1991 IEEE conference on visualization*, pp. 33–40. IEEE Computer Society Press, Los Alamitos, CA.
- Gnanadesikan, R. (1977). *Methods of Statistical Data Analysis of Multivariate Observations*. Wiley, New York.
- Hartigan, J. (1975). Printer graphics for clustering. *J Stat Comput Simulat*, 4:187–213.
- Helman, J. and Hesselink, L. (1991). Visualizing vector field topology in fluid flows. *Comput Graph Appl*, 11:36–46.
- Keim, D. and Hermann, A. (1998). The Gridfit algorithm: an efficient and effective approach to visualizing large amounts of spatial data. In: *Proceedings of 1998 IEEE conference on visualization*, pp. 181–186. IEEE Computer Society Press, Los Alamitos, CA.
- Kerlick, G. (1990). Moving iconic objects in scientific visualization. In: *Proceedings of 1990 IEEE conference on visualization*, pp. 124–130. IEEE Computer Society Press, Los Alamitos, CA.
- Klassen, R. and Harrington, S. (1991). Shadowed hedgehogs: a technique for visualizing 2D slices of 3D vector fields. In: *Proceedings of 1991 IEEE conference on visualization*, pp. 148–153. IEEE Computer Society Press, Los Alamitos, CA.
- Kleiner, B. and Hartigan, J. (1981). Representing points in many dimension by trees and castles. *J Am Stat Assoc*, 76:260–269.
- Lee, M.D., Reilly, R.E. and Butavicius, M.A. (2003). An empirical evaluation of Chernoff faces, star glyphs, and spatial visualizations for binary data. In: *Proceedings of 2003 Australasian symposium on information visualisation*. Australian Computer Society, Australia.
- Leung, Y. and Apperley, M. (1994). A review and taxonomy of distortion-oriented presentation techniques. *ACM Trans Comput-Hum Interact*, 1:126–160.
- Levkowitz, H. (1991). Color icons: merging color and texture perception for integrated visualization of multiple parameters. In: *Proceedings of 1991 IEEE conference on visualization*, pp. 164–170. IEEE Computer Society Press, Los Alamitos, CA.
- Lipchak, B. and Ward, M. (1997). Visualization of cyclic multivariate data. In: *Proceedings of 1997 IEEE conference on visualization, Late Breaking Hot Topics*, pp. 61–64. IEEE Computer Society Press, Los Alamitos, CA.
- Mezzich, J.D. (1978). A comparison of graphical representations of multidimensional psychiatric diagnostic data. In: Wang, P. (ed) *Graphical Representation of Multivariate Data*. Academic, New York.
- Morris, C.J., Ebert, D.S. and Rheingans, P. (1999). An experimental analysis of the effectiveness of features in Chernoff faces. In: *Proceedings of SPIE AIPR Workshop: 3D visualization for data exploration and decision making*, 3905:12–17.

- Peng, W., Ward, M. and Rundensteiner, E. (2004). Clutter reduction in multi-dimensional data visualization using dimension reordering. In: *Proceedings of 2004 IEEE symposium on information visualization*, pp. 89–96. IEEE Computer Society Press, Los Alamitos, CA.
- Pickett, R. and Grinstein, G. (1988). Iconographic displays for visualizing multi-dimensional data. In: *Proceedings 1988 IEEE conference on systems, man, and cybernetics*, pp. 164–170.
- Post, F., Walsum, T., Post, F. and Silver, D. (1995). Iconic techniques for feature visualization. In: *Proceedings of 1995 IEEE conference on visualization*, pp. 288–295. IEEE Computer Society Press, Los Alamitos, CA.
- Ribarsky, W., Ayers, E., Eble, J. and Mukherjea, S. (1994). Glyphmaker: creating customized visualizations of complex data. *Computer*, 27:57–64.
- Rohrer, R., Ebert, D. and Sibert, J. (1998). The shape of Shakespeare: visualizing text using implicit surfaces. In: *Proceedings of 1998 IEEE conference on visualization*, pp. 121–129. IEEE Computer Society Press, Los Alamitos, CA.
- Rosario, G.E., Rundensteiner, E.A., Brown, D.C., Ward, M.O. and Huang, S. (2004). Mapping nominal values to numbers for effective visualization. *Inf Visualizat*, 3:80–95.
- Schroeder, W., Volpe, C. and Lorensen, W. (1991). The Stream Polygon: a technique for 3D vector field visualization. In: *Proceedings of 1991 IEEE conference on visualization*, pp. 126–132. IEEE Computer Society Press, Los Alamitos, CA.
- Siegel, J., Farrell, E., Goldwyn, R. and Friedman, H. (1972). The surgical implication of physiologic patterns in myocardial infarction shock. *Surgery*, 72:126–141.
- Ward, M. and Keim, D. (1997). Screen layout methods for multidimensional visualization. In: *Proceedings of 1997 CODATA Euro-American workshop on visualization of information and data*
- Ward, M. and Lipchak, B. (2000). A visualization tool for exploratory analysis of cyclic multivariate data. *Metrika*, 51:27–37.
- Ward, M. (2002). A taxonomy of glyph placement strategies for multidimensional data visualization. *Inf Visualizat*, 1:194–210.
- Wilkinson, L. (1982). An experimental evaluation of multivariate graphical point representations. In: *Proceedings of the conference on human factors in computing systems*, pp 202–209. Association for Computing Machinery, New York.
- Wittenbrink, C., Pang, A. and Lodha, S. (1996). Glyphs for visualizing uncertainty in vector fields. *IEEE Trans Visualizat Comput Graph*, 2:266–279.
- Woodruff, A., Landay, J. and Stonebraker, M. (1998). Constant density visualization of non-uniform distributions of data. In: *Proceedings of 1998 ACM symposium on user interface software and technology*, pp 19–28.
- Yang, J., Ward, M.O. and Rundensteiner, E.A. (2002). Interactive hierarchical displays: a general framework for visualization and exploration of large multivariate data sets. *Comput Graph*, 27:265–283.

Linked Views for Visual Exploration

Adalbert Wilhelm

8.1	<i>Visual Exploration by Linked Views</i>	200
8.2	<i>Theoretical Structures for Linked Views</i>	202
	Linking Sample Populations	204
	Linking Models	205
	Linking Types	208
	Linking Frames	209
8.3	<i>Visualization Techniques for Linked Views</i>	209
	Replacement	209
	Overlaying	210
	Repetition	211
	Special Forms of Linked Highlighting	212
8.4	<i>Software</i>	213
8.5	<i>Conclusion</i>	214

Visual Exploration by Linked Views

The basic problem in visualization still is the physical limitation of the 2-D presentation space of paper and computer screens. There are basically four approaches to addressing this problem and to overcoming the restrictions of two-dimensionality:

1. Create a virtual reality environment or a pseudo-3-D environment by rotation that is capable of portraying higher-dimensional data at least in a 3-D setting.
2. Project high-dimensional data onto a 2-D coordinate system by using a data-reduction method such as principal component analysis, projection pursuit, multidimensional scaling, or correspondence analysis.
3. Use a nonorthogonal coordinate system such as parallel coordinates which is less restricted by the two-dimensionality of paper.
4. Link low-dimensional displays.

The idea of linked views has been around for quite some time in order to escape the limitations of 2-D paper or, as Tufte (1990) puts it, “the 2-D poverty of endless flatlands of paper and computer screen.” Identical plot symbols and colors are a common way to indicate that different displays refer to identical cases. This has been widely used in the development of static displays; see Tufte (1983) and Diaconis and Friedman (1983). In McDonald (1982) this concept of linked graphics was first implemented in a computer program to connect observations from two scatterplots. Still by now, the linking in scatterplots and scatterplot matrices, also known as “scatterplot brushing” as promoted by Becker et al. (1987), is the most prominent case of linked views.

The main advantages of linked views are the easiness of the underlying graphical displays and the speed and flexibility with which different aspects of the data can be portrayed – three features that are essential in the exploratory stage of data analysis. Linking a barchart with a histogram, for example, provides the opportunity to compare not only those groups that are defined by each particular category but also those that originate from uniting similar categories without actually changing the underlying data. Figure 8.1 shows in the background an average shifted histogram for the total population and in the foreground the average shifted histogram for a selected

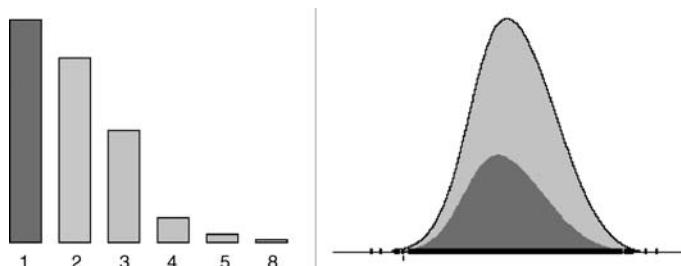


Figure 8.1. Does students reading behavior have an impact on their performance in mathematics? The distribution for those students who do not read at all outside school falls below the overall distribution

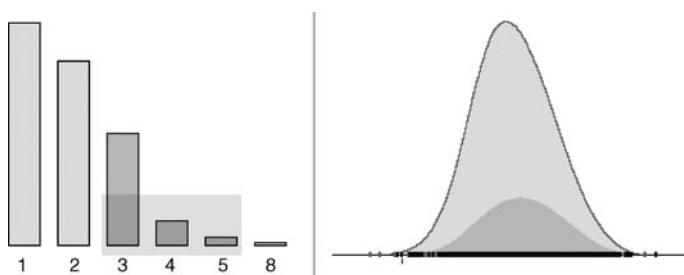


Figure 8.2. Does students reading behavior have an impact on their performance in mathematics? The distribution for those students who read more than 3 h a day is shifted toward the higher scores

subgroup. In Fig. 8.2 three categories are selected at the same time and the resulting conditional distribution is displayed in the histogram. The data used here and in most other examples which follow are a subset of the *Third International Mathematics and Science Study*, an international survey to assess the level of and possible influences on the mathematics and science achievements of 13- and 14-year-old students. The dataset used here consists of a German sample of 5763 students. The dataset is rather typical for surveys in the social sciences and contains a few continuous variables, like the scores on the mathematics and science test, and a large number of categorical variables originating from a questionnaire using a five-point Likert scale.

Another main advantage of linked views is the applicability to complex data structures. The linking concept comes quite naturally with geographically referenced data by connecting the measurements with the geographic location at which the measurements were made. Figure 8.3 shows a map of Bavaria that indicates those counties with a high percentage of forestry.¹

Anselin (1999), Wills (1992) and Roberts (2004) provide a comprehensive discussion of linking in spatial data exploration.

The main application focus of linked displays is in statistical exploration of datasets, in particular, addressing issues such as

- Investigating distributional characteristics,
- Finding unusual or unexpected behavior, and
- Detecting relationships, structure, and patterns.

A particular asset of linked views comes with categorical data and the easy availability of conditional views. Figure 8.4 shows the conditional distribution of the variable *reading for pleasure* for male students. Since the number of students in the group of students who read outside school is in inverse relation to the amount of time spent reading, it is difficult to see whether the males show a particular pattern.

By switching the barchart on the right to a spine plot (Hummel, 1996), we can see the proportions of males falling into each category of reading habits. Here it becomes immediately obvious that males are underrepresented in the medium class. Male stu-

¹ The dataset used here is from the Bavarian Office of Statistics and includes information about land usage for the 96 counties in Bavaria.

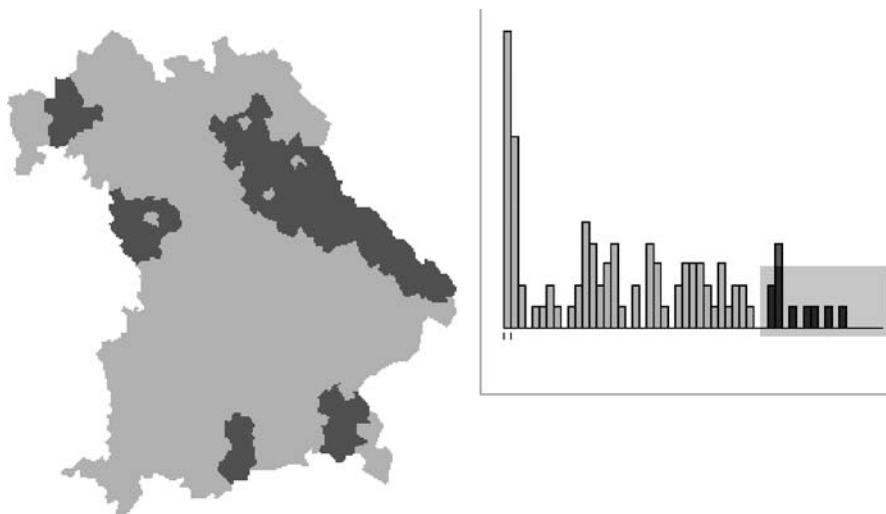


Figure 8.3. Land usage on the county level in Bavaria. Highlighted are those counties with a high percentage of forestry

dents tend to spend less time reading than females, but in the class of reading addicts the share of males equals the share of females.

As we have seen with the previous examples, visual exploration of data requires a flexible and adaptive framework of software ingredients to enable the user to investigate possibilities in a quick and intuitive manner. While flexibility is a virtue on one hand, a stabilizing element is needed on the other hand that makes plots comparable and ensures that the patterns seen in the linked displays are in fact data features and not visual artifacts. The general paradigm of linked views to be described in the following sections provides a systematic approach to flexible and adaptive visualization tools while at the same time offering guidelines and principles for the information exchange between plots and the user. In the following sections, the general paradigm of linked views will be explained pointing to the essential characteristics

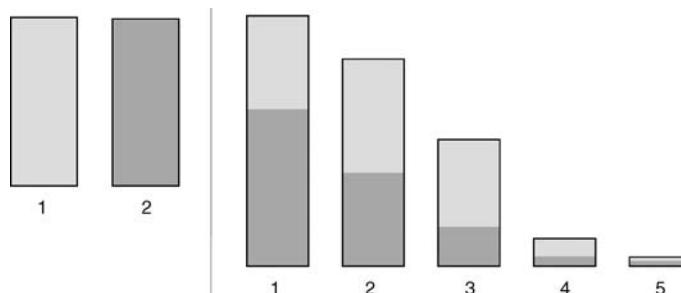


Figure 8.4. Barchart of gender linked to barchart displaying student reading habits. As you move to the *right*, the bars indicate a higher percentage of time spent by students on reading outside school. (1 = female students, 2 = male students)

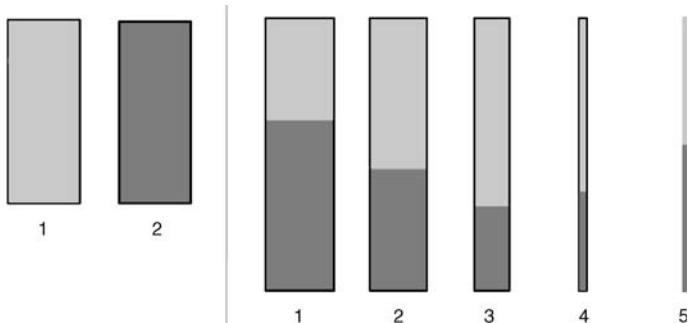


Figure 8.5. Instead of a barchart as in Fig. 8.4 a spine plot is used to portray a different reading behavior. The share of males first decreases but then increases again for the groups of students who spend a lot of their spare time reading

that are needed for linked views so that their concept can be used for a successful exploration of datasets.

Theoretical Structures for Linked Views

8.2

As a general paradigm, linking views means that two or more plots share and exchange information with each other. To achieve the exchange of information, a linking procedure needs to establish a relationship between two or more plots. Once a relation between two plots has been established, the question is which information is shared and how the sharing of information can be realized? To explore the wide range of possibilities of linking schemes and structures, we use the separation of data displays in their components as proposed in Wilhelm (2005). According to this definition, a data analysis display \mathcal{D} consists of a frame \mathcal{F} , a type, and its associated set of graphical elements \mathcal{G} as well as its set of scale representing axes $s_{\mathcal{G}}$, a model \mathcal{X} and its scale $s_{\mathcal{X}}$, and a sample population Ω , i.e., $\mathcal{D} = (\mathcal{F}, (\mathcal{G}, s_{\mathcal{G}}), (\mathcal{X}, s_{\mathcal{X}}), \Omega)$. The pair $((\mathcal{X}, s_{\mathcal{X}}), \Omega)$ is the data part and $(\mathcal{F}, (\mathcal{G}, s_{\mathcal{G}}))$ is the plotting part.

To effectively put the idea of linked views into practice, a communication scheme between the plots has to be established. The (external) linking structure controls the exchange and transfer of information between different plots. In principle, information from all plots may be used and combined; in practice it is reasonable to label one plot “the active plot,” while all other plots are labeled “passive.” This distinction is analogous to the notions “sender” and “receiver” in communication theory. The active plot sends a message to all passive plots, which act accordingly. The definition of data displays and the abstract concept of linking opens the possibility of defining a linking structure as a set of relations among any two components of the two displays. However, only relations between identical layers of the data displays are of practical relevance. The diagram in Fig. 8.6 illustrates the possible linking schemes between the active display $\mathcal{D}_1 = (\Omega_1, \mathcal{X}_1, \mathcal{G}_1, \mathcal{F}_1)$ and the passive display $\mathcal{D}_2 = (\Omega_2, \mathcal{X}_2, \mathcal{G}_2, \mathcal{F}_2)$ under this restriction.

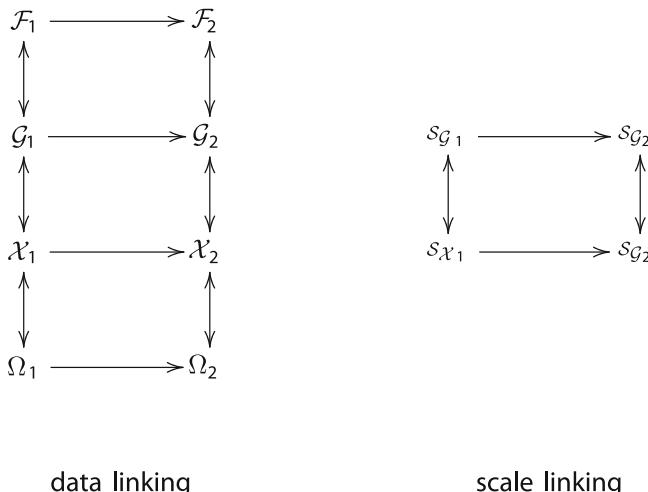


Figure 8.6. A general view on possible linking structures between the active plot \mathcal{D}_1 and the passive plot \mathcal{D}_2 assuming that information sharing is only possible among identical plot layers

Thus, four types of linking structures can be distinguished: *linking frames*, *linking types*, *linking models*, and *linking sample populations*. At the type and at the model level the linking structures can be further differentiated into *data linking* and *scale linking*, the latter being used when scales or scale representing objects are involved in the linking process.

Sharing and exchanging information between two plots can now be resolved in two different ways. The one involves using the direct linking scheme from one layer in display \mathcal{D}_1 to the corresponding layer in display \mathcal{D}_2 . The other is a combined scheme that first propagates the information internally in the active plot to the sample population layer; then the sample population link is used to connect the two displays, and the linked information is then internally propagated in the passive plot to the relevant layers. Hence the most widely used and most important linking structure is sample population linking.

Linking Sample Populations

Sample population linking connects two displays and provides a general platform for all different kinds of user interactions. In general, sample-population-based linking for two data displays \mathcal{D}_1 and \mathcal{D}_2 can be defined as a mapping $m : \Omega_1 \rightarrow \Omega_2$ that maps the elements of the sample population space Ω_1 to some elements of the space Ω_2 . Sample population linking is usually used to create subsets of a dataset and to look at conditional distributions. From this point of view it is intrinsically necessary that the relation between Ω_1 and Ω_2 generates a joint sample space such that the conditional distributions to be investigated are properly defined. Some natural sample population linking structures encompass this property by default.

Identity Linking

The easiest and most common case of sample population linking, which is also known as empirical linking, uses the identity mapping $id : \Omega \rightarrow \Omega$. This linking scheme originates from the goal to visualize the connection between observations that have been taken at the same individual or case. It provides the means to use the natural connection between features observed on the same set of cases. It is intrinsically built into the common data matrices used in statistics in which each row represents one case and each column a variable that has been measured for this case. Identity linking is not necessarily restricted to identical sample populations. Whenever two variables have the same length they can be bound together in a single data matrix and then all software programs will treat the variables as if they have been observed with the same individuals. However, one has to be careful when interpreting such artificially linked variables.

Hierarchical Linking

In practice, databases to be analyzed come from different sources and use different units of analysis. Nevertheless, data bases that are to be analyzed together typically show some connection between the various sample populations. A quite common case is some kind of hierarchy for the various sample populations. This hierarchy can result from different aggregation levels ranging from the micro level of individual persons via different social groups up to the macro level of different societies. Similar situations arise quite common with spatial data which are measured on different geographical grids, like on the local, the regional, the country and the continental level. For such kind of data it is convenient to visualize the connection obtained by the hierarchical aggregation also in the displays. The connection between the sample population spaces has then to be established by a relation $m : \Omega_1 \rightarrow \Omega_2$ for which each element of Ω_1 is mapped to an element of Ω_2 in such a way that some kind of filtration is generated.

Neighborhood or Distance Linking

A special case arises when we work with geographic data where quite often the most important display is a (chorochromatic) map and the focus is on investigating local effects. It is thus quite often desirable to see differences between one location and its various neighbors. So here the linking scheme points also toward the same display and establishes a self-reference to its sample population. A variety of neighborhood definitions are used in spatial data analysis, each one leading to a somewhat different linking relation of the kind $m : \Omega_1 \rightarrow \Omega_2, m(\omega^*) = \{\omega \in \Omega_2 : \text{dist}(\omega^*, \omega) \leq d\}$. Each definition of neighborhood or distance leads to a new variant of linking relation, but the main principles remain the same.

Linking Models

8.2.2

As presented in Wilhelm (2005) models are symbols for variable terms and define the set of observations that shall be represented in the displays. Models are the central

part of the data display definition and describe exactly the amount of information that is to be visualized. The histogram of a quantitative variable, for example, is based on the categorization model. This model is determined by a vector $C = (C_0, \dots, C_c)$ of real values that segments the range of a variable A . For each segment the frequencies of observations that fall into the category are counted and stored. The scale component of the histogram model consists of the categorization vector C , the ordering π_c of the values in C (due to the ordered nature of real values only two orderings make sense, ascending or descending, and the ascending one is the standard used for histogram representations), and the maximum number of counts for any bin. This also implicitly assumes that the vertical axis of the histogram starts at 0 and shows the full range of values from 0 to the maximum number of counts in a category. Notationally, the categorization model can be written as:

$$A \boxplus C = \left([C_0, C_1], (C_1, C_2], \dots, (C_{c-1}, C_c]; \right.$$

$$\text{count}(AC) := \left(\sum_{i:C_0 \leq A_i \leq C_1} \text{count}(A_i), \dots, \sum_{i:C_{c-1} < A_i \leq C_c} \text{count}(A_i) \right) \Bigg)$$

$$s_{\mathcal{X}} = (C, \pi_c, \max(\text{count}(AC))) .$$

A model link for this example can now be established either via the set of observations $A \boxplus C$ or the scale $s_{\mathcal{X}}$. Linking scales for the above example of the categorization operator yields three different cases: linking the categorization vector C , linking the ordering of the categorization values, and linking the maximum count for one bin. Assuming that the categorization operator model is represented by a histogram, the third case basically links the scales used for the vertical axes of the histograms, and the other two link the scales of the horizontal axes, in particular, the bin width and the anchor point.

Linking histogram scales is implemented, for example, in MANET. In Fig. 8.7 two histogram scales are linked. The plot to the right is the active plot that propagates its scale to the one at the left. For the plot at the left also the frame size is adjusted. The

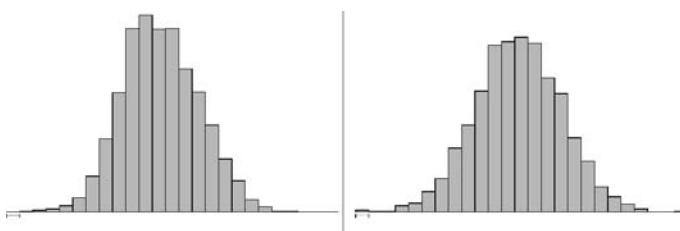


Figure 8.7. Two histograms with linked scales. Both histograms start with the same anchor point, have the same bin width, and use the same scaling for the vertical axes. The plot on the *right* does not fully extend to the *upper border* of the plot because the maximum number of counts for a bin is smaller than the one for the *left plot*. Hence it becomes clear that in this instance, the *left plot* is the active plot.

Otherwise, the largest bins of the *left plot* would exceed the frame boundaries of the plot

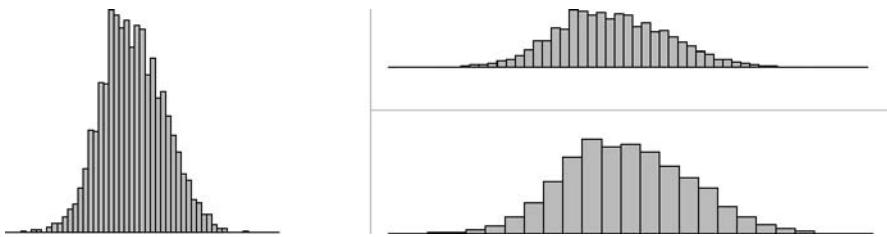


Figure 8.8. Three histograms for the same variable. The two plots on the *right* have the same frame size, but different scales. The plot in the *top right* uses the same scale as the plot on the *left*. The differences in the visual impact are only due to the different uses of frame size or scales

histogram definition in MANET specifies five parameters, the lower and upper limit of the horizontal scale, the bin width, the number of bins, and the maximum height of a bin. Any two of the first four parameters combined with the fifth one are sufficient for a complete specification of a histogram. Via linking all these parameters can be shared. Moreover, the user has the choice of adjusting the frame size as well. For a proper comparison, it is essential to also use the same frame size and not only the same scales. Figure 8.8 shows three histograms for the same variable. The plot on the left is the active one, the plot on the bottom right is not linked at all, and the one on the top right has the same scales as the active one, but a different frame size. These simple examples of linked scales point toward the importance of linking scale information in general. A very widespread use of linking scales is in the form of sliders. Sliders are 1-D graphical representations of model parameters, which the user can change dynamically. Moving the slider yields a change of the underlying model parameter that is then automatically propagated to all plots that display the model. Sliders are a widely used tool to provide a visual representation for dynamic queries (Shneiderman, 1994) to filter and dissect the data into manageable pieces following the visualization mantra: overview first, zoom, and filter, then details on demand (Shneiderman, 1997). Another common application for such sliders is interactively controlling Box-Box transformations or checking various lags for time-series analyses.

The order of the categorization values is of less importance for continuous data displayed in a histogram. This scale component becomes more important for nominal categories that do not have a clearly defined natural ordering. Linking this scale parameter is common and helpful for barcharts and mosaicplots. For the histogram, the categorization vector C actually belongs to both the observation component and the scale component. Using the same categorization vector could thus also be seen as a form of linking the observations of a model. In general, however, it is more appropriate to restrict the concept of linking observations to the variables that are used in the model and not the categorization vector. In this sense, linking models means that plots share the same variables. In Fig. 8.8 all three plots are in principle linked via the model observations because all three plots represent the same variable. In this static form, this kind of linking does not really provide a particular achievement. However, using this form of linking in a prespecified cluster of graphical displays can give a rather complete picture of a dataset. Young et al. (1993) have created a system

in which various views of the same dataset are combined in one window. One plot – typically a scatterplot matrix – controls which variables are displayed in all the other plots included in the window. Clicking a cell in the scatterplot matrix shows marginal views for these variables, for example. The aim of such a system – called empirical linking in Young et al. (1993) – is to create multiple views that act as a single visualization of the data space and provide a complete picture by offering a magnitude of view points and visual aspects. The major benefit lies in the fact that the user can routinize and partially automatize the exploration of a dataset. Once interesting and appropriate plots have been found for one variable, the user can investigate similar or related variables from the same angles by simply exchanging the variables in the control plot.

The model layer of a data display is flexible enough to comprise also more complex models such as regression models, grand tours, and principal components. For these models, a straightforward model link consists of a system of intertwined plots that display the raw observations, the model, and residual information. Young et al. (1993) had introduced such a form in the context of grand tour plots. They designed a spread plot consisting of a rotating plot and two scatterplots presenting residual information for the main components. Similarly, in MANET each biplot is accompanied by two scatterplots of residuals showing the residuals against the first and second principal component; see Hofmann (2001). Changes in the model, as for example initiated by rotating the point cloud in a spread plot, results in an immediate update of the residual information. Young et al. (1993) called this form of linking algebraic linking.

8.2.3 **Linking Types**

The type layer covers most of the visible components in a graphical display and aims at representing the model as well as possible. The distinction between type level and model level basically lies in the fact that, due to the limited plot space and screen resolution, not all models can be visualized without loss of information. The close connection between the two layers also means that congruities at the type level of two displays almost always is a consequence of linked models. For example, it is clear that two histograms that use the same categorization operator also have the same bin widths. A direct link between the type levels of two displays without having a corresponding linkage between the models is rather uncommon. Color and size are attributes of graphical elements that can be linked, in most cases either with or without establishing a corresponding model link. Pie charts, for example, use different colors for the slices to enhance the differentiation between the various categories. These colors are typically assigned individually for each plot and do not carry any intrinsic meaning. Colors could be linked on a type level by assigning identical colors to the first slice in each plot, the second slice, and so on. As long as the ordering of the slices does not reflect information of the model level, the linking will be on the type level only. Alternatively, if the slices are ordered alphabetically or by size, the color also carries some model information and linking via colors could be a realization of

a model link. A similar situation might arise with color maps. Whether these linkages of color constitute a meaningful information exchange depends on the context. At least using an identical color scheme reduces the effect of misinterpretation due to color effects. Linking axis information typically comes down to the fact that all linked displays use the same axis parameters. Also for the axis information it can be said that in most instances the axis parameters are identical to the corresponding scale parameters. For histograms, for example, the limits of the axes are typically depending on the scale specified in the model. Differences between scales and axes usually yield an inefficient usage of the available plot space because part of the plot remains deliberately unused. This can be intended if, for example, observations that are expected to fall in a similar range cover completely different portions of the observation space. If the axes are adjusted to match exactly the scales, one would not notice this feature immediately. If the same axes are used, the different range of the observations becomes clearly visible. A similar situation can also occur with choropleth maps and their corresponding color scheme. Linking type information is usually an essential ingredient for proper comparisons between various plots. The prototypes of incorrect visual representations that can be found in many books (e.g., Wainer, 1997; Monmonier, 1996; Tufte, 1983) are all based on adjusting the axis parameters too closely to the corresponding scale parameters and then interpreting the visual differences in the plots independently of the scales.

Linking Frames

8.2.4

The frame level is the coarsest level of a data display and basically determines the general shape and size of a plot window. Linking the size of frames is not only relevant for a screen-space-saving layout of displays, but it is also one of the prerequisites for a correct comparison of graphical displays, as has been already seen in Fig. 8.8. Using different frame sizes distracts the analyst and can lead to wrong conclusions. Linking of other attributes, such as background color, printing black on white or white on black, is not directly important for correct interpretations and comparisons. However, a common framework to set and change these parameters is convenient, especially for creating different scenarios of data analysis.

Visualization Techniques for Linked Views

8.3

The linking paradigm advocates the sharing of information between displays. This can happen whenever a new display is created by making use of the information that is available in the current displays. A second instance for information sharing is present in interactive environments whenever the user makes changes to a plot while investigating and exploring the data. While the first case is typically easily realized by creating an additional plot window, the second case introduces the question of where the information goes and how the information can be best represented. Roberts et al.

(2000) distinguish three different strategies of exploration: replacement, overlay, and replication.

8.3.1

Replacement

In the replacement mode, old information is typically lost and gets replaced by new information. While this strategy is reasonable for plot parameters, it is rather useless for the subsetting and conditioning approach because the important information on the marginal distributions is lost. It only works fine when we have individual plot symbols for each observation, as in scatterplots for example, where some attributes are changed by the user interaction. But even when replacing plot parameters the user loses the possibility to compare the current plot with previous versions. The user can only compare the current image with a mental copy of the previous image and hence the comparison might get distorted. Especially in the exploratory stage of data analysis for which interactive graphics are designed, it is helpful to keep track of changing scenarios and the different plot versions. A history system that stores the history of plot changes as they are implemented in some geovisualization systems (Roberts, 2004) is very helpful.

8.3.2

Overlaid

In the realm of direct manipulation graphics, overlaying is the typical strategy when looking at conditional distributions in area plots. In Fig. 8.9 a histogram is linked to a barchart. The two classes to the left of the barchart are selected and a histogram for these data points is overlaid on the original plot. The representation of the conditional distribution inherits the plot parameters of the original plot. This eases the comparison between the conditional distribution and the marginal distribution. It also provides a common framework for a comparison of two conditional distributions if the user changes the selection of the conditioning set.

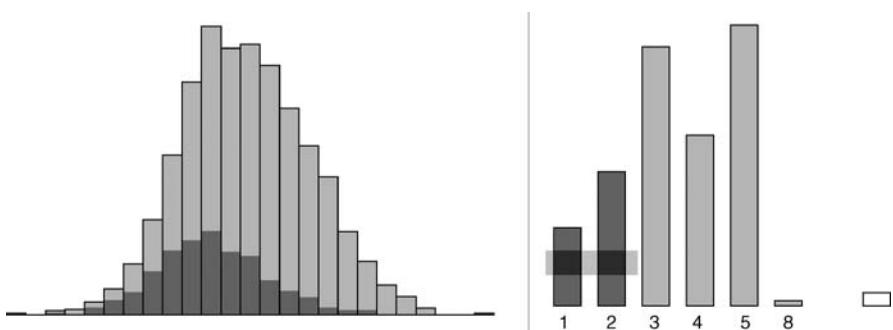


Figure 8.9. Two categories in the barchart are selected. This selection is propagated to the histogram in which a histogram representing the selected subset is overlaid. The overlaid histogram uses the same axis, scale, and plot parameters as the original display and hence establishes comparability between the subgroup and the total sample



Figure 8.10. Overlaying a boxplot for the selected group covers part of the graphical elements of the original box plot. To minimize the confusion and the loss of information, a slightly different style is used for drawing the subgroup boxplot

Overlaying creates two kinds of problems: the one is a basic restriction in the freedom of parameter choice for the selected subset since the plot parameters are inherited from the original plot; the other is the problem of occlusion or overplotting. Part of the original display might become invisible due to the fact that the new plot is overlaid. This can occur in particular when the data representing objects on the type level differ substantially for the subset and the total sample. While this problem might not be present for most area-based displays and while it is rather unimportant for scatterplots, it is essential for more complex plots such as boxplots (Fig. 8.10).

Repetition

8.3.3

Repetition is the third strategy of visualizing linked interactions. Here, the displays are repeated and different views of the same data are available at the same time. The advantage is a rather comprehensive picture of the data; the user has a complete overview on all different representations and can clearly observe the impact of parameter changes and other user interactions. The disadvantage is that the user might get lost in the multitude of slightly changed and adapted views. The repetition strategy requires an easy way to keep track of the various changes and adaptations that have been issued by the user. It also requires an easy and powerful system to arrange the displays on a computer screen. A condensed form of the repetition strategy that works very well for the purpose of subsetting is juxtaposition. This means placing the plot for the selected subgroup not directly on top of the original plot but close to it to the side. Juxtaposition avoids masking important features of the original plot and still allows easy comparison between the two representations. The comparative

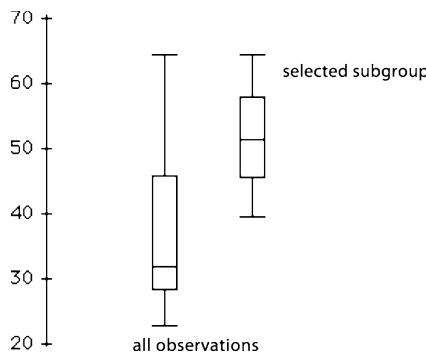


Figure 8.11. Instead of overlaying the plot for the selected subgroup, it is placed next to the original one such that no overlapping occurs

frame of the original plot remains available and the subset is clearly visible as well. Juxtaposition is well known for static plots but has not yet gained acceptance in interactive graphical systems. There seems to be a psychological barrier that prevents the use of juxtaposition because usually it requires rearrangement of the original plot to make space for the additional plot. It also means that each plot has to be fully redrawn after each user interaction, which can be quite demanding depending on the size of the dataset and the complexity of the graphical display. But current computer power should render the interactive creation of juxtaposed plots possible. Juxtaposition also opens the possibility of seeing a full sequence of user interactions and thus allows one to meticulously inspect the flow of subsetting. Juxtaposition also is a worthwhile representation scheme for dynamic animations. The principle of juxtaposition can be straightforwardly extended from graphical displays to statistical model displays. Having calculated a statistical model for the whole sample, we might be interested in particular subsets. Interactively specifying a subset of the sample population should then run the model for the selected subgroup only. The new results could then be juxtaposed to the original ones to allow for easy model comparison.

8.3.4

Special Forms of Linked Highlighting

Different problems occur when the linking scheme is not a simple 1-to-1 linking but a more complex form such as m -to-1 linking as occurs in hierarchical linking. Assume that there are two levels in the hierarchy: the aggregated macro level, a set of counties for example, and the micro level, a set of towns in these counties. Whenever some but not all towns in a county are selected, it would be nice to represent this partial selection of the county by a partial highlighting. If the representation of the macro level is based on regular shapes, the partial highlighting can be done by subdividing this shape into a selected and a nonselected component to indicate the amount of selected objects on the micro level. The more general approach, however, is to use different intensities of the filling color of graphical elements to represent the various selected proportions. This is not only recommended for displays with graphical elements that have a nonrectangular layout, but it is the more general approach that is usually easier to decode. Figure 8.12 refers to the Bavaria dataset of Fig. 8.3 and shows two maps, the left map portraying the micro level of 96 counties, the right map showing the macro level of 7 regions in Bavaria. The selection of some elements of the micro level in the left map is propagated to the right plot and portrayed there. The varying intensities of the filling color reflect the proportion of highlighted counties in a region.

Although the general linking schemes in this paper have been introduced under the restriction of a directed linking process, a few comments shall be made on bidirectional linking, which allows the mutual exchange and sharing of information between plots. Such bidirectional links would be useful when linking plot parameters that govern the general layout and size of a display. In the unidirectional case, one plot inherits the axis limits from the other. These boundaries might be too small for the passive plot and hence lead to a misrepresentation in the linked plot. It would be much better in such instances to change both sets of plot parameters in such a way

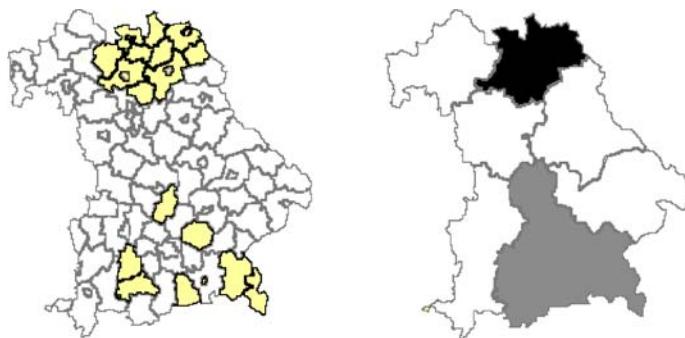


Figure 8.12. Visualization of a hierarchical linking scheme for two maps. The map on the *right* constitutes a coarser grid of geographic locations than the map on the *left*. Selection of some counties in the *left map* are represented by varying intensities in the regions of the *right map*. The darker a region is colored, the more counties in this region are selected in the *left plot*

that both plots represent their model in a meaningful way, e.g., by choosing the joint axis parameters as the minimum and the maximum of the individual limits. A bidirectional link based on the selection of a subset in a plot is implemented in MANET for the trace plot. The linking scheme that is used there is a combination of 1-to- n linking and m -to-1 linking. When a region in the map is selected, then all points in the trace plot are highlighted that depend on the value observed at the selected region. When a point in the trace plot is selected, then all regions that contribute to this value are highlighted. This highlighting action is then returned to the trace plot and all points are highlighted that depend on the currently highlighted regions (Wilhelm, 2005).

Software

8.4

Although linked views are readily available in many statistics research software packages, e.g., LISPSTAT (Tierney, 1990), DATA DESK (Velleman, 2000), MONDRIAN (Theus, 2002), or MANET (Unwin et al., 1996; Hofmann and Theus, 2000), they have not yet been widely included in the major commercial statistics programs. However, some features of linking, usually in a noninteractive realization, can be found in the commercial flagships, like SPSS and SAS. The principles of the linking paradigm laid out in this text have been mostly inspired by DATA DESK and MANET. Lately, interactive features have been added to the R project (Urbanek and Theus, 2003). With respect to software it is also worthwhile mentioning that the increasing size of datasets puts challenges to most of the software in terms of speed. While MANET works pretty fast with datasets of up to 10 000 cases, it slows down significantly for datasets with 80 000 cases. As large datasets might acquire the new standard, implementation might improve. However, for research software this might require a complete reprogramming. DATA DESK is more robust in this respect and works even with datasets of one million cases fairly quickly.

Conclusion

The use of multiple linked views is a well-known concept in interactive statistical graphics. By establishing a close connection between plots that show different aspects of related data, the analyst can explore the data in an easy and flexible way. Linking simple low-dimensional views enables the user to understand structures and patterns of more complex datasets. Multiple linked views are an essential contribution to the field of visual data mining and can provide the required human–computer interaction (Fayyad et al., 1996) to understand the hidden structures and relations. Embedded in efficient software systems, the paradigm of linked views is a powerful tool to explore a broad range of data. The simplicity of the individual plots combined with an intuitive, easy-to-use, and flexible user interface is especially rewarding when using it for consulting experts in the data domains. Applied researchers are familiar with most of the displays used as ingredients in linked systems. Hence a broad audience can easily use and interpret these plots.

Linking procedures become particularly effective when datasets are complex, i.e., they are large (many observations) and/or high-dimensional (many variables), consist of a mixture of categorical and continuous variables, and have a lot of incomplete observations (missing values). Generalization of linking aims to give consistent views of data, consistent not only for each individual point but also for a plot as a whole. To offer consistent comparisons of visual displays, plots should have the same scale and should allow one to compare proportions. The interactive spirit of an analysis offers a way to build in prior knowledge and metadata. Impressive results have been obtained in the exploratory analysis of spatial data; the same can be expected for other areas.

References

- Anselin, L. (1999). Interactive techniques and exploratory spatial data analysis, in P. Longley, M. Goodchild, D. Maguire, and D. Rhind (eds), *Geographical Information Systems: Principles, Techniques, Management and Applications*, Wiley, New York, pp. 251–264.
- Becker, R.A., Cleveland, W.S. and Wilks, A.R. (1987). Dynamic graphics for data analysis, *Statistical Science* 2:355–395. With discussion.
- Diaconis, P.N. and Friedman, J.H. (1983). *M* and *N* plots, in M. Riszi, J. Rustagi and D. Siegmund (eds), *Recent Advances in Statistics: Papers in Honor of Herman Chernoff on His Sixtieth Birthday*, Academic Press, New York, pp. 425–447.
- Fayyad, U., Piatetsky-Shapiro, G. and Smyth, P. (1996). The KDD process for extracting useful knowledge from volumes of data, *Communications of the ACM* 39:27–34.
- Hofmann, H. (2001). *Graphical Tools for the Exploration of Multivariate Categorical Data*, Book on Demand.
- Hofmann, H. and Theus, M. (2000). MANET, <http://stats.math.uni-augsburg.de/manet>.

- Hummel, J. (1996). Linked bar charts: analysing categorical data graphically, *Computational Statistics* 11:36–44.
- McDonald, J.A. (1982). *Interactive graphics for data analysis*, PhD thesis, Stanford University.
- Monmonier, M.S. (1996). *How to lie with maps*, University of Chicago Press.
- Roberts, J.C. (2004). Exploratory visualization with multiple linked views, in J. Dykes, A.M. MacEachren and M.-J. Kraak (eds), *Exploring Geovisualization*, Elsevier, pp. 149–170.
- Roberts, J.C., Knight, R., Gibbins, M. and Patel, N. (2000). Multiple Window Visualization on the Web using VRML and the EAI, in R. Hollands (ed), *Proceedings of the Seventh UK VR-SIG Conference*, pp. 149–157. <http://www.cs.kent.ac.uk/pubs/2000/1123>.
- Shneiderman, B. (1994). Dynamic queries for visual information seeking, *IEEE Software* 11(6):(6)–77.
- Shneiderman, B. (1997). Information Visualization: White Paper, <http://www.cs.umd.edu/hcil/members/bshneiderman/ivwp.html>.
- Theus, M. (2002). Interactive data visualization using Mondrian, *Statistical Computing & Statistical Graphics Newsletter* 13(2):(2)–13.
- Tierney, L. (1990). *LispStat – An Object-oriented Environment*, Wiley, New York.
- Tufte, E.R. (1983). *The Visual Display of Quantitative Information*, Graphics Press, Cheshire, CT.
- Tufte, E.R. (1990). *Envisioning Information*, Graphics Press, Cheshire, CT.
- Unwin, A.R., Hawkins, G., Hofmann, H. and Siegl, B. (1996). Manet – missings are now equally treated, *Journal of Computational and Graphical Statistics* 5:113–122.
- Urbanek, S. and Theus, M. (2003). iPlots – High interaction graphics for R, in K. Hornik, F. Leisch and A. Zeileis (eds), *Proceedings of the 3rd International Workshop on Distributed Statistical Computing*, Vienna.
- Velleman, P. (2000). DataDesk, <http://www.datadesk.com>.
- Wainer, H. (1997). *Visual Revelations*, Springer, New York.
- Wilhelm, A.F.X. (2005). Interactive statistical graphics: The paradigm of linked views, in C. Rao, E. Wegman and J. Solka (eds), *Handbook of Statistics*, Vol. 24, Elsevier, pp. 437–537.
- Wills, G. (1992). *Spatial Data: Exploration and Modelling via Distance-Based and Interactive Graphics Methods*, PhD thesis, Trinity College, Dublin.
- Young, F.W., Faldowski, R.A. and McFarlane, M.M. (1993). Multivariate statistical visualization, in C. Rao (ed), *Handbook of Statistics*, Vol. 9, Elsevier, pp. 959–998.

Linked Data Views

II.9

Graham Wills

9.1	<i>Motivation: Why Use Linked Views?</i>	218
9.2	<i>The Linked Views Paradigm</i>	221
9.3	<i>Brushing Scatterplot Matrices and Other Nonaggregated Views..</i>	224
9.4	<i>Generalizing to Aggregated Views</i>	227
9.5	<i>Distance-based Linking</i>	231
9.6	<i>Linking from Multiple Views</i>	232
9.7	<i>Linking to Domain-specific Views</i>	235
9.8	<i>Summary</i>	238
9.9	<i>Data Used in This Chapter</i>	239

The linked views paradigm is a method of taking multiple simple views of data and allowing interactions with one to modify the display of data in all the linked views. A simple example is that selecting a data case in one view shows that data case highlighted in all other views. In this section we define the underlying methodology and show how it has been applied historically and how it can be extended to provide enhanced power. In particular we focus on displays of aggregated data and linking domain-specific views such as graph layouts and maps to statistical views.

9.1

Motivation: Why Use Linked Views?

A “data view” can be thought of as anything that gives the user a way of examining data so as to gain insight and understanding. A data view is usually thought of as a barchart, scatterplot, or other traditional statistic graphic, but we use the term more generally, including “views” such as a display of the results of a regression analysis, a neural net prediction, or a set of descriptive statistics. A plot of geographic information, such as a map of a country, is a data view. A node and edge graph displaying interrelationships between relatives (more commonly known as a “family tree”) is a data view. A printout of the R^2 value from a regression is a data view, albeit a very simple one. Views of data are also known as graphs, charts, diagrams, plots, and visualizations, but each of those terms has connotations that can restrict how we think of a linked view. Thus, in this chapter, we use the simple term “data view” – something that allows us to view data.

A linked data view is a data view that communicates with another view. If a modification is made to one of the views, the other view will change its appearance in reaction to the modification. A simple example of linked data views is a scroll bar in a text editor that is linked to the text view. The scroll bar has a “thumb” that shows which part of the document is being displayed. When the user modifies the scroll bar by dragging the thumb around, the text view updates to show that portion of the document. This example is presented first to highlight the ubiquity of the linked views approach; the linked views paradigm is used in standard user interfaces (Apple Computer, 1992; Microsoft, 1999) and in game software (Maxis, 1985) as well in more specifically data-analytic software.

Figure 9.1 shows a more directly data-analytic version of linked views. The dataset used is described in the appendix, which gives details of the data used to generate each figure in this chapter. The data are taken from an archive of baseball statistics collected 1871 and 2004. In this figure we are interested in comparing players’ salaries to their performance and so create a scatterplot showing the relationship between salary and batting average (the single most commonly used measure of a player’s batting ability). We create a histogram of the year for this dataset (which incidentally shows us that salary data only became available starting in 1985) and then select the histogram bars

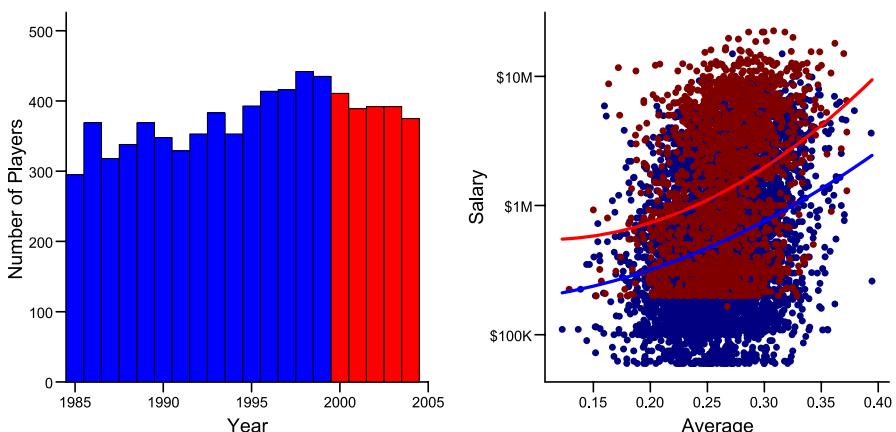


Figure 9.1. Simple linking between a barchart and a scatterplot. The *left view* is a histogram of the number of players in baseball by year, and the *right view* is a scatterplot of players' salaries (on a log scale) against their batting average. A *quadratic curve* has been imposed on the salary data to show level differences between the selected and unselected groups; it is not intended to be a good fit to the data

for years 2000 through 2004. In both the histogram and the scatterplot, those selected data elements are shown in *black* and the unselected ones in *light gray*.

The linking between the views is achieved by allowing the user to select part of one view, in this example using a rectangular selection device, which has the effect of selecting the graphic elements that intersect that rectangle. In Fig. 9.1 above, the linking shows us that there is a level effect for years on salaries, but there is no evidence that it affects the basic relationship between batting average and salary.

A fundamental question for any visualization technique that should always be asked is: “Why should I use this?” or “Why should I need to link views together – what benefit do I get from this?” From the analyst’s point of view, if the analyst has created a view of the data and seen something of interest (or, as can also often occur, not seen something of interest when they expected to), then they will want to explore further. They will want to know, for example, if data form clusters under a particular projection of the grand tour or if there is a change in the relationship between salary and years playing baseball when the latter is above a given threshold? When they see something interesting, they want to explain it, usually by considering other data views or by including additional variables. With some types of view, it is not hard to add in variables and see if those variables can explain the feature, or indeed if they have any effect whatsoever. In a regression analysis, you can add a variable to the set of explanatory variables (taking due care with respect to multicollinearity and other confounding factors). If a histogram of X shows something of interest, you can “add” a variable Y to it by making a scatterplot of X against Y . If you want to explain something in a scatterplot, then it is possible to turn it into a rotating point cloud in 3-D. Using projection pursuit or grand tour techniques, you can go to still higher dimensions.

Despite the undoubtedly utility of this approach, it does present some problems that prevent it from being a complete solution. The main ones are:

- As plots become increasingly complex, they become harder to interpret. Few people have problems with 1-D plots. Scatterplots, tables, and grouped boxplots or other displays involving two dimensions are easily learnable. But the necessity of spinning and navigating a 3-D point cloud or understanding the contributions to a multivariate projection make views that contain many variables intrinsically less intuitive.
- It is harder for monolithic data views to accommodate differences in the basic types of data. High-dimensional projection techniques assume the variables are numeric, as do techniques that display multivariate glyphs and, to a large extent, parallel-axis techniques. Given a table of two categorical variables, adding a numeric variable requires changing to a quite different type of view, such as a trellis display.
- Data that are of a type specific to a particular domain can be impossible to add directly. Exploring relationships in multivariate data collected at geographical locations, on nodes of a graph, or on parts of a text document is very hard because of the difficulty of building views that correlate the statistical element and the structural element of the data. Often, two completely different packages are used for the analysis, with results from one package mangled to fit the input form of the other package – a frustrating situation to be in.

The linked views paradigm can be used to overcome these problems. The idea is simple; instead of creating one complex view, create several simpler views and link them together so that when the user interacts with one view the other views will update and show the results of such an interaction. This allows the user to use views that require less interpretation and views that are directly aimed at particular combinations of data. It also allows the easy integration of domain-specific views; views of networks or maps can easily be linked to more general-purpose views.

It should not be argued that linked data views are a uniformly superior method to that of monolithic complex views mentioned above. That is not the case, as there are examples where a single multivariate technique is necessary to see a given feature, and multiple simpler views simply won't do. It is also generally harder to present the results of an interactive view exploration to another person than it is to present the results if displayed as a single view. Having said that, for many problems, especially those where conditional distributions are of interest, the linked data views technique works extremely effectively.

In Fig. 9.2, we have changed the variable being displayed in the histogram to be the number of years in the league. (0 indicates a rookie, 1 indicates one previous year of experience, etc.). The shape of the histogram fits our intuition by being close to a Poisson distribution. We select those players with 5 years of experience or greater in the league and see that not only do they have a higher salary on average, but the relationship between batting average and $\log(\text{salary})$ is much closer to linear. For the younger players, a reasonable case might be made that performance has no strong effect on pay unless the batting average of a player is itself better than average.

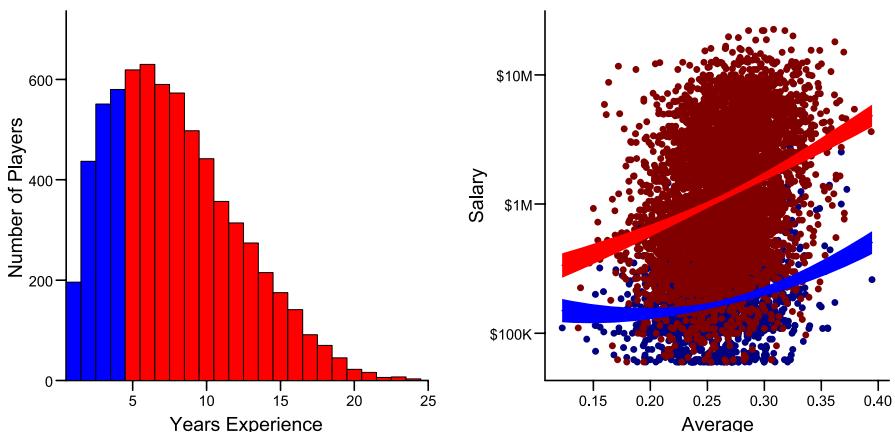


Figure 9.2. Changing variables in one view. The *right view* is similar to Fig. 9.1, with the addition of a 99.9 % confidence interval for the mean fitted value. The *left view* has been left unchanged in definition, except that a different base variable has been used; the histogram is now of the number of years a player has been in the league

The Linked Views Paradigm

9.2

In Sect. 9.1, we showed examples of the linked views paradigm. In this section we will define it more exactly and detail how a linked views paradigm may be implemented.

To implement a linked views system, the fundamental need is an interactive environment. Linking is an interaction mechanism and therefore requires a computer or other system for interaction with graphical representations of data. Given such an environment, a linked views environment needs to have more than one view, and the views available to the user must satisfy the following conditions:

1. At least one view must be capable of detecting user interaction and translating that user interaction into a measure of the degree of interest that user has in the data being displayed in that view. The degree of interest must be able to distinguish between subsets of the data based on the user's interaction.
2. A mechanism is needed to propagate the degree of interest measures created in view (1) to other views in the system.
3. At least one view that is not the view described in (1) should be capable of responding to the measure of interest propagated to it under (2). The response should be to change its visual appearance so as to show the degree of interest measures as they relate to the data being displayed by this view.

The concept of “degree of interest” is worth considering in more detail. It is intended to measure the degree to which the user finds any subset of data interesting. By selecting the bars corresponding to five or more years in the league in Fig. 9.2, the user indicates which portion of the data they are interested in, namely rows where the number of years in the league is 5 or more. The degree-of-interest measure should

return some value in a known range for every possible subset of the data; this is necessary when displaying results in aggregated views. An *aggregated view* is a view where a single graphic item represents multiple cases of data, as is the case for a histogram or barchart where a single bar summarizes multiple data rows. In contrast, an *unaggregated view* is one where each row of the data gets a separate visual representation, as is the case for scatterplots. In this section we use the terms *data case* and *data row* interchangeably, defined as a single observation of data, typically encoded as a row in a data table. A *graphic item* is slightly harder to define but should be thought of as a visually distinct representation that is perceived as a distinguishable unit. A bar in a barchart, a glyph in a scatterplot, and a 3-D surface are all graphic items. A segment of a path is not; the path is a perceptual whole and is thought of and manipulated as a single entity.

In practice, a simplified form of degree of interest can be used where each row of data (each data “case”) is given a degree of interest value, and the measure of a subset is defined as a summary function of the degrees of interest of the cases comprising the subset. Throughout this chapter, we use a degree of interest for the cases in the range $[0, 1]$, where “0” corresponds to “no interest whatsoever” and “1” corresponds to “maximal interest.” For a subset, we will use the mean value summary to summarize the values in the subset. This remains in the range $[0, 1]$ and has the useful property that subsets of different sizes that have the same distribution of measures of interest on their cases will have similar measures of interest. However, it should be noted that other summaries can be useful for certain tasks. For example, the maximum summary function will highlight subsets that have any selected items in them. This will be very useful for spotting outliers, as typically outliers are small in number, so using the mean summary when outliers are the source of a degree-of-interest measure will result in most subsets having zero or near-zero measure of interest, whereas the maximum summary statistic will show immediately any subset containing one or more outliers.

An additional simplification we can make is to state that any view that defines a degree-of-interest value for each case must define it as either zero or one. In other words, the process of interacting with a data view to perform linking will result in a split into selected cases (1s) and unselected cases (0s). This technique is by far the most commonly implemented technique, but there are cases where the more general version is useful. In Sect. 9.5, we explore distance-based linking, and in Sect. 9.6 we show examples of how multiple views can create nonbinary degrees of interest for cases.

For requirement (1) above, we need a means for the user to indicate what is of interest to them. There are multiple means of doing so, the most common of which are:

Brushing. In the brushing paradigm, the user has a fixed shape (typically a rectangle or circle) that they drag over a view of data. As this brush moves over graphic items in the view, it paints them. This is defined as setting the degree of interest to 1 for those items and then typically using colour to code the degree of interest in all linked views. In Sect. 9.3 we give more details on scatterplot brushing, which was one of the earliest implementations of linked views.

Rectangle or rubber-band selection. For this selection mechanism, the user clicks and drags with the mouse to define a rectangle (or other shape, such as a circle) and, on releasing the mouse, the graphic items intersecting the shape are considered selected and their associated cases' degree of interest set to 1.

Lassoing. For this mechanism, the user clicks and drags to create a polygon selection shape, either by clicking and dragging to define a shape or by successive clicks to define a polygon. When completed, either by releasing the mouse or by completing the polygon, the graphic items intersecting the lasso/polygon are considered selected and their associated cases' degree of interest set to 1.

To fulfill requirement (3), a view must be capable of displaying the base data for the view together with a representation of the degree of interest. If we think of the degree of interest as a variable, then we can imagine it being used in the same way as any other variable. Using the language of the Grammar of Graphics (Wilkinson, 1999), we can use a variable such as:

- A *positional* variable, used to define the geometry of the view;
- An *aesthetic* variable, used to modify the appearance of a graphic element;
- A *faceting* variable, used to define a paneling of the view.

In Fig. 9.3 below, barcharts showing the three basic methods for displaying a degree of interest are displayed. Note that for the 3-D barchart, the degree of interest could easily have been a continuous variable; the locations along the z-axis would have been well defined. For the aesthetic version, we split each bar in two to show the relative proportion of selected and non-selected data. This requires the degree of interest to be at worst a finite number of

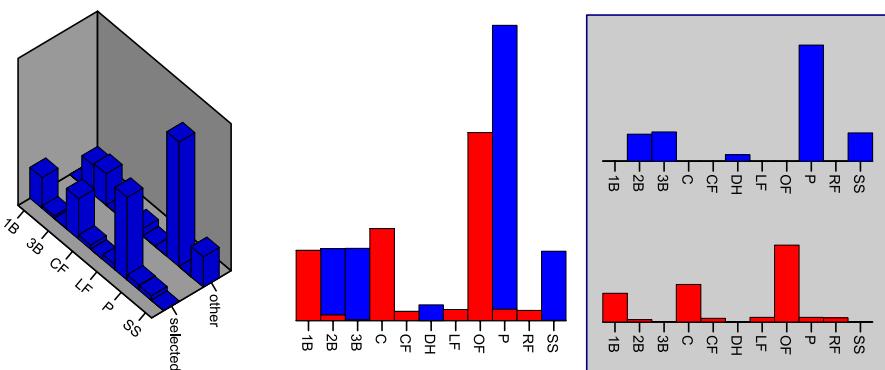


Figure 9.3. Three basic methods for displaying a degree of interest. The basic chart is a barchart of counts of players at different fielding positions. The *left view* adds a “z” position variable to the bar chart, splitting the dataset along the z-axis. The *center view* sets the *brightness* of bars by the degree of interest, with the “interesting” bars shown in *black*. The *right view* facets (*panels*) views by the degree of interest (although there are two facets, we term this a single “view”). The data show players’ fielding positions in baseball, and we have selected those players with more putouts than assists. Clearly this is highly correlated with the fielding position

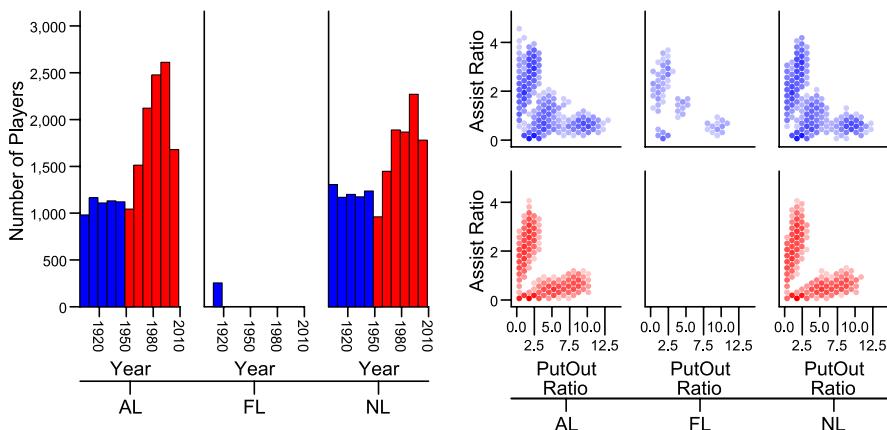


Figure 9.4. Faceting and degree of interest. The *left view* shows players since 1900, faceted by the league in which they played. Note the brief existence of the Federal League (FL: 1914–1915). On the *right* a binned scatterplot of the putout-to-assist ratio is shown. After selecting players playing post-1945 in the *left panel*, the *left view* shows the interesting cases in *black*, whereas the *right view* uses degree of interest to define the vertical paneling (“interesting” cases are in the *lower row*) and uses *brightness* to show the counts of points in the binned scatterplot

categories and works best with binary variables. Systems for displaying aggregated data elements are discussed in Sect. 9.3. We could have used a summary function instead to display the mean degree of interest, mapping it to a *brightness* scale. Section 9.9 gives details on the dataset being displayed, with notes on the meanings of the variables used.

The rightmost view in Fig. 9.3 splits the data into two panels, one for the selected and one for the unselected subsets. This faceting should not be considered a special functionality but should fit into existing faceting schemes, so that, for example, a trellis view would simply incorporate the degree of interest as another variable within the tablelike structure. As a further example, Fig. 9.4 shows how a binary selection status is incorporated into the faceting structure. Baseball has had many different leagues over its history, which are separate pools of teams; each team is a member of one league only in any year and plays almost exclusively members of its league. Note that post-1945, the putout-assist scatterplot resolves itself into two clusters, whereas pre-1945 there is good evidence for at least another cluster. We investigate that more closely later.

Brushing Scatterplot Matrices and Other Nonaggregated Views

One of the earliest linked views works to achieve wide attention was the scatterplot brushing technique of Becker, Cleveland, and Wilks (Becker et al. 1987). By arranging

scatterplots of n variables in a table so that all the $n(n-1)$ ordered combinations of axes are present, the eye can quickly scan a row or column and see how a given variable depends on every other variable. This useful arrangement technique is enhanced by the use of a brush as described in Sect. 9.2. As in typical later scatterplot brushing tools, the data points brushed over are painted in a different colour, both in the panel in which the brush is active and in all other panels of the matrix. In our terminology, the brush is the mechanism that creates the degree of interest “variable” that links the scatterplot data views.

One of the reasons this technique is effective is that in each linked view, there is a one-to-one correspondence between cases of the data matrix and graphical representations of these cases, so that in each scatterplot we have complete freedom as to what colour or glyph to use to represent this data item. Linking scatterplots do not require considerations of aggregation; each data row maps directly to a single glyph in the scatterplot. This simplicity can be seen in Figs. 9.1 and 9.2, where we selected bars and linked to the scatterplot; if we had reversed the direction of interaction, we would have been left with bars that were partially selected. Deciding how to display such views requires some thought and will be discussed in Sect. 9.4.

Even when restricted to data views that display distinct graphical elements for each case, linking is a powerful tool. An example of a successful tool in this area is XGobi (Swayne et al. 1991). XGobi is an X-Windows-based tool that presents the user with several glyph-based views (dotplots, scatterplots, rotating plots, grand tour, and projection pursuit tours) and uses brushing to link the views along the above lines. The latest incarnation of this software is GGobi (Swayne et al. 2003).

Scatterplots and dotplots are the most obvious examples of unaggregated views. Raw tables of data can also be considered unaggregated. One useful technique is to show a table of only those data that have been selected. This is often termed a “drill-down” view but is actually a simple example of linked views. Selection in one view leads to a second view where the “visibility” aesthetic is used to code the degree of interest – only the selected rows are shown.

Parallel coordinates views are a relatively novel form of view introduced by Inselberg (1985). They work well only for relatively small numbers of cases as they show an n -dimensional point as a line in 2-D, taking up a lot of display space for each row. However, within this limitation, they are an ideal candidate for linked views, as they are unaggregated and encourage the user to spot differences between the selected and unselected lines.

Figure 9.5 below shows data for players in the 2004 season. Even restricting ourselves to a few thousand players makes this chart hard to read, but we can see the higher values on each scale are in the selected group and the ones at the bottom are in the unselected group. For this many lines it is hard to distinguish them either by colour or by dashing style and the result is not illuminating. In Sect. 9.7 we will show more compelling uses for parallel coordinates for linking to maps and hierarchical clustering plots.

One interaction feature that is important for brushing is that of the *brush mode* or *paint mode*. In transient mode, where the user drags the brush over the plot, when the brush moves away from an item, the item reverts to unselected state. In additive

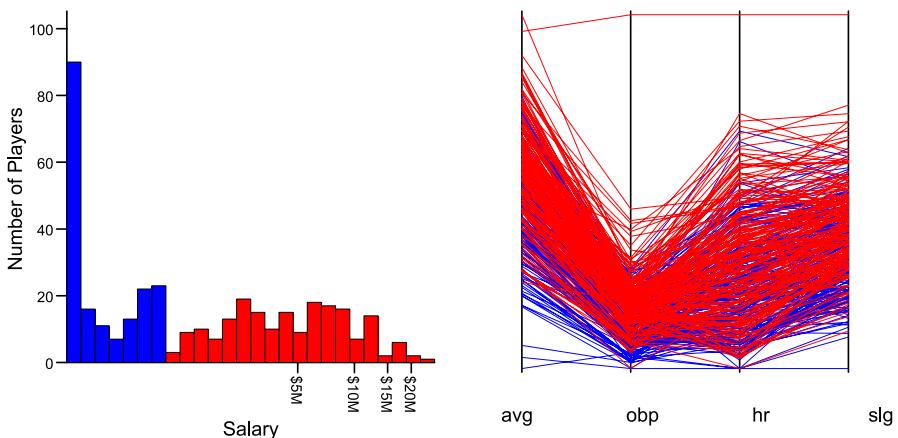


Figure 9.5. Faceting and parallel coordinates. For the 2004 season, the *left view* shows a histogram of salary on a log scale. Players earning over \$1,000,000 were selected and are shown in the linked *parallel coordinates plot* as dashed line. The parallel coordinates view shows four major measures of performance: batting average, on-base percentage, home-run rate, and slugging average

mode, the brush leaves the item in a selected “painted” state. In subtractive mode, the brush is used to change state from selected to unselected, “unpainting” the items. These modes are examples of the general case, in which we have two possible initial states (selected and unselected) and a brush or other selection mechanism states whether or not an item was indicated by the user. For each of the 4 combinations of states there are two possible outputs (selected or unselected), and so we have 24 different possible modes. However, some of these modes, such as the mode that simply selects everything, or the mode that ignores the user’s input, are of little use. Wills (2000) argues that the useful modes are:

Replace: the original state is ignored; the selection defines the output exactly. This is equivalent to the transient mode.

Toggle: the new state is an exclusive OR of the original state with the selection status. This is what happens when we control-click on files in the Windows operating system, for example.

Add: the new state is an inclusive OR of the original state and the selection status. This is the additive mode for brushing scatterplot matrices.

Subtract: the new state is an AND of the original state with the negation of the selection status. This is the subtractive mode for brushing scatterplot matrices.

Intersect: the new state is an AND of the original state with the selection status. This mode is not common but has a strong advantage for data analysis in that it allows repeated conditioning in an interactive environment. In Sect. 9.6 we use it for multiple linked views.

Generalizing to Aggregated Views

The unaggregated approach runs into problems with more than small amounts of data. If you have tens of thousands of points, often you want to look at views that aggregate the data, such as barcharts, histograms, and frequency tables. In this chapter, we use a reasonably sized dataset to examine linking; there are about 16 000 players represented over a total of ca. 86 000 player seasons. A quick review of Figs. 9.1 through 9.5 should convince the reader that the summary views are easier to use than the unaggregated views for these numbers. Therefore, in this section, we consider techniques for linking aggregated views.

One of the first commonly available tools to feature linking was the statistical analysis package Data Desk (Velleman, 1988). It was originally built as a teaching tool but is now a full-featured statistical package that has linked views designed in at the core. Brushing works with aggregated views such as barcharts and histograms as well as within unaggregated views, and the outputs of analyses such as regression and correlation analysis can be visualized and are linked to the other views. If a user spots some unusual cases in a view of residuals from an analysis, they can brush those points, see if there is anything that might explain it in other variables, modify the model, and instantly see the residual view update to reflect the new model. By incorporating model results as linkable views, Data Desk provides a rich set of linking paradigms. In this section we consider simply the way it performs linking to aggregated views. Figure 9.6 shows an example of such linking. This figure is similar to Fig. 9.1, except that the linking is being performed in the reverse direction – from the scatterplot to the barchart.

As in Fig. 9.3, the degree of interest for a bar has been represented by dividing the bar into two sections; one for the selected subset and one for the nonselected subset. An alternative method is used by LispStat (Tierney, 1990), in which each data item is assigned its own place in the bar and that section of the bar has an appropriate *brightness*. This solution is very close to the “one-to-one” relationship method in the previous section, as under this display system each bar is really a set of stacked rect-

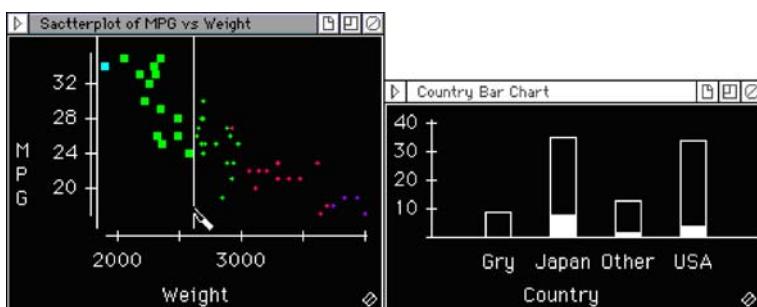


Figure 9.6. Linked views in Data Desk. Points selected in a scatterplot of miles per gallon vs. weight are highlighted in the country barchart. Selecting the points in the low end of the weight scale shows which country makes the lightest cars

angles, one for each case. Both drawing and brushing over the bars is handled as if the bars were a collection of separate little boxes.

One of the more powerful novel features of LispStat is its implementation in Lisp – as an interpreted language, the user is free to write any function that can link views together, and indeed can write any desired data view. If you want to interactively colour items in a scatterplot based on their distance from the center of the brush (as we do in Sect. 9.5), it is an easy job for Lisp programmers.

So for bars one technique is to stack sections of the bar on top of each other – either grouping all the selected items together or leaving them in a fixed location on the bar. Then we can use an aesthetic (typically colour or pattern) to colour the separate sections. Another technique was alluded to in Sect. 1; instead of dividing a bar up, we use a summary function to create a single value for the bar that summarizes the cases it contains and then map that to an aesthetic. This approach is more general in that it can handle numeric degrees of interest beyond the 0–1 binary selection needed for stacked bars, and it is also generalizable to graphical objects that cannot be as conveniently split into components, such as lines and paths. Both techniques are valuable and should be considered for use in any set of linked views. Figure 9.7 shows both techniques in action.

This figure combines five different variables for over 80 000 cases – aggregation is clearly needed and a monolithic view that would adequately show the data is not obvious. In the bottom left panel we show a hexagonal binning of the data and represent each bin with a circle. The mean summary statistic has been applied to the degree of interest for the cases in the bin, each of which is either 0 or 1. This effectively gives us the fraction of the elements that are selected. Figure 9.8 below shows two alternative ways of displaying the information, using the size aesthetic and by dividing up the glyph. In both cases the mean degree of interest maps to a circle area, not radius, facilitating a better mental model of the distribution.

Although Fig. 9.7 is easier to interpret for most people, Fig. 9.8 might be preferred in certain circumstances. For example, if we wished to use hue to encode another characteristic, the mind has a tendency to combine hue and *brightness* into a gestalt colour, which would make a colour aesthetic added to the scatterplot of Fig. 9.7 a more confusing view than if we added colour to either view of Fig. 9.8. Conversely, we might want to show the distribution of heights and weights rather than just the fraction of designated hitters in each bin. To do that we might use a size aesthetic to show the count of players in each bin and apply that aesthetic to the scatterplot of Fig. 9.7.

From Fig. 9.7 we can see that, although designated hitters tend to be of average height, with relatively few very short or tall players, they are well over average weight. This makes intuitive sense if we consider their role. They are players who do not field or pitch. Their only role is to bat when the pitcher would otherwise be batting, replacing him in the batting order. The designated hitter position was introduced because the gap between the pitcher's ability to hit the ball and the rest of his team's ability is generally large. Therefore, rather than have a known poor performance on every rotation through the team, one league, the American League (AL), decided to introduce this position. It is a “power hitter” position. The designated hitter has

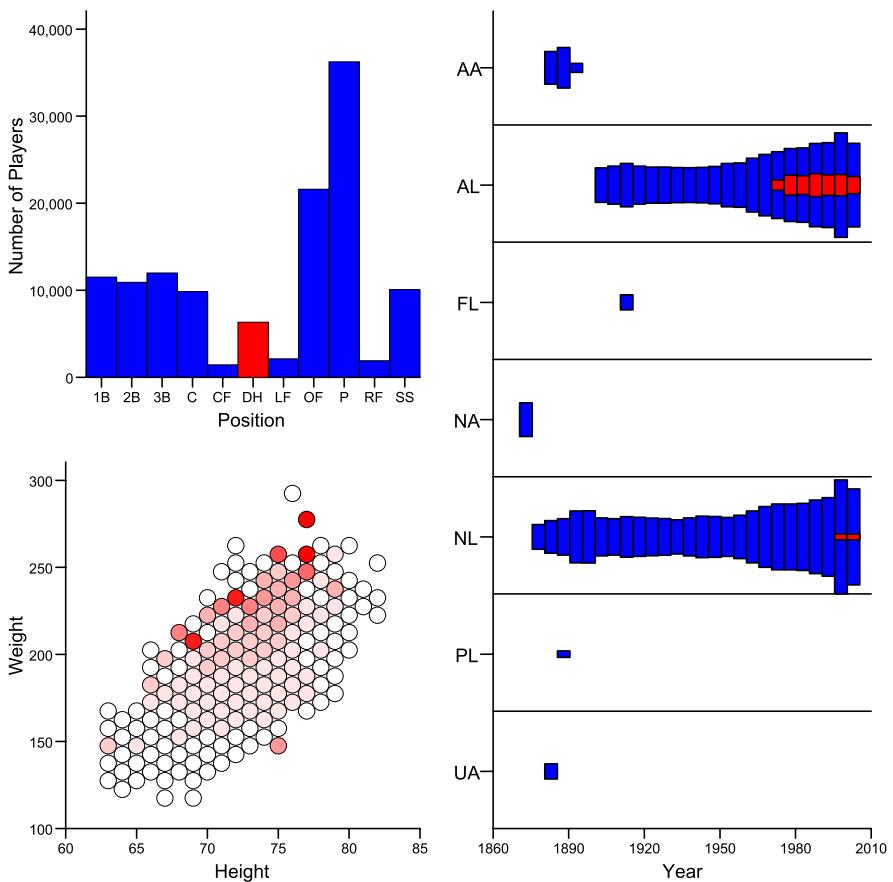


Figure 9.7. Multiple linked aggregated views. *Top left view:* barchart of players' fielding positions. The "designated hitter" position has been selected. Below is a hex-binned scatterplot showing players' height and weight, with the *brightness* indicating the fraction of designated hitters in each bin. On the *right* is a paneled view of league against year, with each *bar's* size proportional to the count of the number of players in the league for that year. Each *bar* is split into designated hitters and other positions. The selected cases (designated hitters) are drawn in *black* throughout

exactly one job – to hit the baseball hard – and so the players are powerful players, as the height–weight scatterplot shows.

On the right-hand side of Fig. 9.7, a modified faceted barchart represents the number of players in the various different baseball leagues since 1870. The flurry of early leagues settled down into the National League (NL) in 1876, which then shrank as players were raided to form the American League in 1901. Both these leagues then grew in size, with the Federal League a minor blip on the way. The degree of interest, here representing the presence of designated hitters, has been represented by splitting the bars and setting the *brightness* of the sections corresponding to the two subsets.

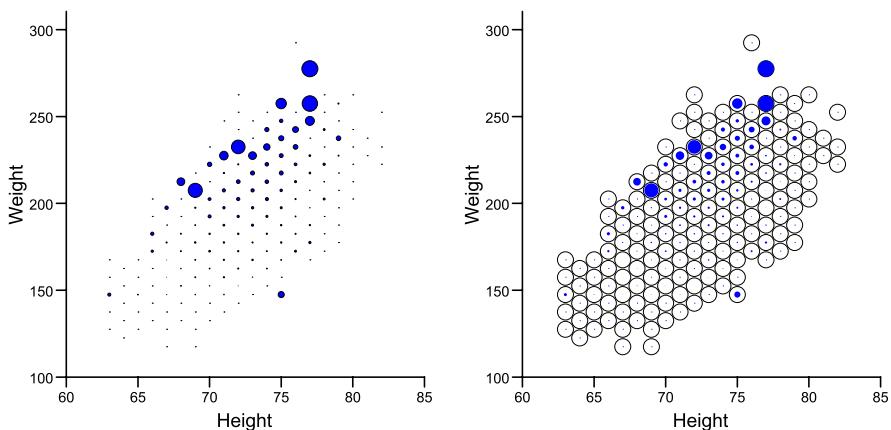


Figure 9.8. Degree of interest for aggregated points. The *left view* shows the mean degree of interest mapped to the size aesthetic. The *right view* divides each *circle* into a selected area and an unselected area

You can clearly see the appearance of the designated hitter starting in 1973. Seeing the position also appear in the NL in recent years was at first a surprise for the author. The reason is that only in recent years have there been regular-season games between the two existing leagues. Previously, two teams from different leagues would only meet in the World Series. Now it is common for NL players to play AL players. When they do so at an AL ballpark, they play under their rules and so must field a designated hitter. Thus we have statistics for designated hitters in the NL.

The Grammar of Graphics (Wilkinson, 1999) provides a useful list of basic elements that are used in graphics. All elements can be used in an augmented coordinates system where one coordinate is used for the degree of interest variable, and all may be faceted by a categorical degree of interest variable. For our final choice – using an aesthetic – we have already discussed points and intervals (bars). Since lines and paths are hard to split into parts, we will generally use a summary statistic for them or draw two different elements superimposed, one for selected and one for unselected. Areas and polygons (maps, for example) will similarly require a summary operation. Areas have the additional advantage of being able to be displayed with selected and unselected subsets stacked on top of each other in a similar fashion to bars (e.g., Fig. 9.3).

The schema element – defined in the Grammar of Graphics as a composite element – is trickier. Examples of schemas are boxplots and Chernoff faces. For these elements, superimposition is hard to interpret; the elements generally look too confusingly overlaid. A better choice is to add a dimension to differentiate between selected and unselected subsets, as in Fig. 9.9 below.

Figure 9.9 is similar to Fig. 9.5, with the boxplots replacing the parallel coordinates plot. It is much easier to compare the performance of highly paid players; they appear not to be paid for average batting and getting on base, but for power hitting – home runs and slugging. The linked boxplots allow us to compare distributions more

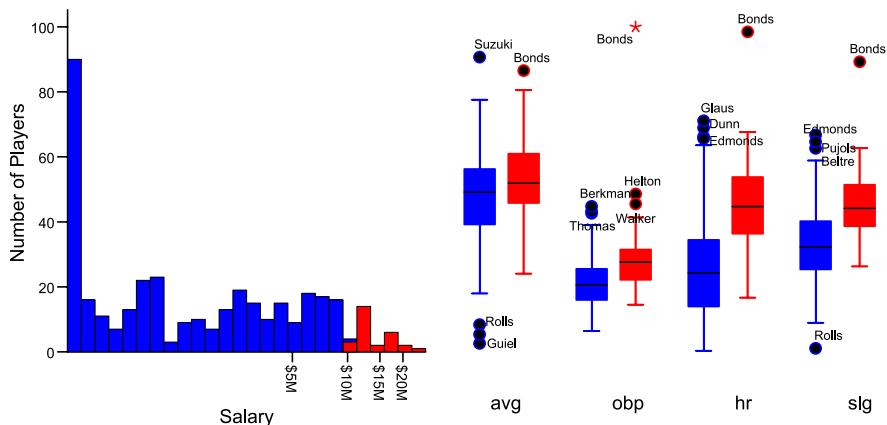


Figure 9.9. Faceting and boxplots. For the 2004 season, the *left view* shows a histogram of salary on a log scale. Players earning over \$1 000 000 were selected and are also shown in the linked *boxplot*, which shows four major measures of performance: batting average, on-base percentage, home-run rate, and slugging average

efficiently than the linked parallel coordinates plot, but, with the exception of Barry Bonds, who is an outlier in all stats, it does not allow us to see individuals' statistics in each category. We have no idea from this chart whether high on-base percentage implies a high home-run rate. Both parallel coordinates views and multiple boxplots views are useful, but they facilitate different analytic goals.

Distance-based Linking

9.5

In Sect. 9.2 we proposed a simplification to the general implementation in which the degree-of-interest value for each case must be either *zero* or *one*. Although this is the most common implementation, other methods have been proposed that relax this condition. One of particular interest is distance-based linking. In distance-based linking, instead of defining a region of the screen that is used for selection and using that to define a zero-one variable, a location in the data display is indicated, and the degree of interest measures how close each item is to that location. We demonstrate a simple version of it below in Fig. 9.10.

Interested by the relationship between body shape (height and weight) and fielding position, the analyst creates a chart of the two major fielding statistics and links it with the height/weight hex-binned scatterplot. Distance linking is used to relate the two together, with a *brightness scale* used to map the mean degree of interest in the binned scatterplot. Players in this cluster with more assists than putouts tend to be short and light. Other variations on distance-based linking are possible. In this example, we used the distance between items' graphical representations to define the distance. An alternative might be to use a distance measure based directly on the

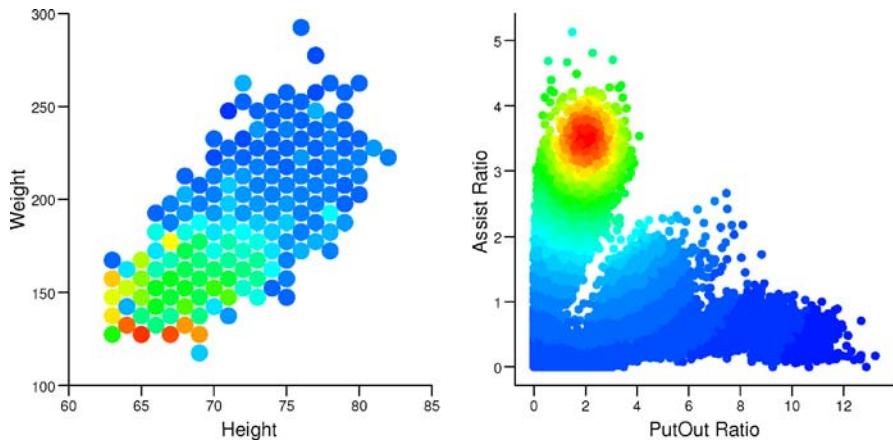


Figure 9.10. Distance-based linking. The view on the *right* shows a scatterplot of the two major fielding statistics. The user has clicked at approx. the (2, 3.5) coordinate to investigate this cluster and the degree of interest is set to reflect the closeness to this point in this plot. The *left* view shows a hex binning of the players' physical statistics, and the mean degree of interest for a bin is coded using the same brightness coding as the other plot. In both cases we are using a *brightness scale* with *black* mapping to 100 % selected (complete interest) and *light gray* mapping to 0 % selected (no interest)

data – defining a location in the data space based on the view and then using distance as measured in the data space. Normalization of the data dimensions would also probably be necessary for this. The transfer function that converts distance into degree of interest is also important. The choice of function involves similar issues as are found with choice of kernel and bandwidth for kernel density estimation. The function should give a value of one at a distance of zero and should decrease to zero as the distance increases, so that the point selected is given complete interest and points further away are assigned decreasing degrees of interest. Following are some choices that have been found effective in practice. No research has been done into the optimality of any of the following possibilities; they have been chosen in an ad hoc manner. The third method was used in the creation of Fig. 9.10 above.

- $\text{DoI} = \max(0, 1 - d / C)$, for some constant C .
- $\text{DoI} = 1 - d / (d + 1)$
- $\text{DoI} = 1 / (d + C)$, for some constant C .

9.6 Linking from Multiple Views

In Sect. 9.3 we discussed how to combine an existing degree-of-interest measure with a new selection, in the case of a binary degree of interest. The goal was to allow users to repeatedly modify the degree of interest by successive interactions with the same view. The same principle can be used to link multiple views together. We term this

a memoryless system. In a memoryless system, no history is kept of previous selections; only the current degree of interest is kept track of, with no knowledge of how that state was achieved. Thus when a selection interaction is performed, only the previous degree of interest and the view selection are used to create the resulting degree of interest. By contrast, in a system with memory, each selection operation is remembered and changing one selection runs through all existing selection operations to determine the final selection. An example is Ahlberg and Shneiderman's implemen-

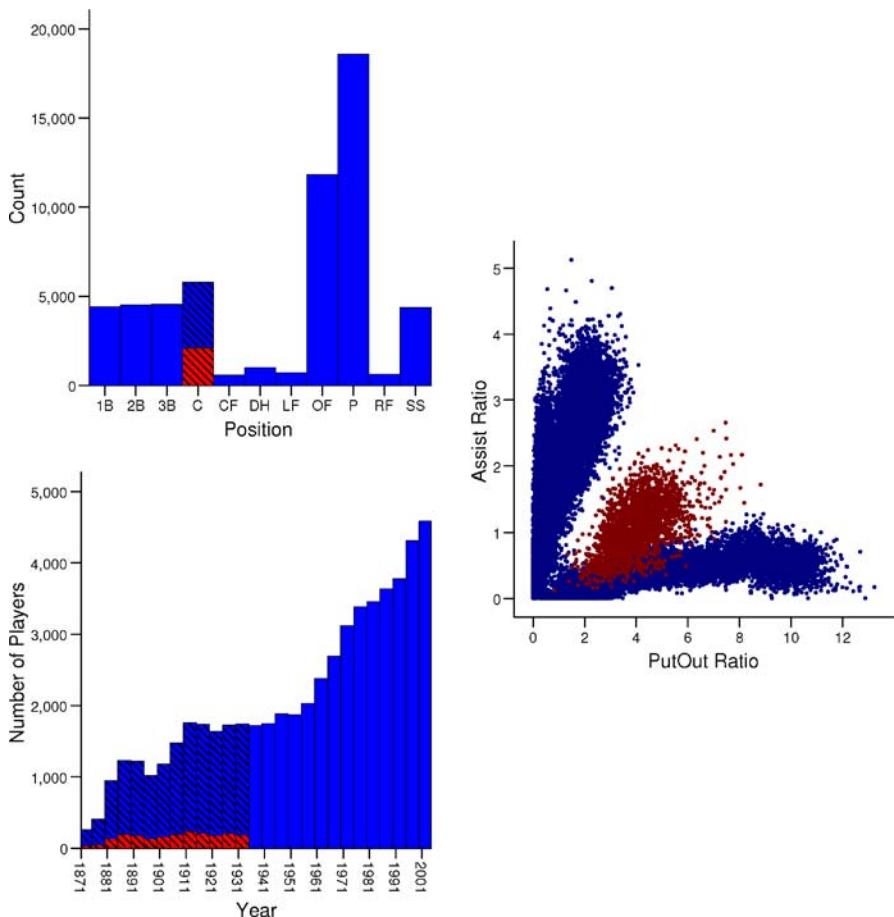


Figure 9.11. Multiple linked views with memory. The *left barcharts* show counts of number of players by fielding position and by year. The *scatterplot* shows the relationship between putouts and assists – two measures of fielding effectiveness. The user has selected sections of both *barcharts* independently; in each chart the *bars* corresponding to that selection are displayed with a *hash pattern*. The *top left view* has the catcher (C) position selected; the *lower view* has years before 1935 selected. The intersection of those selection states defines the degree of interest, which is displayed in all views in *black*

tation of dynamic queries in the FilmFinder (Ahlberg and Shneiderman, 1994). Here, each variable maintains its own selection state (for example, 1990 > *year* > 1980 or *actor* = *Connery*) and the overall selection is defined as the intersection of the individual variable selections. In Fig. 9.11 below, we show memory-based linking applied to the baseball data.

In the discussion of Fig. 9.4, we noted the extra cluster in the fielding scatterplot and the linking in that figure showed the cluster appearing only pre-1945. Figure 9.11 was created to follow up on this cluster. The position bar chart was added, as it is known that fielding position and fielding statistics are related! The linked views have

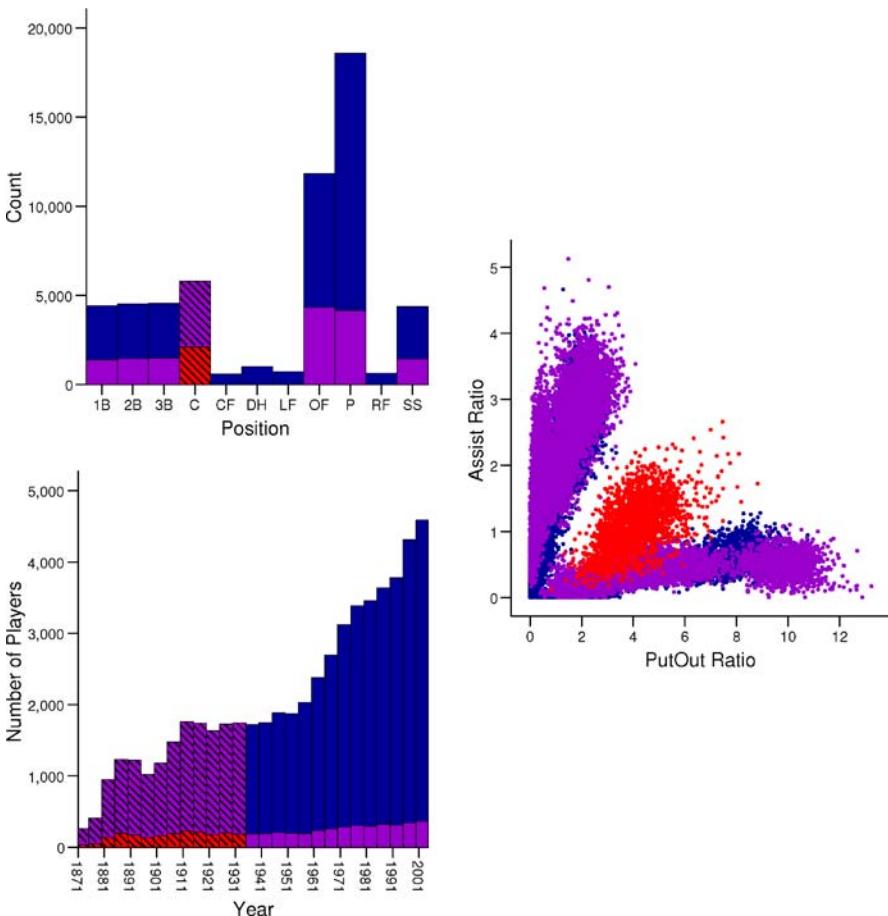


Figure 9.12. Multiple linked views with memory and nonbinary degree of interest. This figure is similar to Fig. 9.11, with the *hashed bars* showing the user's selection as in that figure. In this figure, however, the degree of interest is not a binary variable but takes three values: 0, when a case is completely unselected; 1/2, when it is selected in one of the two bar charts; and 1, when it is selected in both *barcharts*. The mapping from degree of interest to *brightness* is: 0=light gray, 1/2=medium gray, 1=black

been set up to have memory, so the user was able to select the critical year range more closely, and then each bar was successively selected in the position bar chart. When the catcher position was selected, the intersection with the year histogram identified the cluster in the scatterplot. It appears that catchers had a fundamentally different method of fielding prior to 1935.

The advantage of a system with memory is that it allows the user to make a query involving a number of variables with little effort, and is forgiving; if you make an error in adjusting one selection, it is easy to change it. The advantage of a memoryless system is in power and adaptability. It is hard to generalize a memory-based system while retaining an intuitive interface (imagine the difficulties in coordinating selections from a map, a scatterplot and a network view, for example), and it makes choosing different selection operations difficult as the method of combination of the different selections is typically fixed. From observation of existing systems, it appears that the memory-based approach is particularly good for directed queries and answering the question “Which objects satisfy this description?” and the memoryless approach is better for discovering structure in the data and answering the question “Are there unusual features in these data?”

Figure 9.12 is the most complex figure in this chapter. Each barchart keeps track of its own selection and displays that selection using a hash pattern. We are also defining a nonbinary degree of interest, in this case with three levels. As well as showing the *black* cluster that was the focus of the investigation, we can also see an interesting gap in the linear feature along the bottom of the scatterplot. A putout ratio of about eight putouts per game was rarely observed prior to 1935 and was not observed for catchers ever. The use of a nonbinary degree of interest, while more complex to use, shows additional information that would have been missed otherwise.

Linking to Domain-specific Views

9.7

One of the attractions of the linked views paradigm is that it makes it easy to integrate a view that is useful only for a particular form of data into a general system. All that is needed is that a view be able to fulfill requirements (1) and/or (3) of Sect. 9.2 and the view can be added directly into the general system. For spatial data, Unwin and Wills built a system that combined a number of statistical views with geographical views. REGARD (Haslett et al. 1990) allowed the user to manipulate maps of geographical information, containing layers of data representing sets of geographical information (towns, rivers, countries, etc.). These different layers contain entities with statistical data, so that users can create data views on one or more of these variables and use the linking system to tie the views together.

Figure 9.13 shows the simplest and most intuitive way to add a geographic view to a linked views system. We simply code the selection by *brightness* to give the well-known choropleth map. This method will work for nonbinary degrees of interest and is an intuitive view. In this figure, we can see that there is a strong relationship between marriage patterns and spatial location. It is left as an exercise to the reader to guess

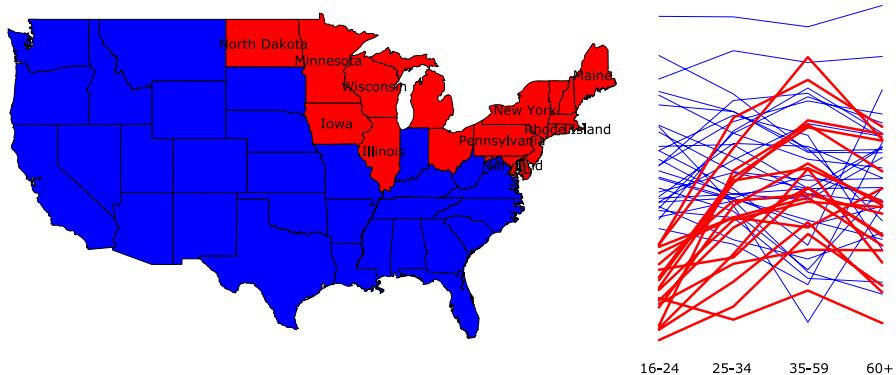


Figure 9.13. Map linking. *Left:* choropleth map of states of the United States, with the *brightness* indicating degree of interest. *Right:* parallel coordinates view of four variables, each of which indicates the percentage of people married at four different age categories. States where fewer people marry young were selected for this view

which state is the one defined by the line at the top of the parallel coordinates plot, having uniformly higher marriage rates than all other states at all age groups.

REGARD also pioneered view linking in networks, which was further developed in NicheWorks (Wills, 1999). Here the domain-specific data consist of information on nodes and links of a graph, and the goal is to use the linking mechanism to facilitate exploration of relationships between the two sets of data, as in Fig. 9.14. A variation on distance-based linking was used in this application, with distance being defined in terms of data as the graph-theoretic distance between nodes. So selecting an individual node would give it a degree of interest of 1, and then the degree of interest would flow along the graph, so nodes n steps away might have degree of interest r^n , for some $r < 1$.

A final and important domain that we will discuss is the domain of modeling results. In Sect. 9.4, we noted that Data Desk incorporated text descriptions of models within the linking paradigm. It is also useful to develop model-specific views and incorporate them within a framework. Consider hierarchical clustering, for example. Hierarchical clustering clusters data using similarities and has a natural tree representation. Figure 9.15 shows a representation of such a model for the data of Fig. 9.13, namely, the marriage patterns of states.

In the hierarchical clustering view, a leaf node has a degree of interest of either 0 or 1, depending on the selection propagated from the parallel coordinates view. Each parent node created by the clustering algorithm is defined as containing all its children and has been displayed using the mean summary method, displayed as a *brightness scale* between *black* (all data rows in this cluster were selected) and *light gray* (no data rows in this cluster were selected). This continuous degree of interest allows the clustering to be explored in the context of the original data, and, by selecting clusters and showing the linking in the parallel coordinates view, it allows users to understand what those clusters represent.

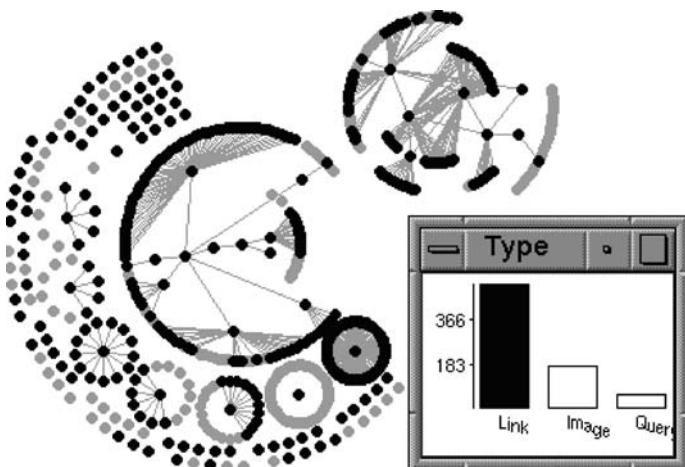


Figure 9.14. Linking between a node-edge graph and a barchart. The *graph* represents web pages. The barchart shows the type of the page, with the largest bar representing “normal” pages (not images or queries/scripts) selected

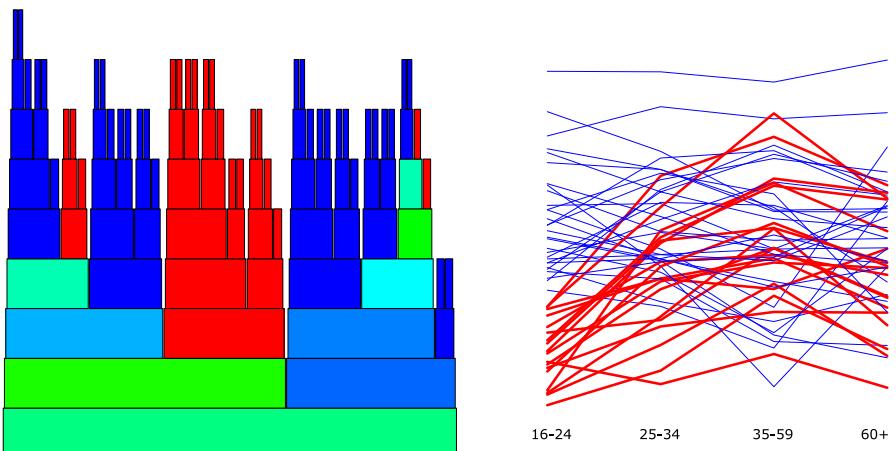


Figure 9.15. Linking between hierarchical clustering and a barchart. The *right graph* is a parallel coordinates view of marriage percentages for four different age ranges. The view on the *left* is a linked hierarchical clustering view showing aggregations into groups. As usual, the *brightness scale* represents degree of interest, with *black* representing 100 % interest and *light gray* 0 % interest

We are not even tied down to one particular representation of the clustering tree. In Fig. 9.16 we show several different representations of the same clustering tree.

The polar version is a simple transformation of the tree view shown in Fig. 9.15. The treemap in the bottom two views was invented by Shneiderman (1992). It is a space-filling design reminiscent of mosaic displays. In a treemap, we repeatedly subdivide a rectangular area in proportion to the size of the children of a node. Thus the total space is represented by the whole rectangle, which is then split into two sections (the

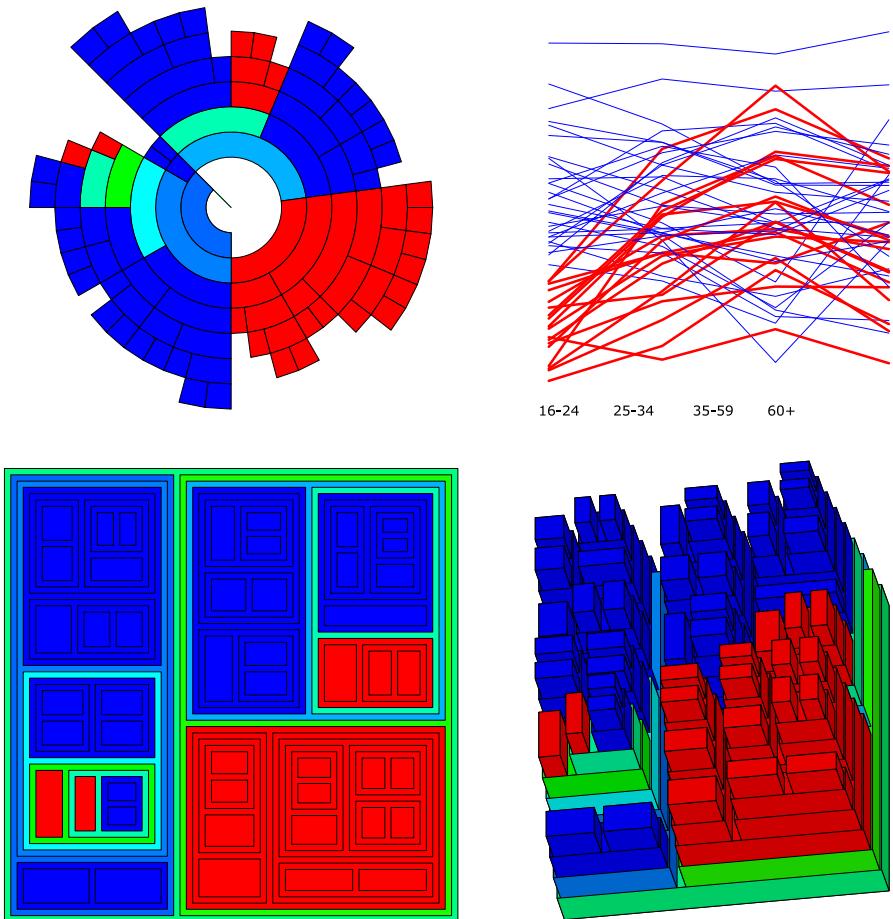


Figure 9.16. Different linked hierarchical clustering views. *Top right:* original parallel coordinates view. *Top left:* tree view of Fig. 9.15, rendered in polar coordinates. *Bottom:* two treemap views of the tree, in two and three dimensions

hierarchical clustering algorithm used produces a binary tree). Each of those is then split up based on the number of children they contain, and so on until leaf nodes are encountered. In this display a gap has been added between children to allow the inclusion to be seen more clearly.

9.8 Summary

The power of the linking technique lies in its ability to display data in one view conditionally on a user's interaction with a second view. It is therefore most useful for goals that involve comparisons of a subset of data to the remainder. Questions like "How are these outliers different?", "If X is high, what effect does that have on the

relationship between Y and Z?", and "My clustering algorithm made these items into a group. Now what does this group actually look like?" are some of the questions for which this technique is highly suitable. Linked views is also a powerful system for integrating domain-specific views into a general-purpose system. Designing a linked view, or modifying an existing view to fit the paradigm, is typically simple and, most importantly, does not require many changes to a specific view that is known to work well. This chapter has presented the necessary techniques and decisions that need to be considered to implement or use a system of linked views. We have demonstrated both simple and advanced versions of the linking environment using real-world data with sizes up to 85 000 records. Linked views is a general technique that is widely applicable, works for all types and sizes of data, and is as robust as the views that it links together. It has been implemented in several systems and with increasing computer power, and it is anticipated that it will continue to advance as a standard analysis technique.

Data Used in This Chapter

9.9

The baseball dataset is taken from *The Baseball Archive*, version 2.0, November 18, 2004, collected by Sean Lahman, which is available at <http://www.baseball1.com/>. Access to the dataset is provided free of charge (although donations are suggested) and has been available for many years. It contains 21 different tables, indexed to allow database joins. The tables used in this chapter are:

Master table: data about players including name, height, weight, handedness, birthplace, and birthdate.

Batting table: detailed data about how a player batted in a given regular season: at-bats, hits, runs, multibase hits, stolen bases, etc.

Fielding table: detailed data about how a player fielded in a given regular season: assists, putouts, errors, games played, etc.

Salary table: how much a player was paid in a given season.

Baseball players can be divided into two groups. *Pitchers*, who throw the ball, and *batters*, who attempt to hit the ball and score runs by doing so. In this section we consider only batters. The statistics for *games(G)* and *at-bats* measure how often a batter played. The other batting statistics measure how well they performed during their appearances.

Batters also play a defensive role when the opposition team is batting. They field the ball in an attempt to limit runs and to dismiss the opposition batters. A fielder has a designated position to play, with *infielders* [1B (first base), 2B (second base), 3B (third base), and SS (shortstop)] playing close to the opposition batters and *outfielders* playing further away and seeing less action in a game. An important special position is the *designated hitter* – DH. A player at this position is not required to field and has no role on the team except to attempt to hit the ball. Effectively he will have no fielding statistics. The *putout* statistic counts the number of times a fielder dismisses

an opponent, and the *assists* statistic counts the number of times a player throws the ball to another player who then puts an opposition batter out. *Errors* count the number of plays that should have resulted in a putout but did not. The decision on whether a play is an error is made by game officials.

There is a considerable amount of missing data, especially for earlier time periods. Also, when calculating rates, some players played too few games for the results to be meaningful. Therefore the following filters were applied:

- For batting data, we limit the figures to player seasons for which at least 100 at-bats were recorded.
- Fielding data are similarly restricted to player seasons for which at least 20 games were recorded.
- Salary data are only known from 1985 onwards. Any figure with salary data is restricted to the years for which salary data are available (1985–2004)

Figures 9.13, 9.15, and 9.16 use data from the US 2000 census, aggregated by the census bureau to the state level.

References

- Ahlberg, C. and Shneiderman, B. (1994). Visual information seeking: tight coupling of dynamic query filters with starfield displays, *Human Factors in Computing Systems. Conference Proceedings CHI'94*, pp. 313–317. citeseer.ist.psu.edu/ahlberg94visual.html
- Apple Computer (1992). *Macintosh Human Interface Guidelines*, Addison-Wesley Longman, Boston.
- Becker, R., Cleveland, W. and Wilks, A. (1987). Dynamic graphics for data analysis (with discussion), *Stat Sci* 2:355–395. Reprinted in: Cleveland WS, McGill ME (eds) *Dynamic Graphics for Data Analysis*, Wadsworth and Brooks/Cole, Pacific Grove, CA.
- Haslett, J., Wills, G. and Unwin, A. (1990). Spider – an interactive statistical tool for the analysis of spatially distributed data, *Int J Geograph Inf Syst* 4(3):285–296.
- Inselberg, A. (1985). The plane with parallel coordinates, *Visual Comput* 1:69–91.
- Maxis (1985). *SimCity User Manual*, Maxis, Walnut Creek, CA.
- McDonald, J., Stuetzle, W. and Buja, A. (1990). Painting multiple views of complex objects, *Proceedings of ECOOP/OOPSLA'90 European conference on object oriented programming*, Vol. 1, ACM Press, pp. 245–257.
- Microsoft (1999). *The Microsoft Windows User Experience*, Microsoft Press, Redmond, WA.
- Shneiderman, B. (1992). Tree visualization with tree-maps: A 2-d space-filling approach, *ACM Trans Graph* 11(1):92–99. citeseer.ist.psu.edu/shneiderman91tree.html
- Swayne, D., Cook, D. and Buja, A. (1991). Xgobi: interactive dynamic graphics in the X Window system with a link to s, *Proceedings of the ASA Section on Statistical Graphics*, American Statistical Association, Alexandria, VA, pp. 1–8. citeseer.ist.psu.edu/swayne91xgobi.html

-
- Swayne, D., Lang, D., Buja, A. and Cook, D. (2003). Ggobi: evolving from xgobi into an extensible framework for interactive data visualization, *Comput Stat Data Anal* 43(4):423–444.
- Tierney, L. (1990). *LISP-STAT: an object oriented environment for statistical computing and dynamic graphics*, Wiley-Interscience, New York.
- Velleman, P. (1988). *The DataDesk Handbook*, Odesta, Ithaca, NY.
- Wilkinson, L. (1999). *The Grammar of Graphics*, Springer, New York.
- Wills, G. (1999). Nicheworks – interactive visualization of very large graphs, *J Comput Graph Stat* 8(2):190–212. citeseer.ist.psu.edu/wills97nicheworksinteractive.html
- Wills, G. (2000). Natural selection: interactive subset creation, *J Comput Graph Stat* 9(3):544–557. <http://www.amstat.org/publications/jcgs/abstracts00/Wills.htm>

Visualizing Trees and Forests

II.10

Simon Urbanek

10.1	<i>Introduction</i>	244
10.2	<i>Individual Trees</i>	244
	Hierarchical Views.....	245
	Recursive Views	249
	Fitting Tree Models	254
10.3	<i>Visualizing Forests</i>	256
	Split Variables.....	257
	Data View.....	259
	Trace Plot	260
10.4	<i>Conclusion</i>	262

Introduction

Tree-based models provide an appealing alternative to conventional models for many reasons. They are more readily interpretable, can handle both continuous and categorical covariates, can accommodate data with missing values, provide an implicit variable selection, and model interactions well. Most frequently used tree-based models are classification, regression, and survival trees.

Visualization is important in conjunction with tree models because in their graphical form they are easily interpretable even without special knowledge. Interpretation of decision trees displayed as a hierarchy of decision rules is highly intuitive.

Moreover tree models reflect properties of the underlying data and have other supplemental information associated with them, such as quality of cut points, split stability, and prediction trustworthiness. All this information, along with the complex structure of the trees themselves, gives plenty of information that needs to be explored and conveyed. Visualization provides a powerful tool for presenting different key aspects of the models in a concise manner that allows quick comparisons.

In this chapter we will first quickly introduce tree models and present techniques for visualizing individual trees. Those range from classical hierarchical views up to less widely known methods such as treemaps and sectioned scatterplots.

In the next section we will use visualization tools to discuss the stability of splits and entire tree models, motivating the use of tree ensembles and forests. Finally we present methods for displaying entire forests at a glance and other ways for analyzing multiple tree models.

Individual Trees

The basic principle of all tree-based methods is a recursive partitioning of the covariates space to separate subgroups that constitute a basis for prediction. This means that starting with the full dataset at each step a rule is consulted that specifies how the data are split into disjoint partitions. This process is repeated recursively until there is no rule defined for further partitioning.

Commonly used classification and regression trees use univariate decision rules in each partitioning step, that is, the rule specifying which cases fall into which partition evaluates only one data variable at a time. For continuous variables the rule usually creates two partitions satisfying the equations $x_i < s$ and $x_i \geq s$, respectively, where s is a constant. Partitions induced by rules using categorical variables are based on the categories assigned to each partition. We refer to a partitioning step often as *split* and speak of the value s as the *cut point*.

The recursive partitioning process can be described by a tree. The root node corresponds to the first split and its children to subsequent splits in the resulting partitions. The tree is built recursively in the same way as the partitioning and terminal nodes (also called *leaves*) represent final partitions. Therefore each inner node corresponds to a partitioning rule and each terminal node to a final partition.

Each final partition has been assigned a prediction value or model. For classification trees the value is the predicted class, for regression trees it is the predicted constant, but more complex tree models exist such as those featuring linear models in terminal nodes. In what follows we will mostly use classification trees with binary splits for illustration purposes, but all methods can be generalized for more complex tree models unless specifically stated otherwise. We call a tree consisting of rules in inner nodes regardless of the type of prediction in the leaves a *decision tree*.

Hierarchical Views

10.2.1

Probably the most natural way to visualize a tree model is to display its hierarchical structure. Let us describe more precisely what it is we want to visualize. To describe the topology of a tree, we want to borrow some terminology for the graph theory. A *graph* is a set of *nodes* (sometimes called *vertices*) and *edges*. There a *tree* is defined as a connected, acyclic graph. Topologically, decision trees are a special subset of those, namely, connected directed acyclic graphs (*DAGs*) with exactly one node of indegree 0 (the root – it has no parent) and outdegrees other than 1 (i.e., at least two children or none at all).

To fully describe a decision tree, additional information is associated with each node. For inner nodes this information represents the splitting rule; for terminal nodes it consists of the prediction. Plots of tree models attempt to make such information visible in addition to displaying the graph aspect of the model. Three different ways to visualize the same classification tree model are shown in Fig. 10.1.

The tree model is based on the Italian olive oil dataset (Forina et al. 1983), which records the composition of Italian olive oils from different regions of Italy. Each covariate corresponds to the proportion (in 1/10 000th) of a fatty acid (in the order of concentration): *oleic*, *palmitic*, *linoleic*, *stearic*, *palmitoleic*, *arachidic*, *linolenic*, and *eicosenoic* acid. The response variable is categorical and specifies the region of origin. The goal is to determine how the composition of olive oils varies across regions of Italy. For illustration purposes we perform a classification using five regions: *Sicily*, *Calabria*, *Sardinia*, *Apulia*, and *North* (the latter consolidating regions north of Apulia).

Although the underlying model is the same for all plots in Fig. 10.1, the visual representation is different in each plot. Visualization of a tree model based on its hierarchical structure has to contemplate the following tasks:

- Placement of nodes
- Visual representation of nodes
- Visual representation of edges
- Annotation

Each task can be used to represent additional information associated with the model or data. Visual *representation of a node* is probably the most obvious way to add such information. In the first (top left) plot, a node consists solely of a tick mark with an annotation describing the split rule for the left child. In the second (top right) plot, a node is represented by a rectangle whose size corresponds to the number of cases

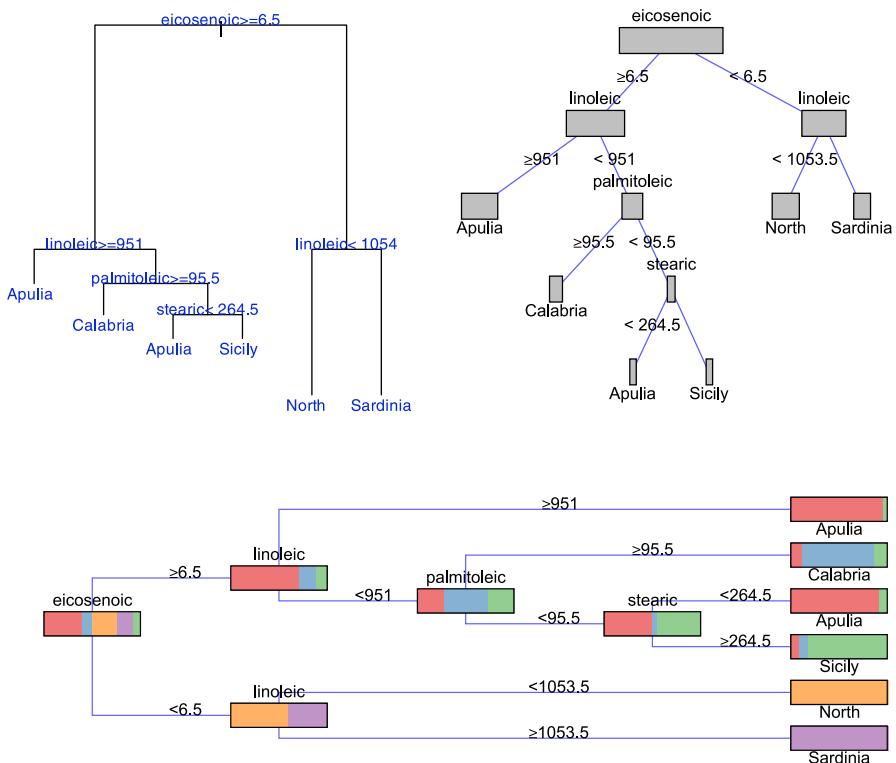


Figure 10.1. Different ways to visualize a classification tree model

of the training data falling into that node. The annotation describes the split variable. Finally, in the third (bottom) plot, each node is represented by a rectangle of the same size, but the colors within show the proportions of classes falling into a given node.

Advanced techniques known from area-based plots can be used in hierarchical views as well if we consider nodes as area-based representations of the underlying data. Figure 10.2 illustrates the use of *censored zooming* in conjunction with tree node size.

The top plot shows node representation without zoom, that is, the size of the root node corresponds to all data. All subsequent splits partition these data, and hence the node area, until terminal nodes are reached. If plotted truly proportionally, the last two leaves split by the *stearic* variable would be hardly visible. Therefore a minimal size of a node is enforced, and the fact that this representation is not truly proportional is denoted by a red border.

To provide a truly proportional comparison of small nodes, we can enlarge all nodes by a given factor. In the bottom plot a factor of four was used. Now those small nodes can be distinguished along with the class proportions, but large nodes would need to be four times as big as in the first plot, obscuring large portions of the

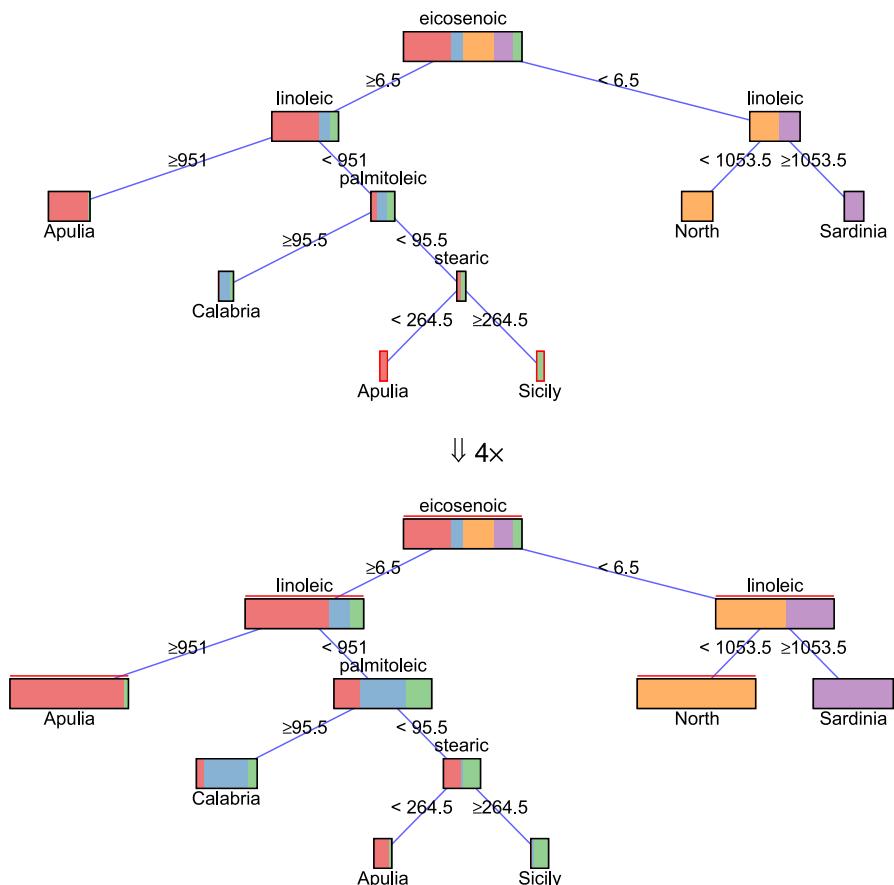


Figure 10.2. Censored zoom of nodes. *Bottom plot:* censored zoom ($\times 4$) of *top plot*. Nodes that would appear too large are censored at a maximum allowed size and flagged by a red line

plot and possibly other nodes. Therefore we also enforce a maximal size of a node. Again, to denote nodes that are not shown proportionally due to upper censoring, we use a red line along the top edge of the node.

The *placement of nodes* is a task that has been discussed intensely in the graph visualization community. For small trees, simple approaches, such as a bottom-up space partitioning, work well. As the trees grow larger, node layout becomes more challenging. For tree model visualization, however, associated information is in most cases more important than differences in local topology, especially where the structure is imposed by the tree-growing algorithm. Therefore interactive approaches, allowing the user to explore the tree model by local magnification while retaining global context, are recommended for large tree models.

In the above examples, basic lateral placement is performed by an equidistant partition of the available space. Only the first plot uses nonequidistant placement of nodes in the direction of tree depth, namely, the distance of two nodes in this di-

rection is proportional to the impurity decrease and thus in a sense to the “quality” of the split. The third plot uses special placement in that it is rotated 90° counter-clockwise relative to the usual representation and all terminal nodes are aligned to facilitate easy comparison of the class proportions.

The visual *representation of edges* is usually restricted to drawing direct or orthogonal lines. Nevertheless, more elaborate representation of edges, such as polygons whose width is proportional to the number of cases following that particular path, is another possibility, creating a visual representation of the “flow” of data through the tree.

Annotations are textual or symbolic representations displayed along the nodes or edges. In Fig. 10.1 annotations describe predictions and splitting rules. Although annotations can be useful, they should be used with caution because they can easily clutter the plot and thus distract from the key points to be conveyed.

Overloading plots with information can offset the benefits of the plot, in particular its ability to provide information at a glance. When the representation of a node is too large, because it, e.g., includes a list of statistics or additional plots, it will consume so much space that it is only possible to display very few levels of the tree on a screen. The same applies to a printed version, because the size of a sheet of paper is still limited. Therefore additional tools are necessary to keep track of the overall structure in order not to get lost. Most of these tools, such as zoom, pan, overview window, or toggling of different labels, are available in an interactive context only. Especially for an analysis, a visualization of additional information is required. There are basically two possibilities for providing such information:

- Integrate the information in the tree visualization.
- Use external linked graphics.

Direct integration is limited by the spatial constraints posed by the fixed dimension of a computer screen or other output medium. Its advantage is the immediate impact on the viewer and therefore easier usage. It is recommended to use this kind of visualization for properties that are directly tied to the tree. It makes less sense to display a histogram of the underlying dataset directly in a node because it displays derived information that can be more comfortably displayed outside the tree, virtually linked to a specific node. It is more sensible to add information directly related to the tree structure, such as the criterion used for the growth of the tree.

External linked graphics are more flexible because they are not displayed directly in the tree structure for each node but are only logically linked to a specific node. Spatial constraints are less of a problem because one graphic is displayed instead of many for each node. The disadvantage of linked graphics is that they must be interpreted more carefully. The viewer has to bear in mind the logical link used to construct the graphics as it is not visually attached to its source (node in our case).

There is no fixed rule as of what kind of information should be displayed inside or outside the tree structure. A rule of thumb says that more complex graphics should use the external linked approach, whereas less complex information directly connected with the tree structure should be displayed in the tree visualization.

Recursive Views

10.2.2

In the introduction we described tree models as recursive partitioning methods. Consequently it is only natural to display partitions induced by the tree, resulting in an alternative way of visualizing tree models. In what follows we will describe visualization methods that are based on the partitioning aspect of the models instead of the hierarchical structure.

Sectioned Scatterplots

Splitting rules are formulated in the covariate space; therefore a way to visualize a tree model is to visualize this space along with the induced partition boundaries. For univariate partitioning rules those boundaries lie on hyperplanes parallel to the covariate axes.

Due to this fact we can use something as simple as a scatterplot as a 2-D projection. In this view, all splits featuring any of the two plotted variables are clearly visible. Such *sectioned scatterplot* featuring the first two split variables is shown in Fig. 10.3 along with the associated tree.

It is the same model as in Fig. 10.1. Each region is denoted by a particular color in the scatterplot, and partition boundaries are added based on the tree model.

The first split of the tree uses an *eicosenoic* variable to separate oils originating in northern Italy and Sardinia from other regions. It is clearly visible in the sectioned scatterplot that this split is indeed very distinctive.

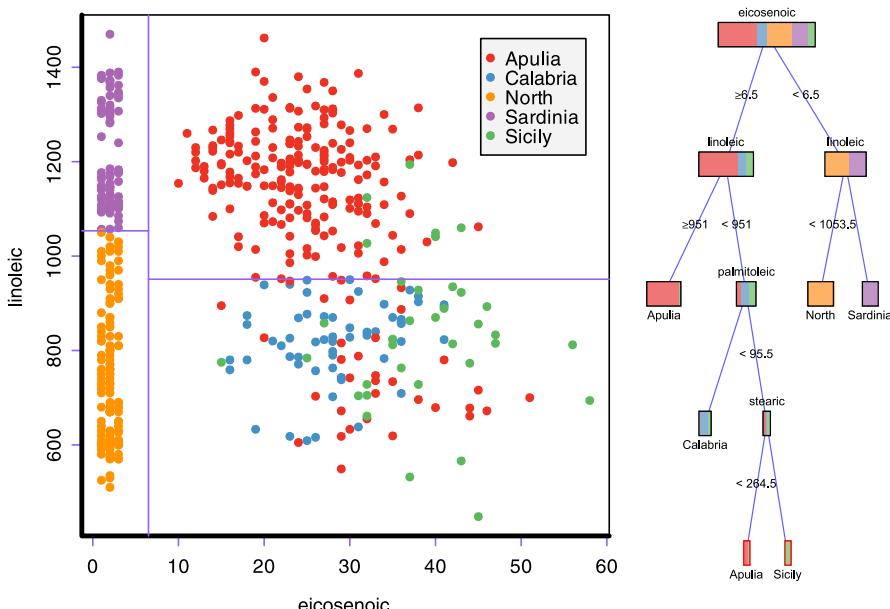


Figure 10.3. [This figure also appears in the color insert.] Sectioned scatterplot (left) showing root splits and splits in its children of a classification tree (right)

Both of the following inner nodes use a *linoleic* variable to further separate Sardinian oils from northern parts of Italy and on the right-hand side Apulian oils from Sicily and Calabria. Further splits are no longer visible in this projection because they feature other variables and are thus parallel to the visible plane. Nevertheless, it is possible to analyze such subsequent splits, especially in an interactive context, by a succession of sectioned scatterplots using drill-down techniques, following a few basic rules.

Sectioned scatterplots should be preferably created using variables that are adjacent in the tree, that is, using split variables of nodes that are connected by an edge. This ensures that both splits are visible in the projection.

Also, the plotted data should be restricted to the data falling in the node closer to the root. In Fig. 10.3 we have used the entire dataset, since we were interested in showing splits of the root node and its children. Figure 10.4 presents a sectioned scatterplot based on data further down the tree, namely, the partition in the bottom-right part of the scatterplot in Fig. 10.3.

Sectioned scatterplots are useful for investigating the vicinity of a cut point. Some cut points are placed in a noisy area, while others are much more clear, as is illustrated in Fig. 10.3. However, they cannot capture more than two covariates used in subsequent splits and thus remain suitable mainly for local analysis of a tree model. In an interactive setting, however, it is possible to quickly “drill-down” from Fig. 10.3 to Fig. 10.4. Linked highlighting further helps to retain the context, especially with the help of a linked hierarchical view.

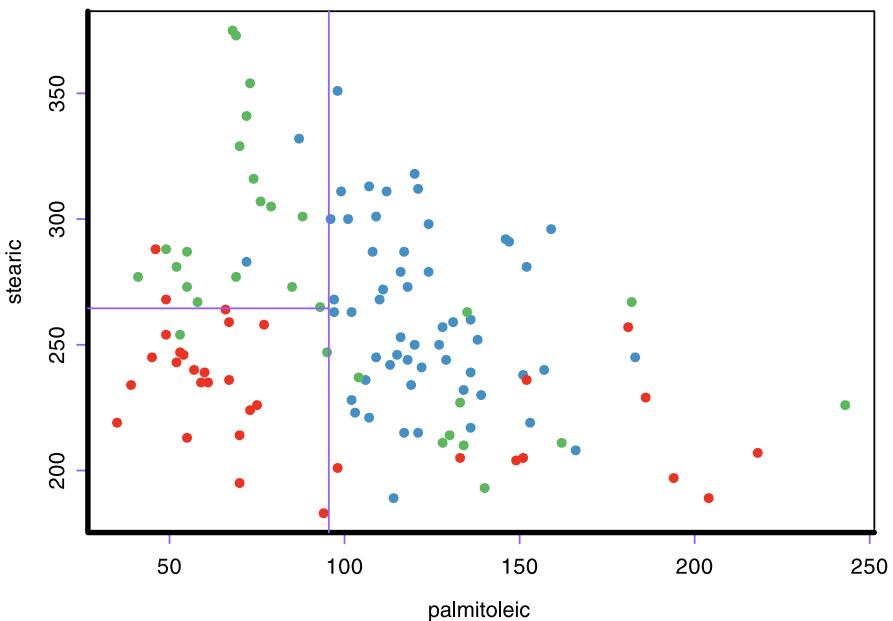


Figure 10.4. Drill-down using a sectioned scatterplot based on a subgroup induced by the tree model

A generalization to multiple dimensions is not straightforward. Although a rotating 3-D scatterplot with splits as hyperplanes represented by meshes proves to be useful, higher dimensions are beyond reach.

Several extensions to sectioned scatterplots are useful in specific situations. In cases where the same variable is used in the tree several times at different depths, it is advisable to vary the opacity or luminance of the partition boundary lines according to their depth, making more “deep” splits lighter. This provides a visual aid when interpreting the plot.

Another technique involves shading of the plot background based on either the depth of the visible partition (depth denoted in shades of gray: *depth-shading*) or the predicted value (semitransparent category color or hue of the predicted value: *prediction shading*). The latter emphasizes misclassified cases or outliers with high absolute residual value because correctly predicted cases blend better into the background.

Scatterplots are primarily useful for continuous variables. If a tree model uses categorical variables, a local treemap can prove useful. Such a plot is in principle similar to mosaicplots, but categories falling into the same node are grouped together. We will discuss treemaps in the following section.

Treemaps

One way of displaying all partitions is to use area-based plots where each terminal node is represented by a rectangle. *Treemaps* belong to this plot category. The main idea is to partition available rectangular plot space recursively in the same way that the tree model partitions data. Therefore treemaps are data-driven representations of the model.

The rectangular area of the treemap corresponds to the full dataset. In the first step this area is partitioned *horizontally* according to the proportions of cases passed to each child node. In the next step each such partition is partitioned *vertically* corresponding to case proportions in its children. This process is repeated recursively with alternating horizontal and vertical partitioning directions, as illustrated in Fig. 10.5, until terminal nodes are reached.

In the resulting plot each rectangle corresponds to a terminal node. The area of each rectangle is proportional to the number of cases falling into that terminal node.

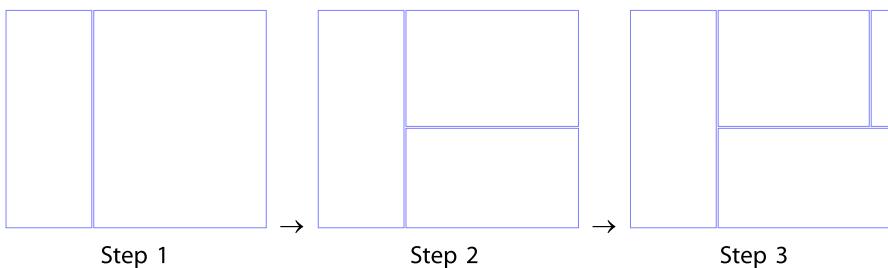


Figure 10.5. Construction of a treemap consisting of three subsequent binary splits

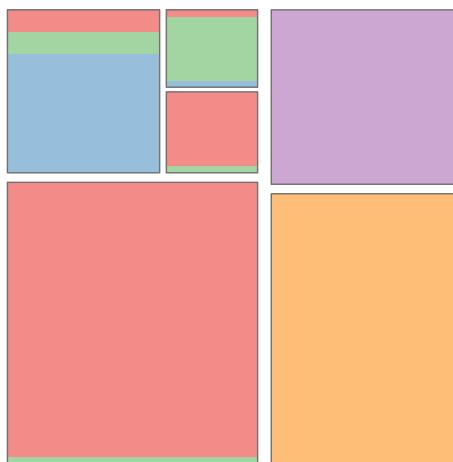


Figure 10.6. Treemap with *stacked bars* representing response classes. Color coding and data are the same as in Fig. 10.3

It is helpful to adjust spaces between partitions to reflect the depth at which a given partitioning took place, showing splits closer to the root with larger gaps.

Treemaps are useful for assessing the balance of a tree model. In very noisy scenarios trees tend to attempt splitting off small, reasonably homogenous subgroups, while leaving a large chunk of cases in one node that is hard to separate. Such behavior is easily detected in treemaps as large terminal nodes.

Moreover, treemaps are suitable for highlighting or brushing, allowing the comparison of groups within terminal nodes. A treemap of the model from Fig. 10.1 with colors stacked by group is shown in Fig. 10.6.

It is clearly visible that the tree model is able to split off large homogenous groups successfully, but more subsequent splitting is necessary for nodes visible in the upper-left part of the plot.

Treemaps described here are an extension of those used in computer science information visualization of hierarchically stored contents. They are also related to *mosaicplots*. More precisely a mosaicplot is a treemap of a decomposition tree, that is, a tree whose splits of the same depth use the same categorical splitting variable and have as many children as there are categories in the data.

The main advantage of treemaps is their very efficient use of display space. They allow absolute comparison of nodes and subgroup sizes while maintaining context of the tree model. They scale well with both increasing dataset size and tree model complexity. What they cannot show is information about splitting criteria, and they do not allow direct relative comparison of groups within nodes. An alternative visualization technique exists for the latter task.

Spineplots of Leaves

Another useful plot for tree model visualization is the *spineplot of leaves* (SPOL). By not alternating the partitioning direction as in treemaps, but constantly using horizontal partitioning, we obtain a plot showing all terminal nodes in one row.

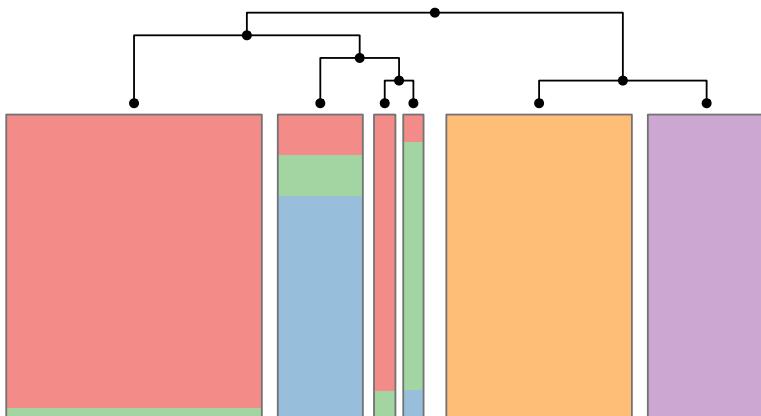


Figure 10.7. [This figure also appears in the color insert.] Spineplot of leaves brushed by response categories with superimposed tree model. The associated tree is sketched on *top* for easier identification of individual leaves

Due to the fixed height, it is possible to visually compare sizes of the terminal nodes that are proportional to the width of the corresponding bar. Moreover, relative proportions of groups are easily comparable when using highlighting or brushing.

A sample spineplot of leaves is shown in Fig. 10.7. The displayed data and model are the same as in Fig. 10.6, as well as the color brushing. Each bar corresponds to a leaf, and the width of each bar is proportional to number of cases in that particular node.

We can clearly see relative proportions of groups within each node. In addition, it is possible to add a simple annotation on top of the plot in the form of a dendrogram of the represented tree. As with treemaps, it is advantageous to choose the size of gaps between bars according to the depth of the split.

SPOLs are mainly useful for comparing group proportions within terminal nodes. They are similar to *spineplots*, which allow the comparison of groups within categories of a variable. They differ in that a “category” is in fact membership of a particular terminal node and uses different rules for gaps between bars.

In this section we have discussed several alternative techniques for visualizing tree models based on the idea of recursive partitioning. The shown methods focus on the visualization of splits, their sequence, and the application of the model to data. One important property of all visualization techniques presented is their applicability to arbitrary subsets of the data. Although most illustrations used training data and the corresponding fitted tree model, it is also feasible to visualize test data instead. Where a view of the training data highlights the adaptability of the model, the view of test data focuses on stability and overfitting. Moreover, it is possible to compare both views side by side.

This leads us to further important aspects of a tree model – the credibility and quality of the splits and the entire model. In the next section we want to briefly discuss tree model construction and present visualization methods that incorporate information about split quality into both existing and new plots.

Fitting Tree Models

So far we have discussed methods for visualizing tree models on their own and including data the models are applied to. There is, however, more information associated with each node that waits to be visualized. In order to understand tree models better, we need to know more about the process of fitting tree models.

Although a tree model is straightforward to interpret and apply, its construction is not trivial. In theory, we would like to consider all possible tree models and pick the one that fits the given data best, based on some loss function. Unfortunately this proves to be unfeasible save for trivial examples because the computational cost increases exponentially with tree size.

Therefore several other approaches have been suggested for fitting tree models. The most commonly used algorithm CART (Classification and Regression Trees) was introduced by Breiman et al. (1984). It performs a greedy local optimization as follows: for a given node, consider all possible splits and choose the one that reduces the relative impurity of the child nodes most relative to the parent node. This decrease of impurity (and hence increase of purity) is assessed using an impurity criterion. Such a locally optimal split is then used and the search is performed recursively in each child node.

The growth is stopped if one of the stopping rules is met. The most common stopping rules are a minimal number of cases in a node and a minimal requirement on the impurity decrease. In practice it is common to relax the stopping rule and use pruning methods; however, discussion of pruning methods is beyond the scope of this chapter. Nevertheless, visualization can be useful for pruning, especially in an interactive context where pruning parameters can be changed on the fly and reflected in various displays.

Measures of impurity can be any arbitrary convex functions, but the commonly used measures are entropy and Gini index, which have theoretical foundations (cf. Ripley, 1996). It is important to note that this search looks for a local optimum only. It has no way to “look ahead” and consider multiple splits at once. Nevertheless, it is computationally inexpensive compared to a full search and performs considerably well in practice.

The consequence of committing to a local optimum at each split point is a potential instability of the model. Small changes in the training data can cause a different split to be chosen. Although the alternate split may lead to a very similar decrease of impurity, the resulting partition can be entirely different. This will have a big impact on any following splits and thus produce an entirely different tree model. We want to present a visualization technique that allows us to learn more about decisions made at the node level during tree model fitting.

Mountain Plots

The basic idea of a *mountain plot* (Urbanek, 2003) is to visualize the decrease of impurity over the entire range of the split variable. This is illustrated on a binary classification problem in Fig. 10.8. In this particular example a binary response denotes

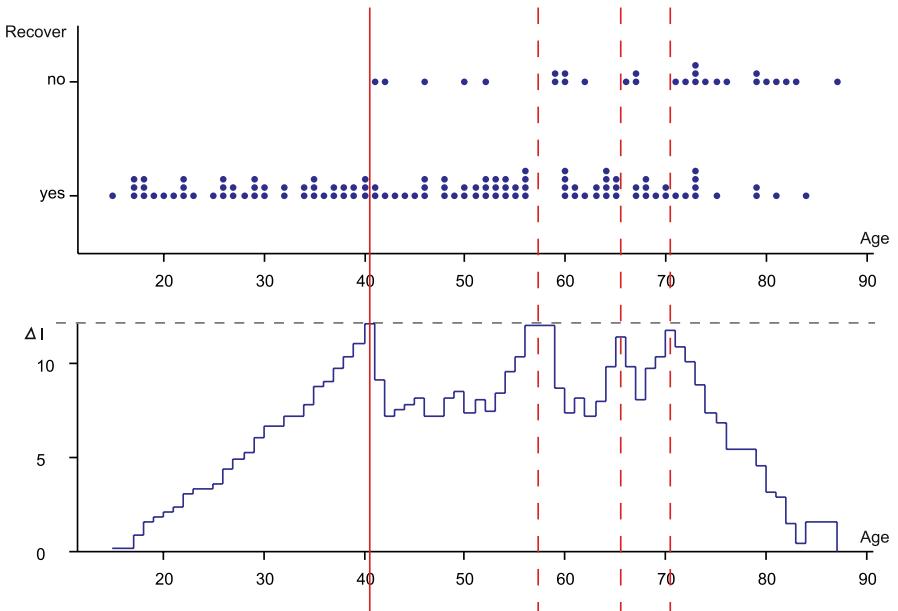


Figure 10.8. Stacked dotplot side by side with split variable (*Age*) and target variable (*Recover*) along with the corresponding *mountain plot* showing the impurity decrease for each cut point. The optimal cut point is denoted as a *solid red line*, runner-up splits as *dotted red lines*

whether a patient was able to recover from a diagnosed meningitis disease, whereas the predictive variable *Age* refers to the patient's age at the time of the diagnosis.

The top part of the figure shows a stacked dotplot of the split variable grouped by binary response. The bottom part of the plot shows a mountain plot. The value of the empirical impurity measure is constant between data points and can change only at values taken by the data. The value of the impurity decrease is by definition zero outside the data range.

In the presented example it is clearly visible that there are three alternative splits that come very close to the “optimal” cut point chosen by the greedy algorithm.

The competition for the best split is not limited to a single variable. Figure 10.9 illustrates a competition among two different variables in a regression tree. The models are based on the Boston housing dataset by Harrison and Rubinfeld (1978).

Although both splits have almost identical impurity-decrease maxima, the data show different patterns. The relationship seen in the left part of the plot is probably better modeled by a linear model, whereas on the right-hand side we see a change in behavior around the chosen cut point.

By plotting mountain plots of candidate variables on the same scale, we can assess the stability of a split. If there is a dominating covariate with a clear optimum, the split will be stable. On the other hand the presence of competing splits in the range of the optimal split indicates possible instability. Mountain plots also show which re-

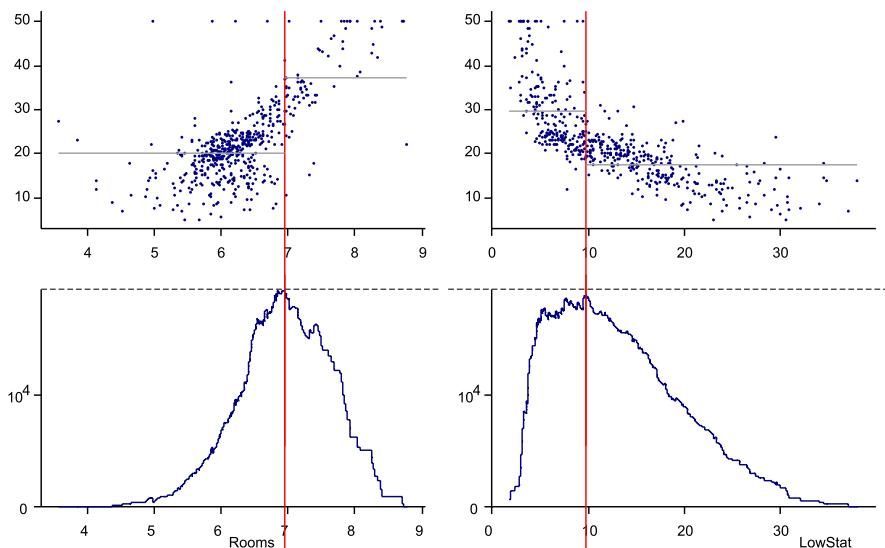


Figure 10.9. Two mountain plots of variables *Rooms* and *LowStat* and corresponding scatterplots vs. response variable. Red lines: optimal splits; gray lines in scatterplots: means for each partition

gions of competing variables are in the vicinity of the optimum, thus allowing domain knowledge to be taken into account.

The name “mountain” plots is derived from the fact that the plots usually resemble a profile of a mountain range. They are mainly useful for assessing the quality of a split along with potential competing splits. This information can be used to interactively influence the tree construction process or to construct multiple tree models and compare their behavior.

10.3 Visualizing Forests

So far we have been discussing visualization of individual tree models. We have shown, however, that there is an inherent volatility in the choice of splits that may affect the stability of a given model. Therefore it is useful to grow multiple trees. In what follows we will briefly introduce tree ensemble methods and present visualization methods for forests consisting of multiple tree models.

There are two main approaches to generating different tree models by making changes to:

Training data: changes in the training data will produce different models if the original tree was unstable. Bootstrapping is a useful technique to assess the variability of the model-fitting process.

Splits: allow locally suboptimal splits that create different partitions in order to prevent the greedy algorithm from getting stuck in a local optimum, which may not necessarily be a global optimum.

Model ensemble methods leverage the instability of individual models to improve prediction accuracy by constructing a predictor as an aggregate of multiple individual models. *Bagging* (Breiman, 1996) uses bootstrapping to obtain many tree models and combines their prediction results by aggregation: majority voting for classification trees and averaging for regression trees. In addition, *random forests* (Breiman, 1999) add randomness by choosing candidate split variables from a different random set in each node.

Split Variables

10.3.1

Bootstrapping models provide a useful tool to analyze properties of the fitted models and therefore shed more light on the underlying data. One of the many advantages of tree models is their ability to perform implicit variable selection. Given a dataset, a tree-growing algorithm will create a hierarchical structure of splits forming the tree. Only variables used in the splits will be evaluated by the model; therefore any other variables are implicitly dropped. In the following discussion we want to illustrate the visualization of forests on Wisconsin breast cancer data (Mangasarian and Wolberg, 1990).

For this purpose we generate 20 trees using bootstrapping. In each bootstrap iteration we grow a tree using the regular CART algorithm. Let us first concentrate on the variables used in the models. A global overview is given in the left plot of Fig. 10.10. Each bar displays how often the corresponding variable was used in the models. The most often used variable is UCS (20 times) and the least often used variable is Mts, which was used just once. Due to the rather small number of variables to choose from, there is no variable omitted by all models.

Clearly this view is very coarse, because it does not take into account what role the variable plays in the models. The number of splits can double with increasing depth, whereas the number of involved cases decreases. Therefore the fact that a variable is used often does not necessarily mean that it is really important, especially if it used

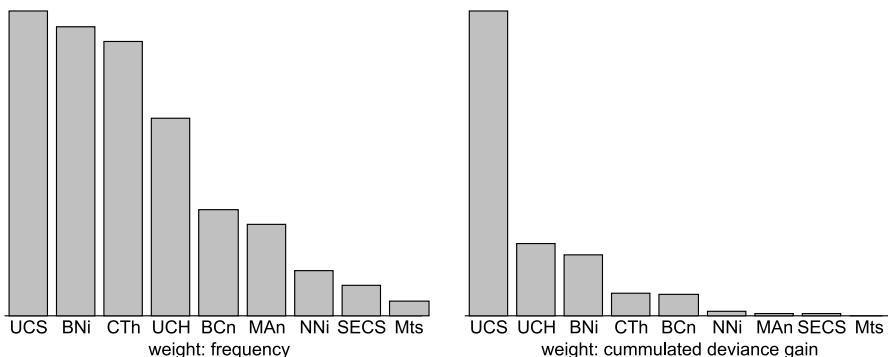


Figure 10.10. Left: frequency of use of individual variables in 20 bootstrapped tree models. Right: cumulated deviance gain in splits featuring the corresponding variable

mainly in the fringe for small groups. Therefore it is advisable to weight the contribution of each split by a cumulative statistic such as the decrease of impurity.

The cumulative value of impurity decrease for each variable of the 20 bootstrapped trees is displayed in the right plot of Fig. 10.10. The variables in each plot are ordered by the bar height, representing their importance. We see that UCS is by far the most influential variable, followed by UCH and BNI.

When making inference on the displayed information, we need to be cautious and keep the tree properties in mind. Variable masking can heavily influence the results of such analyses. Given two highly correlated variables, it is very likely that they will produce very similar split results. Therefore the CART algorithm guided by the bootstrap will pick one of them at random. Since the decision was made, the other variable is not likely to be used anymore. If one of the variables is “weaker,” it will hardly appear in any model, even though in the absence of the stronger variable it may still perform the best out of all the other variables.

To analyze that behavior, but also to see how different the tree models are, it is necessary to take both the variable and the individual tree into account. Two-dimensional weighted fluctuation diagrams showing trees and split variables are shown in Fig. 10.11. Variables are plotted on the y -axis, the models on the x -axis. The area of each rectangle is proportional to the cumulative impurity decrease of all splits using a specific variable in the tree model. In general, fluctuation diagrams are useful for detecting patterns and comparisons in both the x and y directions.

Focusing on the largest gains, we can distinguish four different model groups. In 15 models, UCS is the most influential variable, followed by UCH with 3 models and BNi and BCn with one model each. Looking at the large group of 15 models we can also spot several patterns. In 8 cases, UCH is also used, although not contributing as heavily as in its dominant position, but then we see another 7 cases where UCH is not used at all. Visually we get the impression that BNi replaces UCH in those cases, which hints at variable masking. We see a similar behavior with UCS and UCH, too.

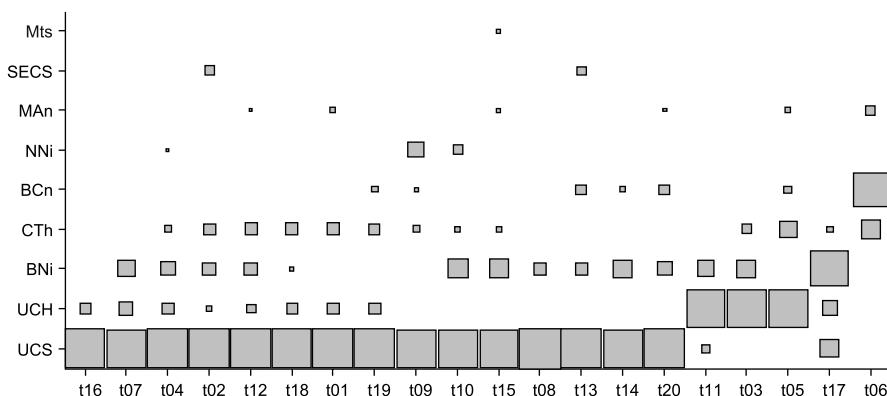


Figure 10.11. Fluctuation diagram of trees and variables displaying cumulated deviance gain of splits featuring that combination of tree and split variable

This overall impression indicates that bootstrapping indeed produces very different models, and we see a confirmation of the tree model instability for this dataset.

With large numbers of trees, an alternative representation based on parallel coordinate plots can be used. Each coordinate corresponds to a tree and each case to a variable. The value of the cumulative gain for each combination of tree and variable is then plotted on the axes. The ordering of axes is important to obtain a coherent picture. Some possible heuristics include ordering by the value of the most influential variable and distance measures based on the global weight of each variable.

Data View

10.3.2

The importance and use of variables in splits is just one aspect of the tree models to consider. In Sect. 10.2.2, we discussed another way of visualizing trees that allowed an assessment of cut point in the data context, sectioned scatterplots. Fortunately, sectioned scatterplots can also be used for the visualization of forests, preferably using semitransparent partition boundaries.

Such a sectioned scatterplot of a forest is shown in Fig. 10.12. To make the classification more difficult, we have increased the granularity of the response variable of the olive oil data to nine regions. The sectioned scatterplot displays variables *linoleic* vs. *palmitoleic* and partition boundaries of 100 bootstrapped trees. The use of semitransparent boundaries allows us to distinguish between occasionally used cut points that are shown as very faint lines and frequently used cut points shown in dark blue.

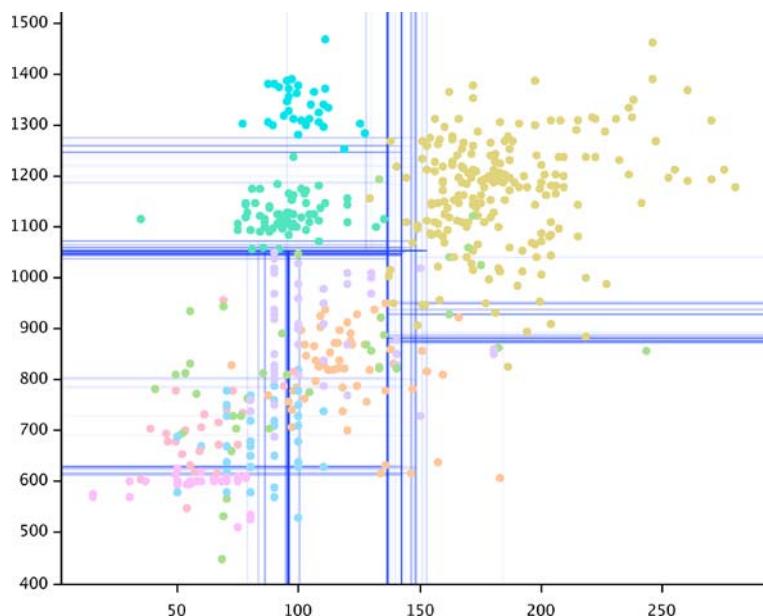


Figure 10.12. [This figure also appears in the color insert.] Sectioned scatterplot of a forest of 100 trees

In contrast to sectioned scatterplots for individual trees, we do not have the convenient ability of a drill-down, unless several models agree on the same subset. Therefore the aim of the visualization technique described in the next section is to show all trees and their splits at a glance.

10.3.3

Trace Plot

The aim of a *trace plot* is to provide a plot that allows comparison of arbitrarily many trees with respect to splits, cut points, and the hierarchical structure. This is not possible using any of the visualization methods described so far.

The basis of the trace plot is a rectangular grid consisting of split variables as columns and node depths as rows. Each cell in this grid represents a possible tree node. To distinguish actual split points, each cell contains a glyph representing possible split points. For continuous variables it consists of a horizontal axis, and a split point is represented by a tick mark. Categorical variables are shown as boxes corresponding to possible split combinations. Every two adjacent inner nodes are connected by an edge between their split points.

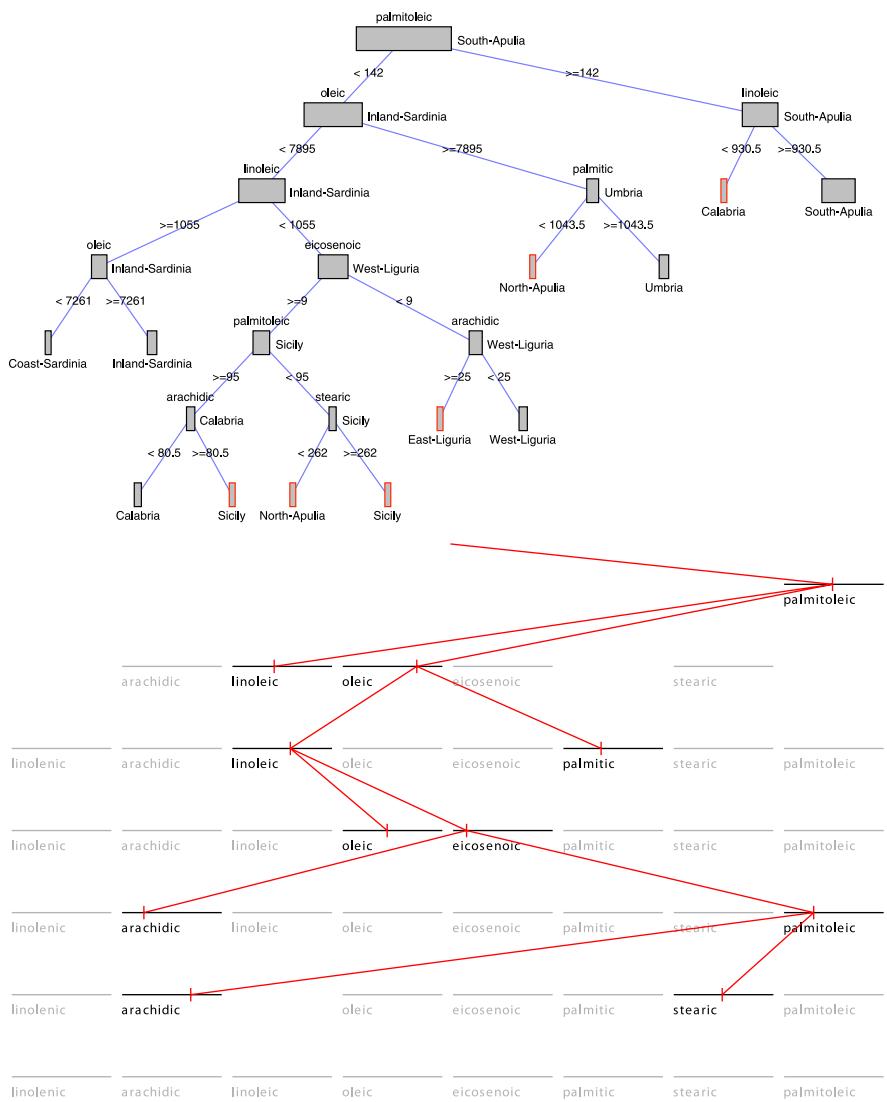
A classification tree and its trace plot is shown in Fig. 10.13. The root node features a split on the variable *palmitoleic*, which is represented by the rightmost column. Its child nodes use splits on the variables *linoleic* and *oleic*, hence the two edges leading from the root node to the next row of splits. There are no further inner nodes as children of the *linoleic* split; therefore the branch ends there. Analogously, all inner nodes are drawn in the trace plot until terminal nodes are reached.

It is evident that all splits of the tree can be reconstructed from its representation in the trace plot because every cut point is shown in the trace plot. Equally, it is possible to reconstruct the hierarchical structure of the tree due to the presence of edges in the trace plot.

Moreover, the trace plot removes an ambiguity known from hierarchical views: the order of the child nodes is irrelevant for the model, whereas swapping left and right children in the hierarchical view produces quite different hierarchical plots. In a trace plot the order of the child nodes is defined by the grid and therefore fixed for all trees in the plot.

One important advantage of trace plots is the ability to display multiple tree models simultaneously, superimposing all models on the same grid. A trace plot of 100 bootstrapped classification trees is shown in Fig. 10.14. This confirms the ability of bootstrapping to produce models that deviate from certain local optima.

To prevent overplotting, we use semitransparent edges. Consequently, often used paths are more opaque than infrequently used paths. We can clearly see that the first split always uses the *palmitoleic* variable. In the next step, however, there are several alternatives for the splits. Some patterns seem to be repeated further down the tree, indicating a rather stable subgroup that can be reached in several different ways along the tree. In this particular example we can recognize substructures that affirm the partial stability of the tree models.

Figure 10.13. A classification tree and its *trace plot*

The remaining instability in this particular example is in most cases given by the sequence in which the subgroups are separated. This is partially due to the fact that we are dealing with a multiclass problem; thus the reduction of impurity can be achieved by splitting off an arbitrary class or a group of classes. Nevertheless, our tree specimen from Fig. 10.13 is a rather rare one, as we see in the trace plot in Fig. 10.14, because its trace does not match with the main, opaque paths.

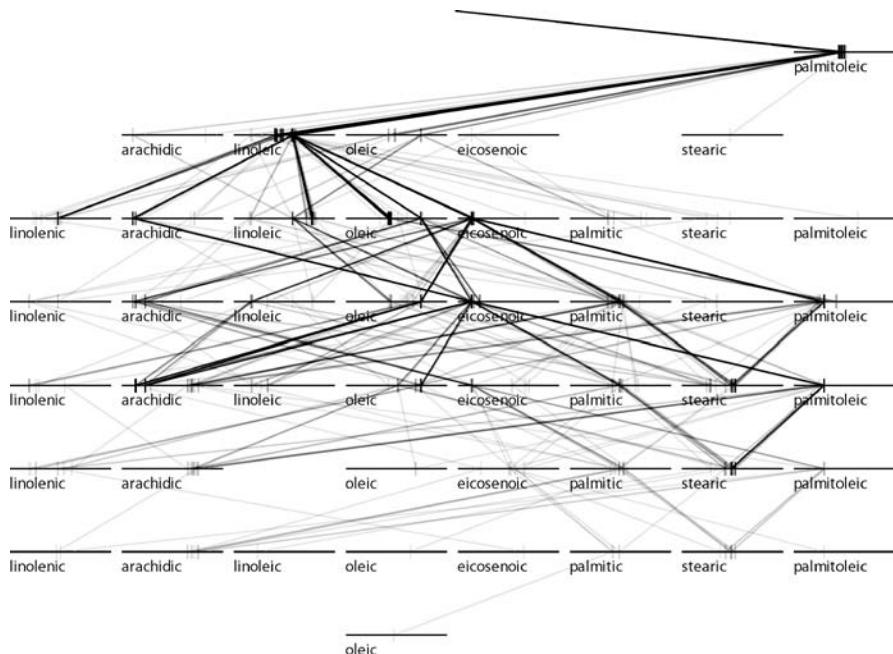


Figure 10.14. Trace plot of 100 bootstrapped trees

10.4

Conclusion

Tree models are very rich and versatile. Equally rich is the variety of possible visualization techniques that provide various views of trees, each time shedding light on different properties of the models.

Hierarchical views are the most commonly used graphical representations and highlight the sequence of splits. They are easy to interpret even by untrained personnel. Node placement and representation can convey additional information associated with the model or data. Size of nodes can be intuitively associated with the size of the data passed into that node. Highlighting and brushing is easily possible in this context, which facilitates interpretation in conjunction with available data. Hierarchical views often allow for additional annotation and supplemental information, such as split quality. Complemental methods are available for large trees and data, such as censored or context-preserving local zoom.

A less known group of tree model visualization techniques are those based on the recursive partitioning aspect. Direct view of the partition boundaries in the observation space can be obtained using *sectioned scatterplots*. The focus here lies on the cut points and their relative position in the data space. They are limited in terms of the number and types of covariates used but prove to be useful as a drill-down technique for local analysis of subgroups throughout the tree model.

Other methods based on recursive partitioning of the plot space are *treemaps* and *spineplots of leaves*. Both allow a concise view of all terminal nodes while retaining hints of the splitting sequence. In conjunction with highlighting and brushing, the main focus here is on the model behavior with respect to data points. As such the plots can be created using training and test data separately and compared. Treemaps are more suitable for absolute comparisons and large, complex trees, whereas spineplots of leaves can be used for relative comparison of groups within terminal nodes up to moderately complex trees.

Tree models are possibly unstable, that is, small changes in the data can lead to entirely different trees. To analyze the stability of splits it is possible to visualize the optimality criterion for candidate variables using *mountain plots*. Competing splits within a variable become clearly visible and the comparison of mountain plots of multiple candidate variables allows a quick assessment of the magnitude and cause for potential instability.

The instability of a tree model can be used to obtain additional insight in the data and to improve prediction accuracy. Bootstrapping provides a useful method for the analysis of model variation by creating a whole set of tree models. Visualization of the use of covariates in the splits as weighted barcharts with aggregate impurity criterion as weight allows quick assessment of variable importance. Variable masking can be detected using *weighted fluctuation diagrams* of variables and trees. This view is also useful for finding groups of related tree models.

Sectioned scatterplots also allow the visualization of partition boundaries for multiple trees. The resulting plot can no longer be used for global drill-down due to the lack of shared subgroups, but it provides a way of analyzing the “fuzziness” of a cut-point in conjunction with the data.

Finally, *trace plots* allow us to visualize split rules and the hierarchical structure of arbitrarily many trees in a single view. They are based on a grid of variables and tree levels (nodes of the same depth) where each cell corresponds to a candidate split variable, corresponding to a potential tree node. Actually used cells are connected in the same way as in the hierarchical view, thus reflecting the full structure of the tree. Multiple trees can be superimposed on this grid, each leaving its own “trace.” The resulting plot shows frequently used paths, common subgroups, and alternate splits.

All plots in this chapter have been produced using R software for statistical computing and KLIMT interactive software for visualization and analysis of trees and forests. Visualization methods presented in this chapter are suitable for both presentation of particular findings and exploratory work. The individual techniques complement each other well by providing various different viewpoints on the models and data. Therefore they can be successfully used in an interactive framework. Trace plots, for example, represent a very useful overview that can be linked to individual hierarchical views. Subgroups defined by cells in the trace plot can be linked to data-based plots, its edges to sectioned scatterplots.

The methods presented here were mostly illustrated on classification examples, but they can be equally used for regression trees and mostly for survival trees as well. Also, all methods described here are not limited to binary trees, even though those represent the most commonly used models. The variety of tree models and further

development of ensemble methods still leaves room for enhancements or new plots. For exploratory work it is of benefit to have a big toolbox to choose from; for presentation graphics it is important to have the ability to display the “key point” we want to convey.

References

- Breiman, L. (1996). Bagging predictors, *Machine Learn* 24(2):123–140.
- Breiman, L. (1999). Random forests – random features, *Technical Report TR567*, University of California-Berkeley, Statistics Department.
- Breiman, L., Friedman, J., Olshen, R. and Stone, C. (1984). *Classification and Regression Trees*, Wadsworth.
- Forina, M., Armanino, C., Lanteri, S. and Tiscornia, E. (1983). Classification of olive oils from their fatty acid composition, in Martens, H. and Russwurm, H. (eds), *Food Research and Data Analysis*, Applied Science, London, pp. 189–214.
- Harrison, D. and Rubinfeld, D. (1978). Hedonic prices and the demand for clean air, *J Environ Econ Manage* 5:81–102.
- Mangasarian, O. and Wolberg, W. (1990). Cancer diagnosis via linear programming, *SIAM News* 23(5):1–18.
- Ripley, B. (1996). *Pattern Recognition and Neural Networks*, Cambridge University Press, Cambridge, UK.
- Urbanek, S. (2003). Interactive construction and analysis of trees, *Proceedings of the 2003 Joint Statistical Meetings*, D.P. Mira (CD-ROM).

Part III

Methodologies

Interactive Linked Micromap Plots for the Display of Geographically Referenced Statistical Data

III.1

Jürgen Symanzik, Daniel B. Carr

1.1	<i>Introduction</i>	268
1.2	<i>A Motivational Example</i>	272
1.3	<i>Design Issues and Variations on Static Micromaps</i>	274
1.4	<i>Web-based Applications of LM Plots</i>	276
	Micromaps on the EPA CEP Web Site	278
	Micromaps on the USDA-NASS Web Site	278
	Micromaps on the NCI Web Site	279
	Micromaps at Utah State University	281
1.5	<i>Constructing LM Plots</i>	283
	Micromaps via S-Plus	283
	Micromaps via nViZn	286
	Micromaps via Java and Other Statistical Packages	287
1.6	<i>Discussion</i>	288

Introduction

Over the last decade, researchers have developed many improvements to make statistical graphics more accessible to the general public. These improvements include making statistical summaries more visual and providing more information at the same time. Research in this area involved converting statistical tables into plots (Carr, 1994; Carr and Nusser, 1995), new ways of displaying geographically referenced data (Carr et al., 1992), and, in particular, the development of linked micromap (LM) plots, often simply called micromaps (Carr and Pierson, 1996; Carr et al., 1998, 2000a). LM plots, initially called map row plots as well as linked map-attribute graphics, were first presented in a poster session sponsored by the American Statistical Association (ASA) Section on Statistical Graphics at the 1996 Joint Statistical Meetings (Olsen et al., 1996). More details on the history of LM plots and their connection to other research can be found in these early references on micromaps. More recent references on LM plots (Carr et al., 2000b; Carr, 2001) focused on their use for communicating summary data from health and environmental studies.

The basic idea behind LM plots is to link geographic region names and their values, as shown in quality statistical graphics such as row-labeled dotplots, with their locations, as shown in a sequence of small maps, called micromaps. This provides the opportunity to see patterns in a geospatial context as well as in the traditional statistical graphics context. Figure 1.1 shows a simple example of LM plots. This figure shows the 25 US states with the highest white female lung and bronchus cancer mortality rates for 2002. The states are sorted in descending order by the mortality rates and partitioned into groups of five to promote focused comparisons. The left-hand column consists of linked micromaps with one micromap for each group of five states. The top micromap has five regions with black outlines. Within each group of five states, the shade of grey fill for a region links to the shade of grey in the dot beside the region's name and to the shade of grey in the dot indicating the region's mortality rate. The same five shades of grey or distinct hues in color plots are links within each group of five states. The linking is basically horizontal within groups of five. The right column of Fig. 1.1 has familiar dotplot panels showing US state mortality rates estimates and 95 % confidence intervals. The data are from the US National Cancer Institute (NCI) Web site <http://www.statecancerprofiles.cancer.gov/micromaps>, further discussed in Sect. 1.4.3.

Figure 1.1 shows a useful variant of linked micromaps that accumulates states outlined in black. That is, states featured in previous groups of five are outlined in black and shown with a white fill or with a sixth distinctive hue in color versions. The black outline brings the outlined states into the foreground and creates a contour composed of state polygons. The bottom micromap contour includes states with values above 42 deaths per 100 000. While the political boundaries are less than ideal for accurately communicating geospatial patterns of local mortality rates, the progression of contours and the visible clusters in the bottom micromap are much more informative geospatially than the values in a table or a dotplot.

Lung and Bronchus Cancer: White Female Mortality 2002

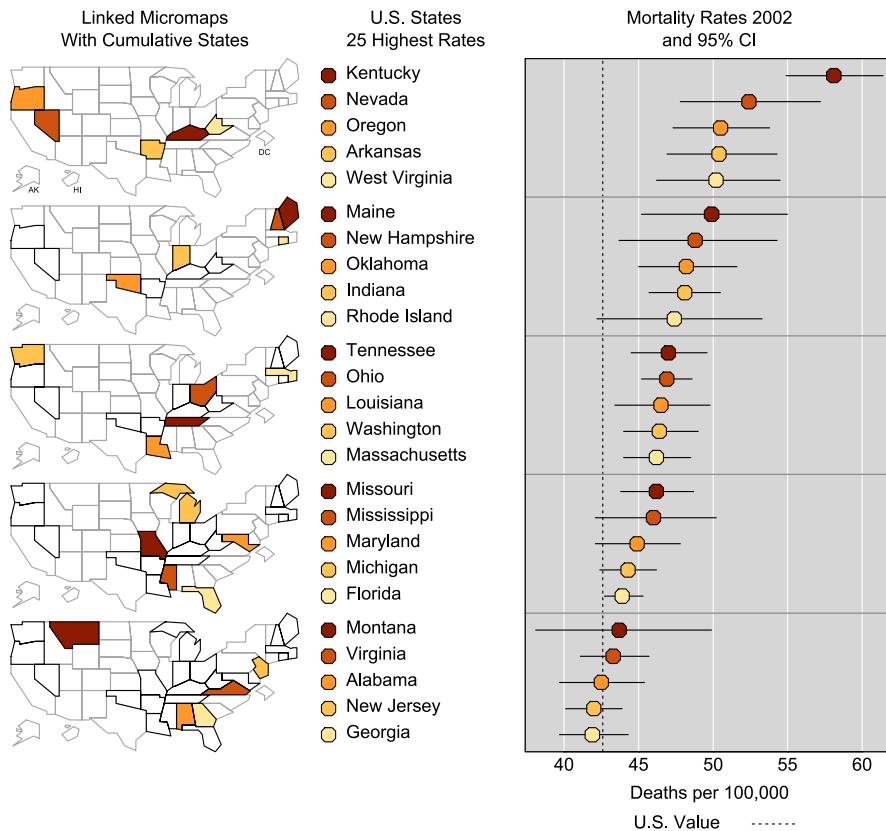


Figure 1.1. LM plots, based on data from <http://www.statecancerprofiles.cancer.gov/micromaps>, showing lung and bronchus cancer mortality rates for white females in 2002. The *left column* shows five micromaps, with five states highlighted in different colors in each of the maps. The same color information is used to link to the names of the US states in the *middle column* and to the data in the *right column*. This data column displays US state mortality rates estimates (*dots*) and 95 % confidence intervals (*lines*)

Many statistical graphics, such as dotplots, means with confidence bounds, boxplots, and scatterplots, start with a strong foundation toward quality by encoding information using position along a scale. Position along a scale encoding has a high perceptual accuracy of extraction (Cleveland and McGill, 1984). Quality statistical graphics often represent estimate uncertainty and reference values using position along scale encodings, provide grid lines to reduce distance effect problems in judging values against a scale, and follow the guidance of Tufte (1990) and others for visually layering information. Good visual layering makes the most important information the most salient. Estimates are generally more important than confidence

bounds and should be layered within the foreground. Grid lines always belong in the background. There are many factors related to statistical graphics quality, starting with content and context, and extending to additional visual considerations, such as perceptual grouping and sorting. While specific instances of LM plots may provide opportunities for improvement, the paradigm can incorporate much knowledge about quality statistical graphics design.

LM plots often provide a good alternative to displaying statistical information using choropleth maps. Choropleth maps use the color or shading of regions in a map to represent region values. Choropleth maps have proved very popular but have many problems and limitations as indicated by writers such as Robinson et al. (1978), Dent (1993), and Harris (1999). Reviewing these problems helps to indicate why LM plots are a good alternative.

There are two kinds of choropleth maps, called unclassed and classed. Unclassed maps use a continuous color scale to encode continuous values (statistics). This is problematic because perception of color is relative to neighboring colors and because color has poor perceptual accuracy of extraction in a continuous context. Classed choropleth maps ameliorate this problem and dominate in the literature.

Classed choropleth maps use class intervals to convert continuous estimates into an ordered variable with a few values that can be represented using a few colors. When a few colors are easily discriminated and regions are sufficiently large for color perception, color identification problems are minimal. The color scheme also needs to convey the class ordering based on values. Brewer (1997) and Brewer et al. (1997) provided results evaluating different color schemes in a mapping context. The Web site <http://colorbrewer.org> (see Leslie, 2002, for a short description) contains guidance on ordered color schemes and additional issues such as suitable schemes for people with color vision deficiencies and for different media. Perfect examples on how colors should be used in choropleth maps can be found in the 1996 "Atlas of United States Mortality" (Pickle et al., 1996).

Even with a good color scheme, three key problems remain for classed choropleth maps. The first problem relates to region area. As suggested above, some map regions can be too small to effectively show color. Examples include Washington, DC, on a map of the United States (US) and Luxembourg on a European map. Map caricatures, such as Monmonier's state visibility map (Monmonier, 1993), can address this problem, by enlarging small regions in a way that maintains region identifiability and shows each region touching the actual neighboring regions. Another facet of the area problem is that large areas have a strong visual impact while in many situations, such as in the mapping of mortality rates, the interpretation should be weighted by the region population. Dorling (1995) addressed this problem by constructing cartograms that changed region shapes to make areas proportional to population. Issues related to this approach are region identifiability, and map instability over time as their shapes change with changing populations. Area related problems persist in choropleth maps.

A second key problem is that converting a continuous variable into a variable with a few ordered values results in an immediate loss of information. This loss includes the relative ranks of regions whose distinct values become encoded with the same

value. The task of controlling conversion loss has spawned numerous papers about proposing methods for defining class intervals. Today, guidance is available based on usability studies. Brewer and Pickle (2002) indicated that quintile classes (roughly 20 % of regions in each class) tend to perform better than other class interval methods when evaluated across three different map-reading tasks. Still, the best of the class interval selection approaches loses information.

The third key problem is that it is difficult to show more than one variable in a choropleth map. MacEachren et al. (1995, 1998) were able to clearly communicate values of a second binary variable (and indicator of estimate reliability) by plotting black-and-white stripped texture on regions with uncertain estimates. However, more general attempts such as using bivariate colors schemes have been less successful (Wainer and Francolini, 1980). Thus, choropleth maps are not suitable for showing estimate standard errors and confidence bounds that result from the application of sound statistical sampling or description. It is possible to use more than one map to show additional variables. However, Monmonier (1996, p. 154) observed that when plotting choropleth maps side by side it can easily happen that "similarity among large areas can distort visual estimates of correlation by masking significant dissimilarity among small areas." The change blindness (Palmer, 1999, p. 538) that occurs as the human eyes jump from one map to another makes it difficult to become aware of all the differences that exist in multiple choropleth maps and hard to mentally integrate information in a multivariate context.

Carr proposed the use of micromaps and the LM plots design in response to Anthony R. Olsen's [from the US Environmental Protection Agency (EPA), Corvallis, OR] challenge to extend traditional statistical graphics called row-labeled plots in Carr (1994) to include geospatial context. As indicated earlier, traditional statistical graphics often use position along scale encodings for continuous values and can readily show estimates and confidence intervals together. Olsen, Carr, Courbois, and Pierson unveiled this new design with a 4 × 8 foot poster at 1996 JSM. The map regions were 78 Omernik ecoregions for the continental US. The various ecoregion barplots and boxplots summarized detailed elevation information and detailed land class information derived from multiple advanced high-resolution radiometer (AVHRR) remote sensing images over time.

The first example in the literature of LM plots (Carr and Pierson, 1996) showed state values for the US. This example adapted the state visibility map of Monmonier to address the visibility problems for small regions. That paper presented a micromap plot of unemployment by state with two data columns (unemployment rate with 95 % confidence interval and total number of unemployed persons). The LM plots paradigm supports the display of different kinds of statistical panels, such as dotplots, barplots, boxplots, binned scatterplots, time series plots, and plots with confidence bounds. In particular, Carr et al. (1998) presented three micromap plots: (i) CO₂ emissions in the Organization for Economic Cooperation and Development (OECD) states with one data column (annual time series), (ii) wheat prices by state with two data columns (average price and monthly time series), and (iii) an improved version of the micromap plot from Carr and Pierson (1996). Carr et al. (2000a) presented four micromap plots based on the Omernik ecoregions: (i) three data columns with

boxplots, (ii) bivariate binned boxplots, (iii) time series plots, and (iv) line height plots. The level 2 Omernik ecoregion micromaps involved large US regions, so region visibility was not a problem. However, highly detailed boundaries lead to a slow graphics production. Faster production in the context of more numerous ecoregions and over 3000 US counties as opposed to 50 US states also motivated the development of generalized micromaps. Geographers traditionally use the best encoding, position along a scale, to show region boundaries. However, they also desire to show statistics more precisely, and thus static LM plots soon appeared in the geographic literature (Fonseca and Wong, 2000) as well.

In this chapter, we continue with a motivational example in Sect. 1.2 that shows the same data via choropleth maps and LM plots. In Sect. 1.3, we discuss design issues for LM plots and outline their main purposes. All of the early examples of LM plots were created as static plots to be displayed as printed pages or as large posters. In Sect. 1.4 we discuss how micromaps can be used interactively on the Web. We return to production resources for static LM plots via statistical software packages in Sect. 1.5. We finish with a discussion and comparison of LM plots with other graphical tools in Sect. 1.6.

1.2

A Motivational Example

Figure 1.2 shows two variables, the soybean yield and acreage from the 1997 Census of Agriculture for the US, displayed in two choropleth maps. Five equal-size class intervals were chosen for each of the maps. A “6-class sequential2 Greys” color scheme, obtained from <http://colorbrewer.org> and reprinted in Table 1.1, was chosen, with the lightest grey representing states where no soybeans were planted. The maps in this figure were produced with the Geographic Information System (GIS) ArcView 3.2. We obtained the data from the US Department of Agriculture–National Agricultural Statistics Service (USDA–NASS) Web site <http://www.nass.usda.gov/research/apydata/soyapy.dat>. Section 1.4.2 provides further details on these data and the USDA–NASS Web site.

The two choropleth maps in Fig. 1.2 indicate that the highest yields and highest acreages for soybeans occur in the Midwest. There seems to be some spatial trend, i.e., some steady decrease for both variables from the Midwest to the Southeast. Over-

Value 1	Value 2	Value 3
0	0	247
0	0	217
0	0	189
0	0	150
0	0	99
0	0	37

Table 1.1. Values for the “6-class sequential2 Greys” color scheme, obtained from <http://colorbrewer.org>, for use in ArcView 3.2. Instead of breaking down the range of possible values (0 to 255) into equally wide intervals, the chosen values represent similar perceptual differences. The first triple (0, 0, 247) represents the lightest grey, the last triple (0, 0, 37) the darkest grey. When represented as red, green, and blue (RGB) values, the zeros will be replaced by the nonzero value, i.e., (0, 0, 247) will become (247, 247, 247) and so on

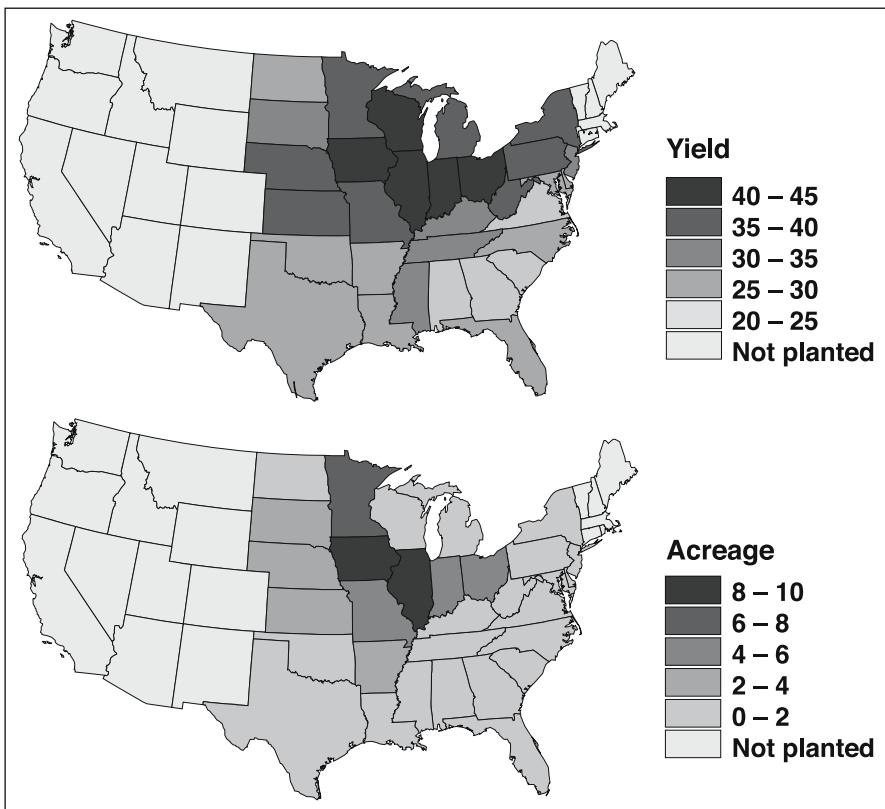


Figure 1.2. Choropleth maps of the 1997 Census of Agriculture, showing the variables soybean yield (in bushels per acre) and acreage (in millions of acres) by state. The data represent the 31 US states where soybeans were planted

all, there appears to be a positive correlation between these two variables since high yields/high acreages and low yields/low acreages seem to appear in the same geographic regions. The correlation coefficient between yield and acreage is only 0.64, suggesting departures from linearity that would be better revealed using scatterplots or LM plots.

The LM plots paradigm sorts the list of regions based on their values or names and partitions the list into perceptual groups of size five or less, which is further discussed in Sect. 1.3. The micromap design assigns a distinct color to each region in a group. The same color is used for plotting the region polygon in the micromap, the dot by the region name, and the region values in one or multiple statistical panels. Figure 1.3 provides a LM plot example with 31 US states and shows three statistical panels with dotplots for three variables: yield, acreage, and production. This example is accessible at <http://www.nass.usda.gov/research/gmsoyyap.htm>. In fact, Fig. 1.3 shows the LM plots of the same two variables as Fig. 1.2, plus a third statistical panel for the

variable production. Data are available for 31 of the 50 US states only. An identical color links all of the descriptors for a region. Successive perceptual groups use the same set of distinct colors. In Fig. 1.3, the sorting is done (from largest to smallest) by soybean yield in those 31 US states where soybeans were planted. Here, the points within a panel are connected to guide the viewer's eyes and not to imply that interpolation is permitted. The connecting lines are a design option and can be omitted to avoid controversy or suit personal preferences. The list of 31 US states is not evenly divisible by five. Two perceptual groups at the top and two groups at the bottom contain four states, while three perceptual groups in the middle contain five states. The middle groups require the use of a fifth linking color. Using distinct hues for linking colors works best in full-color plots. For grey-level plots, colors need to be distinct in terms of grey level. Figure 1.3 shows different shades of green and is suitable for production as a grey-level plot. Readers new to LM plots sometimes try to compare regions with the same color across the different perceptual groups, but they quickly learn the linkage is meaningful only within a perceptual group.

While the micromaps in Fig. 1.3 are not ideally suited to show geospatial patterns, the statistical panels nevertheless are more revealing than the choropleth maps in Fig. 1.2. It is immediately obvious from this graphical display that there is a positive correlation between the two variables yield and acreage (high values for yield are associated with high values for acreage while low values for yield are associated with low values for acreage). However, there were some considerable spatial outliers that could not be detected easily in the two choropleth maps in Fig. 1.2. Wisconsin (in a favorable geographic region for soybeans, neighboring Iowa, Illinois, and Indiana) had a high yield, but only a very small acreage was used for soybeans. On the other hand, Arkansas, with a relatively low yield (similar to its neighboring states in the Central South), used a surprisingly high acreage for soybeans. Geographically, the highest yields for soybeans were obtained in the Midwest, with a spatially declining trend toward the Southeast. When visually comparing acreage and production, these two variables almost perfectly correlate. Overall, the LM plots in Fig. 1.3 provide a much better and simpler geographic reference to the underlying data than the two choropleth maps in Fig. 1.2.

1.3 Design Issues and Variations on Static Micromaps

Figure 1.3 shows one possible partitioning of 31 regions into perceptual groups of size five or less. However, for different numbers of regions there may exist more than just one meaningful partitioning. Table 1.2 shows different ways to partition regions into groups of size five or less. In fact, the partitioning in Fig. 1.3 is called Partitioning 2 in Table 1.2. An algorithm, coded in S-Plus (Becker et al., 1988), produces symmetry about the middle panel or the pair of middle panels (Partitioning 1). It puts small counts in the middle. For complete data situations, a single region appearing in the

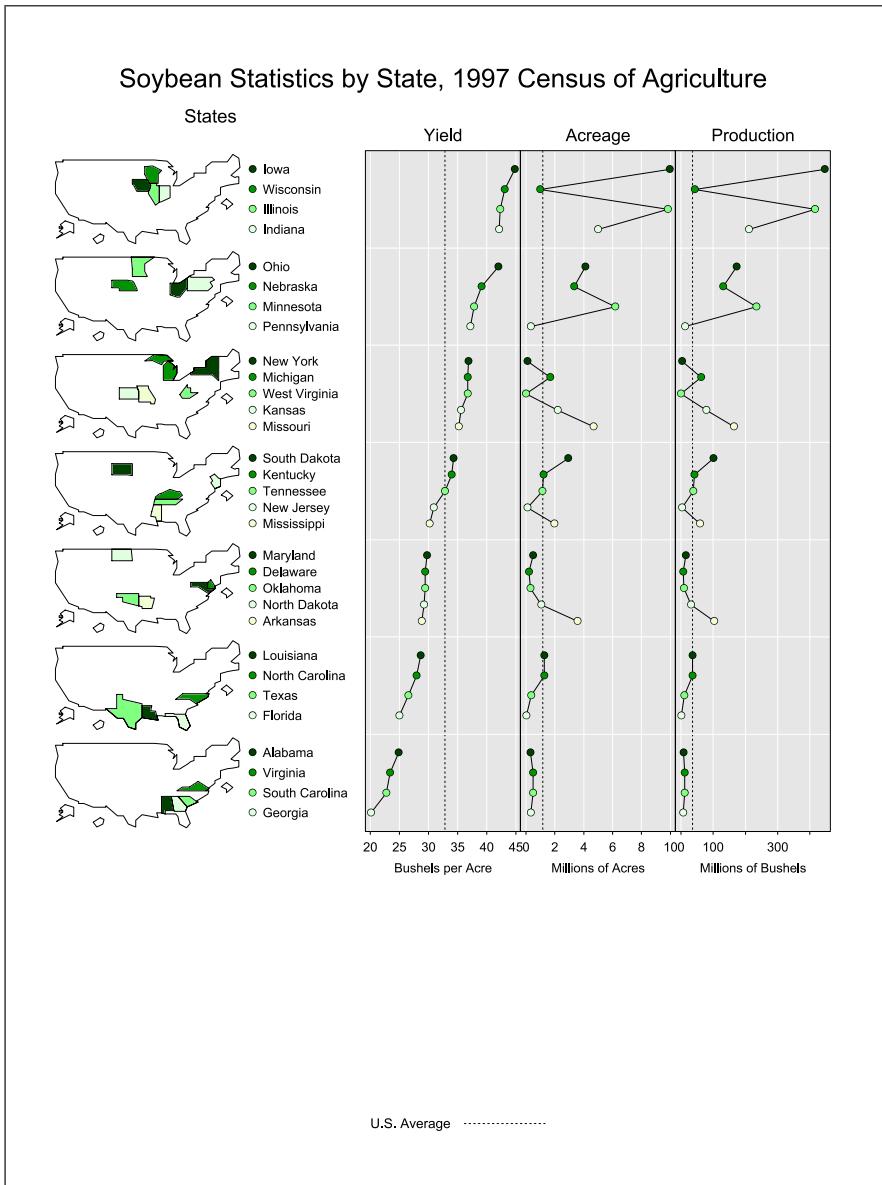


Figure 1.3. [This figure also appears in the color insert.] LM plots of the 1997 Census of Agriculture, showing soybean yield (in bushels per acre), acreage (in millions of acres), and production (in millions of bushels) by state. The data are sorted by yield and show the 31 US states where soybeans were planted. The “US Average” represents the median, i.e., the value that splits the data in half such that half of the states have values below the median and the other half of the states have values above the median. For example, Tennessee is the state with the median yield. This figure has been republished from <http://www.nass.usda.gov/research/gmsoyyp.htm> without any modifications (and ideally should contain much less white space in the lower part)

middle panel shows the median value for the variable used in sorting. Carr et al. (1998) exploit this in a special design for the 50 US states plus Washington, DC. In other situations, it might be preferable to avoid creating groups with few regions. The right column (Partitioning 2) in Table 1.2 is also symmetric, but it avoids small counts where possible. The visual appeal of symmetry is lost when there are too many groups to display in one page or one screen image. In this case, it suffices to fill all panels except a single panel that contains the leftover regions modulo the grouping size. It does not seem to matter whether the partially filled panel appears at the top, near the middle, or at the bottom of the page.

The micromaps in LM plots are intended to be caricatures that generally serve three purposes. The first purpose is to provide a recognizable map. Boundary details are not important here. Boundaries only need to communicate region identity and neighbor relationships.

The second purpose is to provide a visible color link. Tiny areas do not support easy color identification. As indicated above, early micromaps for states adapted the hand-drawn state visibility map of Monmonier (1993). This map simplified boundaries and increased the size of small US states such as Rhode Island. The EPA Cumulative Exposure Project (CEP) Web site (Symanzik et al., 2000) made it necessary to develop an algorithm to generalize boundaries for US states and US counties. Most of the Web-based micromaps described in Sect. 1.4 use these generalized boundaries for US states and US counties. New projects may entail the production of map caricatures or generalizations when the available boundaries are not suitable.

The third purpose of micromaps is to show geospatial patterns. Figure 1.3 shows individual perceptual groups of states. This simple version is good as an introduction and reveals some geospatial grouping. More sophisticated variations on micromaps have also called attention to contours constructed from region polygons. One approach shows background states in light grey with white outlines. States highlighted previously in a perceptual group can appear in an additional color such as light yellow and have a black outline. The nongrey black-outlined states then appear as a foreground contour. Examples with two complementary contours for regions above and below the median will be further discussed in Sects. 1.4.3 and 1.5.1.

1.4

Web-based Applications of LM Plots

Over the last decade, US Federal Agencies and other institutions have increasingly focused attention on distributing large amounts of geographically referenced statistical data, either in print or through the Web. The Web-based distribution of data is aimed at replacing printed tabular displays and at providing access to current data quickly. Several approaches have been developed that provide a user-friendly Web-based interface to tabular and graphical displays of federal data. The user can interactively and dynamically query and sort the data, compare different geographic regions, and look at the data at different spatial resolutions, e.g., at the state or the county level.

Table 1.2. Full symmetry partitionings with targeting groups of size 5. The left column (#) contains the number of regions. The middle column (Partitioning 1) puts the smallest counts in the middle. Full-symmetry alternatives that avoid small counts appear in the right column (Partitioning 2). Abandoning full symmetry can lead to fewer panels. The table ends with 51 regions (the number of US states plus Washington, DC), but it can be easily extended

#	Partitioning 1	Partitioning 2
1	1	
2	2	
3	3	
4	4	
5	5	
6	3 3	
7	3 1 3	2 3 2
8	4 4	
9	4 1 4	3 3 3
10	5 5	
11	5 1 5	3 5 3
12	5 2 5	4 4 4
13	5 3 5	4 5 4
14	5 4 5	
15	5 5 5	
16	5 3 3 5	4 4 4 4
17	5 3 1 3 5	3 4 3 4 3
18	5 4 4 5	4 5 5 4
19	5 4 1 4 5	4 4 3 4 4
20	5 5 5 5	
21	5 5 1 5 5	4 4 5 4 4
22	5 5 2 5 5	5 4 4 4 5
23	5 5 3 5 5	
24	5 5 4 5 5	
25	5 5 5 5 5	
26	5 5 3 3 5 5	5 4 4 4 4 5
27	5 5 3 1 3 5 5	4 4 4 3 4 4 4
28	5 5 4 4 5 5	4 5 5 5 5 4
29	5 5 4 1 4 5 5	4 4 4 5 4 4 4
30	5 5 5 5 5 5	
31	5 5 5 1 5 5 5	4 4 5 5 5 4 4
32	5 5 5 2 5 5 5	5 5 4 4 4 5 5
33	5 5 5 3 5 5 5	4 5 5 5 5 5 4
34	5 5 5 4 5 5 5	
35	5 5 5 5 5 5 5	
36	5 5 5 3 3 5 5 5	4 4 5 5 5 5 4 4
37	5 5 5 3 1 3 5 5 5	4 4 4 4 5 4 4 4 4
38	5 5 5 4 4 5 5 5	4 5 5 5 5 5 5 4
39	5 5 5 4 1 4 5 5 5	4 4 4 5 5 5 4 4 4
40	5 5 5 5 5 5 5 5	
41	5 5 5 5 1 5 5 5 5	4 4 5 5 5 5 4 4
42	5 5 5 5 2 5 5 5 5	5 5 5 4 4 4 5 5 5
43	5 5 5 5 3 5 5 5 5	4 5 5 5 5 5 5 5 4
44	5 5 5 5 4 5 5 5 5	
45	5 5 5 5 5 5 5 5	
46	5 5 5 5 3 3 5 5 5 5	4 4 5 5 5 5 5 4 4
47	5 5 5 5 3 1 3 5 5 5	4 4 4 4 5 5 5 4 4 4
48	5 5 5 5 4 4 5 5 5 5	4 5 5 5 5 5 5 5 5 4
49	5 5 5 5 4 1 4 5 5 5	4 4 4 5 5 5 5 4 4 4
50	5 5 5 5 5 5 5 5 5	
51	5 5 5 5 1 5 5 5 5	4 4 5 5 5 5 5 4 4

Carr and Olsen (1996) provide examples on the visual appearance of patterns in data when properly sorted.

The direction of LM plot development shifted from static LM plots toward interactive micromap displays for the Web. Work done for the EPA CEP Web site (Symanzik et al., 2000) was the first in this regard. This project was soon followed by Web-based examples of micromaps produced by the USDA–NASS such as in Fig. 1.3.

The Digital Government (dg.o) initiative (<http://www.diggov.org>) is a major research initiative funded by the National Science Foundation (NSF) and several federal agencies such as the EPA, the USDA–NASS, the US Census Bureau, the NCI, the US Bureau of Labor Statistics (BLS), etc. This initiative addresses multiple aspects related to federal data such as visualization, access, disclosure, security, etc. One of the proposals funded under dg.o was the Digital Government Quality Graphics (DGQG) project that included the development of LM plots (<http://www.geovista.psu.edu/grants/dg-qg/index.html>).

In the remainder of this section, we look at four main applications of interactive Web-based LM plots, three of them on federal Web sites. A short overview of interactive micromaps, as well as a micromap of the “Places” data (Boyer and Savageau, 1981), can be found in Symanzik (2004). However, additional details are given in this section.

1.4.1

Micromaps on the EPA CEP Web Site

The idea of using micromaps on the Web was first considered for the EPA CEP Web site (previously accessible at <http://www.epa.gov/CumulativeExposure/>). Initially, the EPA wanted to provide fast and convenient Web-based access to its hazardous air pollutant (HAP) data for 1990. In this dataset, concentrations of 148 air pollutants were estimated for each of the 60 803 US census tracts in the 48 contiguous US states (Rosenbaum et al., 1999). The EPA Web site was designed to allow the user to easily move through the dataset to find information on different air pollutants at different geographical locations and at different levels of geographic resolution (e.g., state, county, census tract) via interactive tables and micromaps. Unfortunately, no part of the interactive CEP Web site was ever published due to concerns that the 1990 data were outdated on the intended release date in 1998. Only a static version of the CEP Web site without tables and micromaps was accessible for several years. More details on the work related to the planned interactive CEP Web site can be found in Symanzik et al. (1999a,b, 2000).

1.4.2

Micromaps on the USDA–NASS Web Site

The USDA–NASS Research and Development Division released a Web site (<http://www.nass.usda.gov/research/sumpant.htm>) in September 1999 that uses interactive micromaps to display data from the 1997 Census of Agriculture. The USDA–NASS Web site displays acreage, production, and yield of harvested cropland for corn, soybeans, wheat, hay, and cotton.

A user of this Web site can sort the states by acreage or by yield with respect to a selected crop. Figure 1.3, already discussed in more detail in Sect. 1.2, shows LM plots from this Web site for soybeans, with states sorted by decreasing yield. It is possible to select another crop type and to access and download the raw data for further analysis or additional maps such as the choropleth maps in Fig. 1.2 that are based on these data. While a user who accesses this Web site gets the impression of full interactivity, this is not the case. The ten micromaps (5 crops \times 2 arrangements) plus one overview micromap were precalculated in S-Plus (see Sect. 1.5.1 for more details) and were stored as jpeg images. It is not possible to create any new micromap display “on the fly” on this Web site. Precalculating all possible micromaps is often not possible or desirable for all datasets, as we will see in the next section.

Micromaps on the NCI Web Site

1.4.3

The NCI released the State Cancer Profiles Web site in April 2003 that provides interactive access to its cancer data via micromaps. This Web site is Java based and creates micromaps “on the fly.” Wang et al. (2002) and Carr et al. (2002) provide more details on the design of the NCI Web site, <http://www.statecancerprofiles.cancer.gov/micromaps>.

LM plots with interactive and dynamic features are the main content on NCI’s State Cancer Profiles Web site that combines the latest research in data visualization sponsored by the NSF DGQG initiative with features to ensure accessibility by the visually impaired. The State Cancer Profiles Web site provides a “quick stop” for cancer-related statistics for planners, policymakers, and epidemiologists. It was developed by the NCI in collaboration with the Centers for Disease Control and Prevention (CDC) and is an integral component of NCI’s Cancer Control PLANET, a Web portal that links to resources for comprehensive cancer control. The Web site provides national (US), state, and county views of cancer statistics collected and analyzed in support of annual federal reports. The focus is on eight major cancer types for which there is evidence of the potential to improve outcomes either by prevention or by screening and treatment.

Cancer statistics include mortality and incidence counts, rates, and trends by sex and race/ethnicity. Recent incidence data are available for cancer registries participating in CDC’s National Program of Cancer Registries (NPCR) that met selected eligibility criteria. Both historical and recent incidence data are available for cancer registries participating in NCI’s Surveillance, Epidemiology and End Results (SEER) program. Prevalence of risk factors and screening, Healthy People 2010 US objectives, and demographics complete the profiles. The interactive graphic capabilities allow a user of the Web site to quickly explore patterns and potential disparities. For example, the user can easily compare graphs of national (US) or state trends for Whites, Blacks, Hispanics, Asian or Pacific Islanders, and American Indian/Alaskan Natives. LM plots provide the primary graphic template for users to explore spatial relationships among the latest rates, percents, and counts for cancer statistics, demographics, risk factors, and screening.

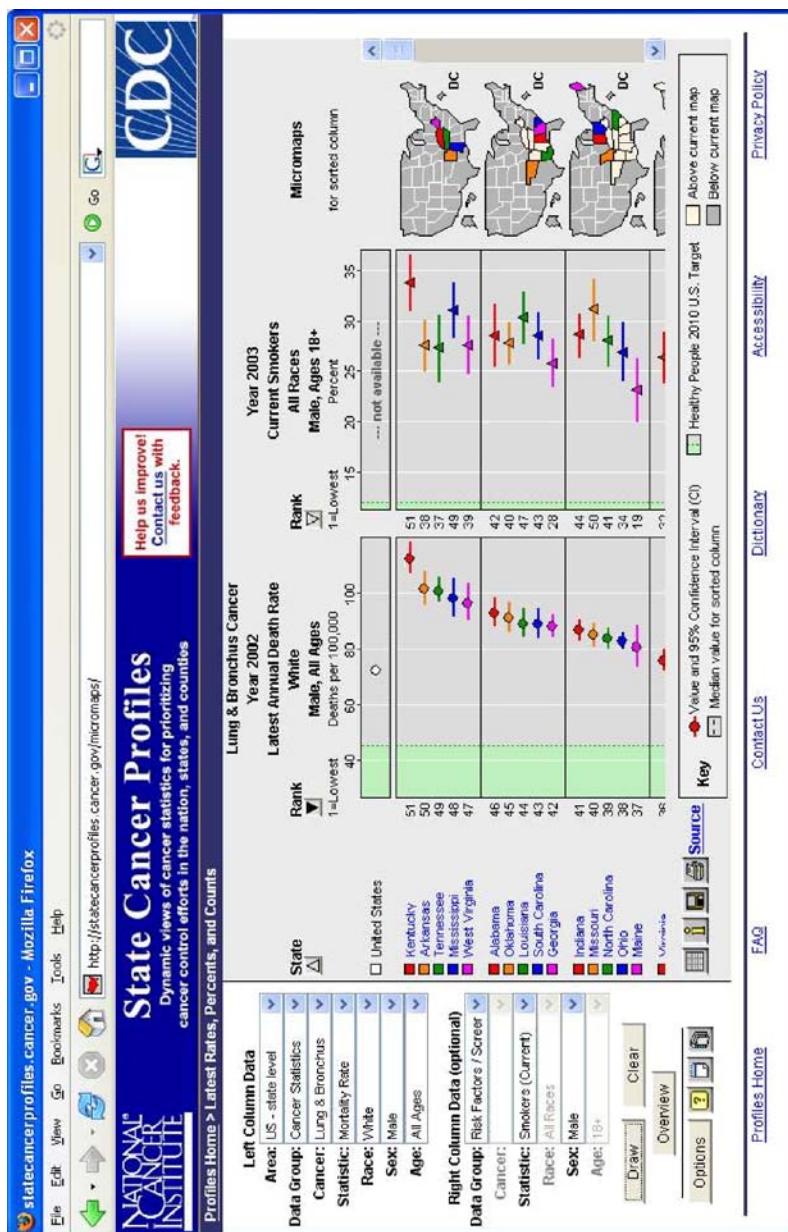


Figure 1.4. LM plots from the NCI Web page, showing lung and bronchus cancer for the year 2002. This interactive version of micromaps is accessible at <http://www.statecancerprofiles.cancer.gov/micromaps/>.

Figure 1.4, a screenshot from the NCI Web site, shows LM plots with state names, lung and bronchus cancer mortality rates, current smoking rates, and micromaps. This figure illustrates some design changes (e.g., variable selection on the left and maps on the right) that are specific to the NCI micromap implementation. The major design challenge for this Web site arose from taking LM plots from a portrait print image to a landscape screen image. The change in orientation significantly affected the number of rows that could be displayed. The requirement for a standard banner and logos for branding exacerbated the problem by further reducing the vertical screen real estate. Since only a few panels could simultaneously appear in view, the revised design provides scrolling within the Java applet that keeps the variable scales and other context information in view. In the current design, there is no scrollbar for the browser window if the screen area setting is for 1024×768 pixels or more. Usability studies identified many health care planners using a 800×600 pixel resolution. A notice now suggests increasing the screen resolution to avoid this problem.

The main features of the NCI micromap Web site are as follows. The scrollbar to the right of the micromaps in Fig. 1.4 controls a small set of statistical and geographical panels, leaving the reference information above and below the statistical and micromap displays in place. The default for the user controllable placement of micromaps puts them by the scrollbar in the right for easy scanning of contours revealed by the cumulative micromaps. Clicking a triangle above one of the columns specifies the sort order that determines how the regions accumulate. The panels often show Healthy People 2010 US target intervals (the green region), estimates, and confidence intervals.

Moving the mouse over an estimate shows the underlying numerical values. Moreover, moving the mouse over any of the linked items (region name, graph glyphs, and map region) causes these items to blink. Clicking on a state name invokes a drill-down to show the counties of the selected state.

Due to the interest in region rankings, the statistical panels have been augmented with numbers indicating the rank order. The interpretation of “1” is explicitly labeled since usability assessment found people who interpret “1” as best.

The color selection options for this Web site include one color scheme suitable for the two most common kinds of color vision deficiencies. There are additional items including popup overviews that augment the LM plots.

Micromaps at Utah State University

1.4.4

Micromaps and other graphical displays were found to be very useful for the display and analysis of the geographic spread of the West Nile Virus (WNV) and other diseases (Symanzik et al., 2003) across the US. For this reason, researchers at Utah State University (USU) obtained the NCI Java micromap code and adapted it for the display of WNV data (Chapala, 2005). Similar to the NCI micromap application, a user can now select among WNV infection rates and counts, and death rates and counts, starting with the WNV data for the US for the 2002 season. A drill-down into US counties is possible given that data at the county level are available. New features of the USU Web site include the plotting of the data for 2 years side by side in one

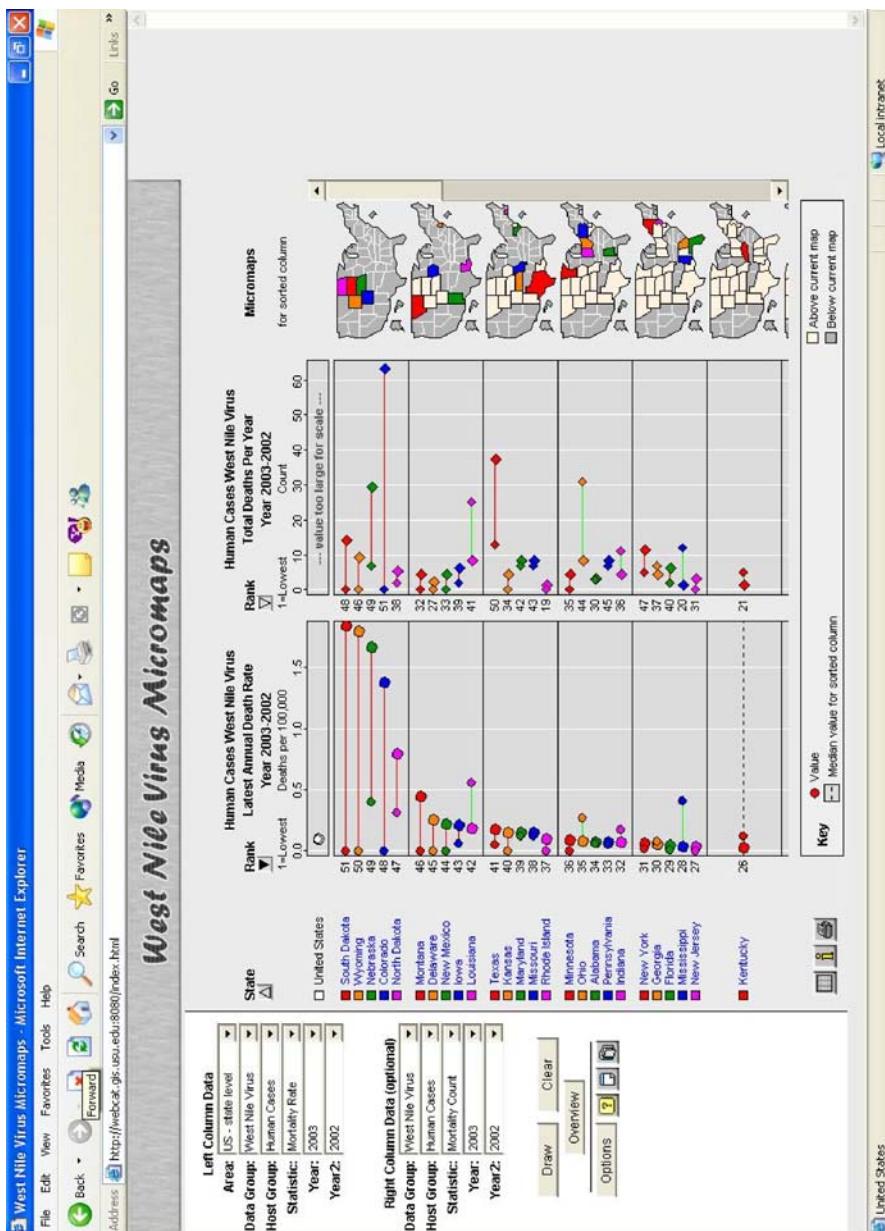


Figure 1.5. LM plots from the USU WNV Web page, showing WNV death rates and death counts for the years 2002 (small dots) and 2003 (big dots). Red lines indicate an increase and green lines a decrease from 2002 to 2003. This interactive version of micromaps is accessible at <http://webcat.gis.usu.edu:8080/index.html>

panel and additional sorting criteria such as sorting from the highest increase over no change to highest decrease in two selected years. The USU WNV micromap Web site can be accessed at <http://webcat.gis.usu.edu:8080/index.html>. Figure 1.5 shows WNV death rates (left data panel) and death counts (right data panel) for the years 2002 and 2003. The data are sorted by highest WNV death rate in 2003. A big dot represents the year 2003, a small dot the year 2002. Red lines indicate an increase in the rates/counts from 2002 to 2003 in a state, while green lines indicate a decrease in the rates/counts from 2002 to 2003. A strong geospatial pattern can be observed, with highest WNV death rates in 2003 in the Midwestern states. While death rates (and counts) considerably increased from 2002 to 2003 for most of the states that are currently visible in this scrollable micromap, the opposite holds for some of the Central states such as Louisiana, Ohio, Mississippi, and Kentucky.

Constructing LM Plots

1.5

While the previous section describes major Web sites that provide access to their geographically referenced statistical data via LM plots, this section summarizes how individual micromap plots can be constructed. This can be done using the statistical software package S-Plus, by providing appropriate data and map files to an application based on the software development kit (SDK) nViZn (Wilkinson et al., 2000), or by using recent Java code developed at the NCI. In particular, when high-resolution LM plots in postscript (.ps) format are required and a screenshot from a Web site or one of the Java applications is not sufficient, a user most likely will have to use existing S-Plus code or will have to make changes to this existing code.

Micromaps via S-Plus

1.5.1

Individual LM plots can best be created via S-Plus. Sample S-Plus code, data files, and resulting plots can be obtained from Dan Carr's micromaps Web site at <http://www.galaxy.gmu.edu/~dcarr/micromaps>. Included are files related to the early micromap articles (Carr and Pierson, 1996; Carr et al., 1998) as well as a frontend to S-Plus that allows the user to create time series micromap plots and dotplot micromap plots. In fact, the micromaps that are accessible at the USDA-NASS Web site were created by USDA-NASS personnel using these tools.

S-Plus functions in the rStateplot library, boundary files, data, and an example S-Plus script file are also available at Dan Carr's micromaps Web site introduced above. This library supports dotplots, dotplots with confidence bounds, barplots, arrow plots to show change, and boxplots based on county values. Boundary files are available for US states and US counties. The datasets are cancer mortality rates from the NCI. The script file addresses how to obtain more mortality rates over the Web as well as examples showing different kinds of plots. More general S-Plus functions for other regions and for times series that may need panel-specific scaling are available from the second author.

It seems to be possible to adapt the S-Plus micromap code for use in the R statistical package (Ihaka and Gentleman, 1996), which can be freely obtained from <http://www.r-project.org/>. Although no full implementation of a micromap library in R exists at this point, the basic panel functions and simple examples of LM plots have been converted from S-Plus to R by Anthony R. Olsen.

Figure 1.6 is based on data from the NCI micromap Web site and illustrates several variations of LM plots, implemented via the S-Plus *r1Stateplot* library. This example uses a special layout for 51 regions that differs slightly from the two suggested partitionings in Table 1.2 and calls attention to the middle state after sorting the states based on one of the variables displayed. Figure 1.6 bases the sorting on white male lung and bronchus cancer mortality rates during the years 1950 to 1969. Vermont is the middle (26th) state in the sorted order and thus has the median value. Instead of showing eleven panels with micromaps as suggested by both partitionings in Table 1.2, the micromap panel layout shows the middle state as a sixth highlighted state in the micromaps immediately above and below the median divider. This layout calls attention to symmetry and saves space by removing the need for an eleventh micromap. The US federal agencies produce so many graphics and tables for the 50 US states plus Washington, DC that it is worthwhile to develop a special layout for routine use. Other situations such as producing LM plots for the major regions of other countries may benefit from developing special layouts. Carr (2001) considered special layouts for the counties of selected states and made first use of the S-Plus functions assembled in the *r1Stateplot* library. This library was written for students to produce LM plots such as in Fig. 1.6.

The micromap construction in Fig. 1.6 introduces another useful design feature. It accumulates states when moving up from the bottom panel or down from the top panel toward the median divider. After highlighting five states in a micromap panel with foreground colors and black outlines, this design continues to show the previously highlighted states with black outlines in panels closer to the median divider. This black outline lifts the states into the foreground, and the micromap progression shows a sequence of foreground contours. A cluster of states from the Northwest and much of the inland US emerges when moving up from the bottom panel. The micromaps immediately above and below the median divider are duals of each other except for Vermont, which is in the foreground of both of these micromaps. Human perception does not treat foreground and background equally, so the logical equivalence of the two panels may not be noticed. The micromap just above the median divider calls attention to most of the US border states other than those in the Northwest.

Full-color LM plots typically use distinct saturated hues to highlight the selected regions in each panel. The nonhighlighted black-outlined regions are shown in a desaturated hue such as light yellow while the background regions are light grey with white outlines. The grey-level representation of Fig. 1.6 makes it harder to distinguish between foreground (black-outlined) states with the lightest colors and the near-white states highlighted in panels further from the median divider. However, the reader can still easily look at one of the two micromaps by the median divider, note whether or not a particular state is in the foreground, based on the black outline, and know which direction to scan in to find the panel where the state is highlighted.

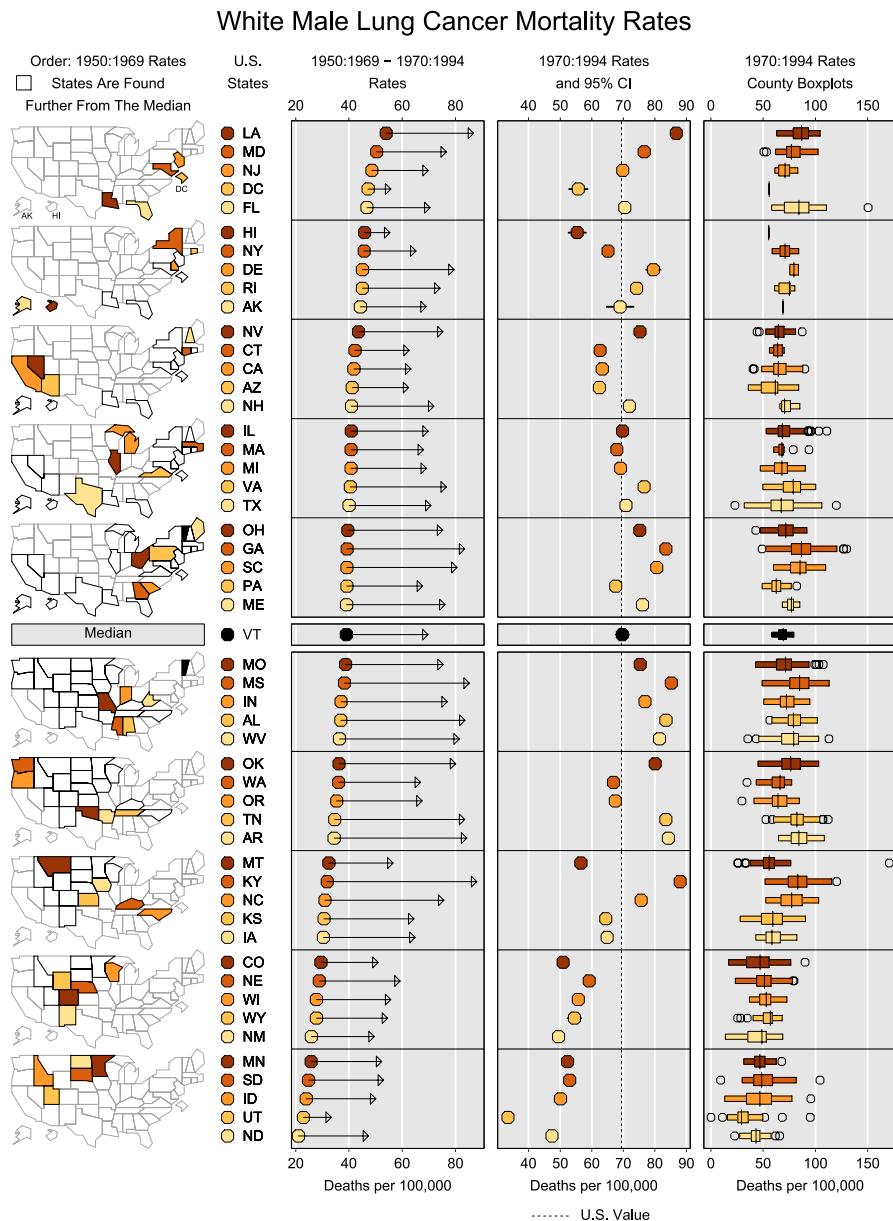


Figure 1.6. [This figure also appears in the color insert.] LM plots, based on data from the NCI Web page, showing summary values for the years 1950 to 1969 and for the years 1970 to 1994 in the *left data panel*, rates and 95 % confidence intervals in the *middle data panel*, and boxplots for each of the counties of each state in the *right data panel*

The Fig. 1.6 approach of accumulating foreground regions from the ends toward the center is one of several micromap variations tried over the years. Accumulating regions from one end to the other has been popular and is easier to explain. One drawback is that micromaps look more complicated when more than half of the regions appear in the foreground. A second drawback is that the chances increase of a background region being completely surrounded by outlined states. This will cause the background region to appear in the foreground. The region color fill will clarify but may not be noticed. Usability studies could help in ranking the various options.

Figure 1.6 illustrates different kinds of statistical panels. The first statistical panel shows summary values for the 20-year period 1950 to 1969 as filled dots. It shows the values for the 25-year period from 1970 to 1994 as arrow tips. Both encodings are position along scale encodings. Arrow length encodes the difference in rates for the two time intervals. Length is a good encoding in terms of perceptual accuracy of extraction. The huge increase in mortality rates for all US states is obvious.

The second statistical panel in Fig. 1.6 shows rate estimates and 95 % confidence intervals for the 25-year period from 1970 to 1994. Typically, the confidence intervals are considered secondary information and the rates are plotted on top. In Fig. 1.6, this totally obscures the confidence intervals except for Alaska, Hawaii, and Washington, DC. There are various remedies, such as showing 99 % confidence intervals. Our purpose in Fig. 1.6 is to call attention to how well the state rates are known and how little this conveys about the geospatial variation within the states.

The third and rightmost statistical panel in Fig. 1.6 shows boxplots of the rate estimates for the 25-year period from 1970 to 1994 for the counties of each US state. The outliers appear as open circles. The geospatial variation based on 25 years of data is substantial. Note that the scale has changed from the panel with state rates and confidence intervals. Using the county scale for both columns would better convey the county variability. Of course, the 25-year state rate estimates are also hiding variation over time.

In recent years, US federal agencies have placed increasing emphasis on confidentiality. The suppression of data is increasingly common. One approach toward making something available to the public has relied on aggregation to obscure details. This leads to aggregation over important factors such as time, geospatial regions, race, and sex. There is currently a serious consideration for suppressing all county-level mortality-rate estimates. Suppression of data is a problem for the public concerned about human health. There are additional issues related to data that is not collected. For example, data are not collected on cigarette smoking in terms of packs per day at the county level.

1.5.2

Micromaps via nViZn

nViZn (Wilkinson et al., 2000) (read en vision) is a Java-based SDK, developed and distributed by SPSS (<http://www.spss.com/visualization/services/>). nViZn was inspired by the idea of building on the BLS Graphics Production Library (GPL), described in Carr et al. (1996), with a formal grammar for the specification of statistical graphics (Wilkinson, 1999). nViZn was created as a distinct product whose wide

range of capabilities includes creating interactive tables and linked micromaps. Experiences with nViZn, its advantages and current problems, and its capabilities for the display of federal data are described in more detail in Jones and Symanzik (2001), Symanzik and Jones (2001), Symanzik et al. (2002), and Hurst et al. (2003).

While the main application of micromaps under nViZn was intended to be a proof of concept, based on the original EPA HAP data, the implementation of this application is very flexible. When additional data files in the appropriate format are provided, these data will be immediately usable within the nViZn application. The current application uses data at the national (US), state, county, and census tract level.

The micromaps created under the nViZn application are dynamic. The user can sort the regions in ascending or descending order with respect to the six statistical criteria minimum, mean, maximum, first quartile, median, and third quartile of the underlying census tract level. The number of regions that are displayed per micromap can be changed "on the fly" via a slider, so a user is not restricted to the perceptual grouping of size five or less that was introduced in Sect. 1.3 and can experiment with other group sizes. Micromaps and tables of the data can be created by selecting a HAP and US state in a drill-down map. Multiple LM plots or tabular displays can be viewed simultaneously.

The nViZn micromap application only works together with nViZn. The micromap application can be obtained freely from the first author upon request. At this point, SPSS no longer sells nViZn as a commercial product. However, the nViZn visualization service is a free simple service-based Java servlet application, accessible via the nViZn Web site mentioned above. Users pass GPL statements to the service via http requests and get a graph in return.

Micromaps via Java and Other Statistical Packages

1.5.3

The NCI recently developed a Java application to make LM plots available to those who are not familiar with statistical packages. The micromap producer must provide region boundary files in the form of .gen (or .shp files in the near future) and data files in the form of comma delimited (.csv) files. The interactive dialog enables the selection of plot types and the selection of columns from the data file to associate with the specific plot. For example, a dotplot with precalculated confidence bounds requires three variables. At the time of this writing, the software is still being tested in the classroom at George Mason University (GMU) and at other NCI-approved sites prior to general release. The software will most likely be available to the public before this chapter is published.

Most major statistical packages are flexible enough to produce LM plots. The basic issues are availability of boundary files, convenience of production, quality of appearance, and output format. For example, LM plots appeared some time ago in Wilkinson (1999), introducing the grammar of graphics. SPSS version 14.0 (<http://spss.com/spss/>) provides an implementation of this grammar, so this should make the production of LM plots easy for those who are familiar with the grammar of graphics.

The NCI LM plot applet in the State Cancer Profiles Web site, discussed in more detail in Sect. 1.4.3, provides LM plots to health planners in countries around the

world. This applet is available with Java Docs to other federal agencies and may be generally available for other developers as well. In fact, as already stated, the USU WNV Web-based micromap application described in Sect. 1.4.4 is based on the NCI Java code. The places in the Java code to make changes for other applications are isolated. The graphical user interface (GUI) prompts, options, and subsequent database access need to be modified for boundary files if regions other than the 50 US states and underlying counties are used. Similarly, the GUI prompts, options, and database access need to be modified for different data. This is straightforward for experienced Java programmers. Two GMU students were able to make the modification to obtain EPA's toxic release inventory data over the Web, and mainly one USU student did the adaptation for the WNV micromap server.

Implementations of LM plots are appearing in new settings. The Economic Research Service (ERS) of the USDA now has the capability to produce LM plots using software called Pop Charts. Micromaps recently were created for French *régions* and *départements* and may be included in the exploratory spatial analysis package GeoXP (<http://w3.univ-tlse1.fr/GREMAQ/Statistique/geoxppageR.htm>).

1.6

Discussion

In this chapter, we have demonstrated the use of interactive linked micromap plots for the display of geographically referenced statistical data. Some of the restrictions of choropleth maps do not apply for LM plots. It is possible to display multiple variables at a time in LM plots, provide summary statistics, and maintain the exact ranking of different subregions. The recent use of interactive LM plots on federal (and other) Web sites and their use in geographic publications are encouraging indicators for their widespread acceptance. We want to finish with a comparison of LM plots with Conditioned Choropleth Maps (CCmaps), introduced in Carr et al. (2000b) and Carr et al. (2002), and Trellis Graphics, based on ideas used in Cleveland (1993) and further discussed in Sect. 4.5 in Venables and Ripley (2002).

Carr et al. (2000b) developed conditioned choropleth maps as a way to show three variables using choropleth maps. The basic idea was to use the conditioning methodology described by Cleveland et al. (1993) to partition the regions in a map in a 3×3 set of panels containing partial maps. Figure 1.7 provides a CCmaps example based on the data from Fig. 1.3. The top slider in Fig. 1.7 shows boundaries to convert the 1997 soybean production (in bushels) per state into three color classes for use in a choropleth map. The bottom slider in Fig. 1.7 shows acreage boundaries for partitioning states into the columns of the 3×3 set of map panels. The right slider in Fig. 1.7 shows the yield boundaries for partitioning states into rows of the map panels. States with high acreage and high yield appear in the top right panel and, as expected, have a high number of bushels. The left column of panels highlights states with low acreage. Maryland and Pennsylvania are in the middle and high yield classes and are in the middle class in terms of bushels. All the other states in the left column are in

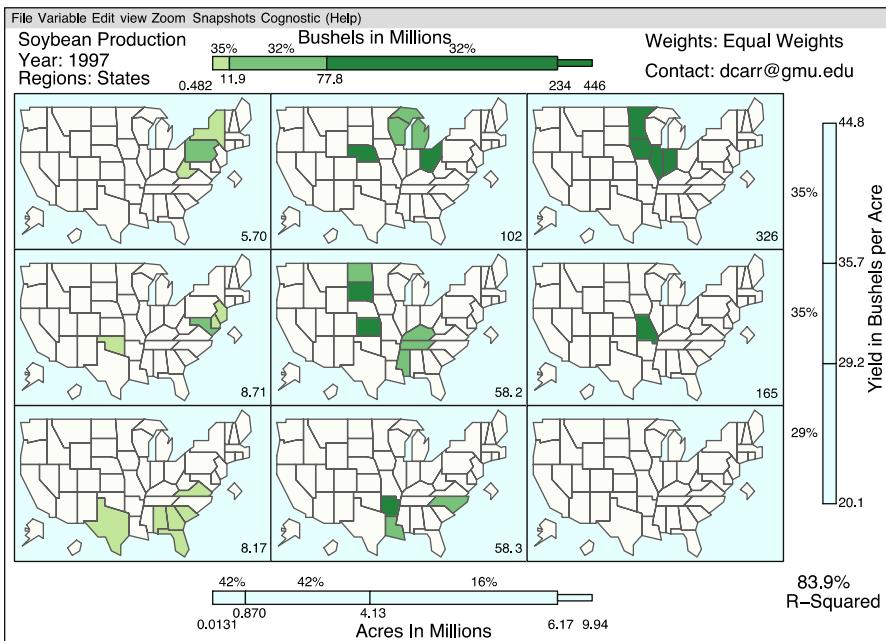


Figure 1.7. CCmaps, based on data from the USDA–NASS Web page. The same three variables production, acreage, and yield and the same data are shown as in Fig. 1.3; however, production is conditioned on acreage and yield here

the low class in terms of bushels. Since yield is calculated as production divided by acreage, there are no big surprises here.

Carr et al. (2005) presented CCmaps in the context of hypothesis generation. Even in this simple example with just three classes per variable (low, middle, and high), one may wonder why the high-yield states in the top row are not all in the rightmost panel with the four high acreage states as shown. Are less-soybean-acreage states smaller states in terms of total area or do they have less available fertile acreage? Is water an issue? Are there other crops that are more profitable? The comparative layout encourages thought and the mapping context often provides memory cues to what people know about the regions shown.

The cautious reader may wonder how much the specific slider settings influence the visual impression, and the curious reader may also wonder about all the numbers that appear in Fig. 1.7. Since CCmaps is dynamic software, it is trivial to adjust the two internal boundaries for each slider to see what happens. The maps change in real time and so do the numbers. The percents by the sliders indicate the percent of region weights in each class. In this example, all states involved in soybean production are weighted equally. For production, 32 % of the states (10 out of 31) with the highest production are highlighted in dark green across all 9 maps. The next 32 % of the states (10 out of 31) with production in the middle range are highlighted in medium green across all maps. Finally, the remaining 35 % of the states (11 out of 31) with the

lowest production are highlighted in light green across all maps. For acreage, 42 % of the states (13 out of 31) fall into the left column, 42 % (13 out of 31) into the middle column, and 16 % (5 out of 31) into the right column. Similarly, for yield, 35 % of the states (11 out of 31) fall into the top row, 35 % (11 out of 31) into the middle row, and 29 % (9 out of 31) into the bottom row. Thus, the four states highlighted in dark green (i.e., Indiana, Illinois, Iowa, and Minnesota) in the top right map belong to the 32 % of states with the highest production, 16 % of states with the highest acreage, and 35 % of states with the highest yield. Wisconsin, one of the spatial outliers identified in Sect. 1.2, is one of the states highlighted in medium green in the top central map and, thus, belongs to the 32 % of states with the medium production, 42 % of states with the medium acreage, and 35 % of states with the highest yield.

Most of the remaining numbers by the sliders indicate the class boundaries with the units communicated in the slider label. The top and bottom sliders have tails on the right. This reflects the presence of very large values relative to the body of the data. The lowest row of labels by each slider gives the minimum value, the upper adjacent value from a box plot calculation (Cleveland, 1993), and the maximum value. The slider scale changes more quickly over the tail. This leaves space for greater slider resolution in the body of the data.

The values in the upper right corner of each map show the weighted means for production for the states highlighted in those maps. Note that no state is highlighted in the bottom right map and, thus, no weighted mean is available. The fitting of these means to state values for the states highlighted in the maps leads to the R-squared value at the bottom right of Fig. 1.7.

The CCmaps software has several other features such as enlarged views of individual maps and the possibility to obtain specific values when the mouse is moved over a particular region. The software also provides zoom features to focus on a portion of a map and conditioned scatterplot smoothers. It is freely available at <http://www.galaxy.gmu.edu/~dcarr/ccmaps>.

So, finally, how do LM plots and CCmaps compare? First, the encoding of values in the Fig. 1.3 LM plots retains much more detail. The CCmaps encoding of three variables each into three classes loses much detail. Second, LM plots can include many additional statistics. For example, Fig. 1.6 shows two values for different time periods in the first statistical panel, rates with a 95 % confidence interval and a reference value in the second statistical panel, and the boxplots distributional summaries in the third statistical panel. CCmaps convey just three values per region. Carrying LM plots a bit further, Carr et al. (1998) illustrate sorting on one dot plot encoded variable, using this as the independent variable to obtain a smoothed fit to another variable. A second LM plots panel then shows observed and fitted values as well as the smooth in a vertical rank-repositioned smooth. This can help in seeing a functional relationship and large residuals. CCmaps does provide a separate view to show one-way dynamically conditioned smoothers. Since a similar separate view could be added to LM plots, this is not really a plus for CCmaps.

The CCmaps software does have a few merits. First, CCmaps scale better to maps involving more regions. Fairly common CCmaps examples show over 3000 US counties on the screen. The NCI State Cancer Profiles Web version of LM plots discussed

in Sect. 1.4.3 will only show counties within a given state, and then only a few counties are on the screen at one time. Texas has so many counties that the vast majority of them are out of sight in any view. Refreshing human memory about a big visual space via a scrollbar that reveals little pieces is not very effective. Second, the partitioning sliders in CCmaps are fun to use. If a search starts to get tedious, there is a built-in option to find initial slider settings with relatively good fits with respect to the R-squared value. Third, the CCmaps software is useful as educational software as it emphasizes two-way comparisons and weighted averages, and the sliders even lead to discussions about the difficulties related to long-tailed univariate distributions.

Trellis Graphics in S-Plus provide another approach to the display of data. Since Trellis Graphics are programmable, discussion here focuses on what is relatively easy and what is harder to do with Trellis Graphics. The perceptual grouping in LM plots can easily be fed to Trellis Graphics as an ordered categorical variable for the control of panel production. Showing dotplots with confidence bounds is not hard, nor is showing reference values. However, Trellis Graphics are primarily designed to handle a single dependent variable. They are poorly suited for showing multiple dependent variables in side-by-side columns such as the lung and bronchus cancer mortality and the percent of current smokers as in Fig. 1.4. Trellis Graphics were not designed to provide a variety of options such as geographic drill-down into subregions and blinking of linked symbols that are built into the Web-based software that produced Fig. 1.4. Though Trellis Graphics can provide some alternative views that may be very useful, they are not ideal for producing LM plots.

Acknowledgement. Jürgen Symanzik's work was supported in part by the NSF "Digital Government" (NSF 99-103) grant #EIA-9983461 and by a New Faculty Research Grant from the Vice President for Research Office from Utah State University. Dan Carr's work was also supported in part by NSF grant #EIA-9983461.

We would like to thank our many coauthors of newsletter articles, conference papers, and journal articles underlying this work. Much of the material presented in this paper goes back to collaborations with colleagues in federal and state agencies, projects with colleagues from several universities, and, most importantly, work with our students who contributed to the development of LM plots over the last 10 years. Additional thanks are due to Naomi B. Robbins for her helpful comments on an early and a late draft of this chapter and to Samson Gebreab for his help with the preparation of some of the figures. Finally, thanks are also due to D. Andrew Carr who worked on the grouping options as part of the Visual Basic front end to LM plots at the BLS.

References

- Becker, R.A., Chambers, J.M. and Wilks, A.R. (1988). *The New S Language – A Programming Environment for Data Analysis and Graphics*, Wadsworth and Brooks/Cole, Pacific Grove, CA.
- Boyer, R. and Savageau, D. (1981). *Places Rated Almanac*, Rand McNally, Chicago, IL.
- Brewer, C.A. (1997). Spectral Schemes: Controversial Color Use on Maps, *Cartography and Geographic Information Systems* 24(4):203–220.

- Brewer, C.A., MacEachren, A.M., Pickle, L.W. and Herrmann, D. (1997). Mapping Mortality: Evaluating Color Schemes for Choropleth Maps, *Annals of the Association of American Geographers* 87(3):411–438.
- Brewer, C.A. and Pickle, L.W. (2002). Comparison of Methods for Classifying Epidemiological Data on Choropleth Maps in Series, *Annals of the Association of American Geographers* 92(4):662–681.
- Carr, D.B. (1994). Converting Tables to Plots, *Technical Report 101*, Center for Computational Statistics, George Mason University, Fairfax, VA.
- Carr, D.B. (2001). Designing Linked Micromap Plots for States with Many Counties, *Statistics in Medicine* 20(9–10):1331–1339.
- Carr, D.B., Chen, J., Bell, B.S., Pickle, L. and Zhang, Y. (2002). Interactive Linked Micromap Plots and Dynamically Conditioned Choropleth Maps, *dg.o2002 Proceedings*, Digital Government Research Center (DGRC). http://www.dgrc.org/conferences/2002_proceedings.jsp.
- Carr, D.B. and Nusser, S.M. (1995). Converting Tables to Plots: A Challenge from Iowa State, *Statistical Computing and Statistical Graphics Newsletter* 6(3):11–18.
- Carr, D.B. and Olsen, A.R. (1996). Simplifying Visual Appearance by Sorting: An Example using 159 AVHRR Classes, *Statistical Computing and Statistical Graphics Newsletter* 7(1):10–16.
- Carr, D.B., Olsen, A.R., Courbois, J.P., Pierson, S.M. and Carr, D.A. (1998). Linked Micromap Plots: Named and Described, *Statistical Computing and Statistical Graphics Newsletter* 9(1):24–32.
- Carr, D.B., Olsen, A.R., Pierson, S.M. and Courbois, J.P. (2000a). Using Linked Micromap Plots to Characterize Omernik Ecoregions, *Data Mining and Knowledge Discovery* 4(1):43–67.
- Carr, D.B., Olsen, A.R. and White, D. (1992). Hexagon Mosaic Maps for Displays of Univariate and Bivariate Geographical Data, *Cartography and Geographic Information Systems* 19(4):228–236, 271.
- Carr, D.B. and Pierson, S.M. (1996). Emphasizing Statistical Summaries and Showing Spatial Context with Micromaps, *Statistical Computing and Statistical Graphics Newsletter* 7(3):16–23.
- Carr, D.B., Valliant, R. and Rope, D.J. (1996). Plot Interpretation and Information Webs: A Time-Series Example from the Bureau of Labor Statistics, *Statistical Computing and Statistical Graphics Newsletter* 7(2):19–26.
- Carr, D.B., Wallin, J.F. and Carr, D.A. (2000b). Two New Templates for Epidemiology Applications: Linked Micromap Plots and Conditioned Choropleth Maps, *Statistics in Medicine* 19(17–18):2521–2538.
- Carr, D.B., White, D. and MacEachren, A.M. (2005). Conditioned Choropleth Maps and Hypothesis Generation, *Annals of the Association of American Geographers* 95(1):32–53.
- Chapala, G.K. (2005). Development of Rich Features for Web-based Interactive Micromaps. Report, Department of Computer Science, Utah State University.
- Cleveland, W.S. (1993). *Visualizing Data*, Hobart Press, Summit, NJ.

- Cleveland, W.S., Grosse, E. and Shyu, W.M. (1993). Local Regression Models, in J. M. Chambers and T. J. Hastie (eds), *Statistical Models in S*, Chapman & Hall, New York, NY, pp. 309–376.
- Cleveland, W.S. and McGill, R. (1984). Graphical Perception: Theory, Experimentation, and Application to the Development of Graphical Methods, *Journal of the American Statistical Association* 79: 531–554.
- Dent, B.D. (1993). *Cartography: Thematic Map Design (Third Edition)*, William C. Brown, Dubuque, IA.
- Dorling, D. (1995). *A New Social Atlas of Great Britain*, John Wiley and Sons, New York, NY.
- Fonseca, J.W. and Wong, D.W. (2000). Changing Patterns of Population Density in the United States, *The Professional Geographer* 52(3):504–517.
- Harris, R.L. (1999). *Information Graphics – A Comprehensive Illustrated Reference*, Oxford University Press, New York, NY.
- Hurst, J., Symanzik, J. and Gunter, L. (2003). Interactive Federal Statistical Data on the Web using “nViZn”, *Computing Science and Statistics* 35. (CD) – Forthcoming.
- Ihaka, R. and Gentleman, R. (1996). R: A Language for Data Analysis and Graphics, *Journal of Computational and Graphical Statistics* 5(3):299–314.
- Jones, L. and Symanzik, J. (2001). Statistical Visualization of Environmental Data on the Web using nViZn, *Computing Science and Statistics* 33. (CD).
- Leslie, M. (2002). Tools: A Site for Sore Eyes, *Science* 296(5567):435.
- MacEachren, A.M., Brewer, C.A. and Pickle, L.W. (1995). Mapping Health Statistics: Representing Data Reliability, *Proceedings of the 17th International Cartographic Conference, Barcelona, Spain, September 3–9, 1995*, Institut Cartographic de Catalunya, Barcelona, Spain, pp. 311–319.
- MacEachren, A.M., Brewer, C.A. and Pickle, L.W. (1998). Visualizing Georeferenced Data: Representing Reliability of Health Statistics, *Environment and Planning A* 30: 1547–1561.
- Monmonier, M. (1993). *Mapping It Out: Expository Cartography for the Humanities and Social Sciences*, University of Chicago Press, Chicago, IL.
- Monmonier, M. (1996). *How to Lie with Maps (Second Edition)*, University of Chicago Press, Chicago, IL.
- Olsen, A.R., Carr, D.B., Courbois, J.P. and Pierson, S.M. (1996). Presentation of Data in Linked Attribute and Geographic Space, 1996 Abstracts, *Joint Statistical Meetings, Chicago, Illinois*, American Statistical Association, Alexandria, VA, p. 271.
- Palmer, S.E. (1999). *Vision Science, Photons to Phenomenology*, The MIT Press, Cambridge, MA.
- Pickle, L.W., Mungiole, M., Jones, G.K. and White, A.A. (1996). *Atlas of United States Mortality*, U.S. Department of Health and Human Services, Centers for Disease Control and Prevention, National Center for Health Statistics, Hyattsville, MD.
- Robinson, A., Sale, R. and Morrison, J. (1978). *Elements of Cartography (Fourth Edition)*, John Wiley and Sons, New York, NY.
- Rosenbaum, A.S., Axelrad, D.A., Woodruff, T.J., Wei, Y.-H., Ligocki, M.P. and Cohen, J.P. (1999). National Estimates of Outdoor Air Toxics Concentrations, *Journal of the Air and Waste Management Association* 49: 1138–1152.

-
- Symanzik, J. (2004). Interactive and Dynamic Graphics, in Gentle, J.E., Härdle, W. and Mori Y. (eds), *Handbook of Computational Statistics – Concepts and Methods*, Springer, Berlin, Heidelberg, pp. 293–336.
- Symanzik, J., Axelrad, D.A., Carr, D.B., Wang, J., Wong, D. and Woodruff, T.J. (1999a). HAPs, Micromaps and GPL – Visualization of Geographically Referenced Statistical Summaries on the World Wide Web. In *Annual Proceedings (ACSM-WFPS-PLSO-LSAW 1999 Conference CD)*. American Congress on Surveying and Mapping.
- Symanzik, J., Carr, D.B., Axelrad, D.A., Wang, J., Wong, D. and Woodruff, T.J. (1999b). Interactive Tables and Maps – A Glance at EPA's Cumulative Exposure Project Web Page. In *1999 Proceedings of the Section on Statistical Graphics*. pages 94–99, Alexandria, VA. American Statistical Association.
- Symanzik, J., Gebreab, S., Gillies, R. and Wilson, J. (2003). Visualizing the Spread of West Nile Virus, *2003 Proceedings*, American Statistical Association, Alexandria, VA. (CD).
- Symanzik, J., Hurst, J. and Gunter, L. (2002). Recent Developments for Interactive Statistical Graphics on the Web Using “nViZn”, *2002 Proceedings*, American Statistical Association, Alexandria, VA. (CD).
- Symanzik, J. and Jones, L. (2001). “nViZn” Federal Statistical Data on the Web, *2001 Proceedings*, American Statistical Association, Alexandria, VA. (CD).
- Symanzik, J., Wong, D., Wang, J., Carr, D.B., Woodruff, T.J. and Axelrad, D.A. (2000). Web-based Access and Visualization of Hazardous Air Pollutants, *Geographic Information Systems in Public Health: Proceedings of the Third National Conference August 18–20, 1998, San Diego, California*, Agency for Toxic Substances and Disease Registry. <http://www.atsdr.cdc.gov/GIS/conference98/>.
- Tufte, E.R. (1990). *Envisioning Information*, Graphics Press, Cheshire, CT.
- Venables, W.N. and Ripley, B.D. (2002). *Modern Applied Statistics with S (Fourth Edition)*, Springer, New York, NY.
- Wainer, H. and Francolini, C.M. (1980). An Empirical Inquiry Concerning Human Understanding of Two-Variable Color Maps, *The American Statistician* 34(2):81–93.
- Wang, X., Chen, J.X., Carr, D.B., Bell, B.S. and Pickle, L.W. (2002). Geographic Statistics Visualization: Web-based Linked Micromap Plots, *Computing in Science & Engineering* 4(3):90–94.
- Wilkinson, L. (1999). *The Grammar of Graphics*, Springer, New York, NY.
- Wilkinson, L., Rope, D.J., Carr, D.B. and Rubin, M.A. (2000). The Language of Graphics, *Journal of Computational and Graphical Statistics* 9(3):530–543.

Grand Tours, Projection Pursuit Guided Tours, and Manual Controls

III.2

Dianne Cook, Andreas Buja, Eun-Kyung Lee, Hadley Wickham

2.1	<i>Introductory Notes</i>	296
	Some Basics on Projections	297
	What Structure Is Interesting?.....	299
2.2	<i>Tours</i>	301
	Terminology: Plane, Basis, Frame, Projection	302
	Interpolating Between Projections: Making a Movie	302
	Choosing the Target Plane	303
	A Note on Transformations.....	310
	A Note on Scaling	310
2.3	<i>Using Tours with Numerical Methods</i>	310
2.4	<i>End Notes</i>	312

Introductory Notes

How do we find structure in multidimensional data when computer screens are only two-dimensional? One approach is to project the data onto one or two dimensions.

Projections are used in classical statistical methods like principal component analysis (PCA) and linear discriminant analysis. PCA (e.g., Johnson and Wichern 2002) chooses a projection to maximize the variance. Fisher's linear discriminant (e.g., Johnson and Wichern 2002) chooses a projection that maximizes the relative separation between group means. Projection pursuit (PP) (e.g., Huber 1985) generalizes these ideas into a common strategy, where an arbitrary function on projections is optimized. The scatterplot matrix (e.g., Becker and Cleveland 1987) also can be considered to be a projection method. It shows projections of the data onto all pairs of coordinate axes, the 2-D marginal projections of the data. These projection methods choose a few select projections out of infinitely many.

What is hidden from the user who views only a few static projections? There could be a lot. The reader may be familiar with an ancient fable from India about the blind men and the elephant. One grabbed his tail and swore the creature was a rope. Another felt the elephant's ear and yelled it was a hand fan. Yet another grabbed his trunk and exclaimed he'd found a snake. They argued and argued about what the elephant was, until a wise man settled the fight. They were all correct, but each described different parts of the elephant. Looking at a few static projections of multivariate data is like the blind men feeling parts of the elephant and inferring the nature of the whole beast.

How can a more systematic presentation of all possible projections be constructed? Static projections can be strung together into a movie using interpolation methods, providing the viewer with an overview of multivariate data. These interpolation methods are commonly called tours. They provide a general approach to choose and view data projections, allowing the viewer to mentally connect disparate views, and thus supporting the exploration of a high-dimensional space. We use tours to explore multivariate data like we might explore a new neighborhood: walk randomly to discover unexpected sights, employ a guide, or guide ourselves using a map. These modes of exploration are matched by three commonly available types of tours. They are the tours available in the software, GGobi (Swayne et al., 2001), which is used in this chapter to illustrate the methods.

- In the *grand tour*, we walk randomly around the landscape discovering unexpected sights – the grand tour shows all projections of the multivariate data. This requires time and we may spend a lot of time wandering around boring places and miss the highlights.
- Using a *PP guided tour*, we employ a tour guide who takes us to the features that they think are interesting. We improve the probability of stopping by the interesting sights by selecting more views that are interesting based on a PP index.
- *Manual control* takes the steering wheel back from the guide, enabling the tourist to decide on the next direction. We choose a direction by controlling the projec-

tion coefficient for a single variable. This allows us to explore the neighborhood of an interesting feature or to understand the importance of a variable on the feature.

Some Basics on Projections

2.1.1

What is a projection? We can think of a projection as the shadow of an object. Especially if it is a 2-D projection, then the projection is the shadow the object casts under a bright light (Fig. 2.1). If the object rotates in the light, we see many different 2-D shadows and we can infer the shape of the object itself.



Figure 2.1. Projections are like shadows. When many projections are viewed, it is possible to obtain a sense of the shape of a dataset. What may look like a horse in one projection may be revealed as a carefully oriented pair of hands by another

Mathematically, a projection of data is computed by multiplying an $n \times p$ data matrix, \mathbf{X} , having n sample points in p dimensions, by an orthonormal $p \times d$ projection matrix, \mathbf{A} , yielding a d -dimensional projection. For example, to project a 3-D object (3 columns, or variables, of data) onto a 2-D plane (the shadow of the object), we would use an orthonormal 3×2 matrix.

Here is a concrete example. Suppose our data matrix and projection were these:

$$\mathbf{X} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 15 \\ 0 & 15 & 0 \\ 0 & 15 & 15 \\ 15 & 0 & 0 \\ 15 & 0 & 15 \\ 15 & 15 & 0 \\ 15 & 15 & 15 \end{bmatrix}_{8 \times 3} \quad \text{and } \mathbf{A}_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}_{3 \times 2} \quad \text{then } \mathbf{XA}_1 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 15 \\ 0 & 15 \\ 15 & 0 \\ 15 & 0 \\ 15 & 15 \\ 15 & 15 \end{bmatrix}_{8 \times 2}$$

is the first two columns of the data matrix. If instead

$$\mathbf{A}_2 = \begin{bmatrix} 0.71 & -0.42 \\ 0.71 & 0.42 \\ 0 & 0.84 \end{bmatrix}_{3 \times 2} \quad \text{then } \mathbf{XA}_2 = \begin{bmatrix} 0 & 0 \\ 0 & 12.60 \\ 10.65 & 6.30 \\ 10.65 & 18.90 \\ 10.65 & -6.30 \\ 10.65 & 6.30 \\ 21.30 & 0 \\ 21.30 & 12.60 \end{bmatrix}_{8 \times 2}$$

is a combination of all three variables.

These projections are illustrated in Fig. 2.2. The top row shows the data projections, \mathbf{XA}_1 and \mathbf{XA}_2 , respectively. The bottom row displays the projection coefficients, \mathbf{A}_1 and \mathbf{A}_2 . A row in \mathbf{A} can also be interpreted as the projection of the coordinate axis (p -dimensional to d -dimensional) for each variable, and it is represented by a line in this display. The length and direction of the line displays the contribution each variable makes to the projected data view. In \mathbf{A}_1 the data projection is constructed purely from variable 1 in the horizontal direction and variable 2 in the vertical direction. In \mathbf{A}_2 variables 1 and 2 share the horizontal direction, and variable 3 makes no contribution horizontally. Vertically all three variables make a contribution, but variable 3 has twice the contribution of the other two variables. This type of axis display is used to match structure in a data projection with the variable dimensions of the data and, hence, enable to the analyst to interpret the data.

We also commonly use 1-D projections in data analysis. With a 1-D projection we typically use a histogram or density plot to display the data. Consider the 2-D data in Fig. 2.3 (left plot) and two 1-D projections (middle, right). The projection matrices are:

$$\mathbf{A}_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad \text{and} \quad \mathbf{A}_2 = \begin{bmatrix} -\frac{\sqrt{3}}{2} \\ -\frac{1}{2} \end{bmatrix}$$

respectively.

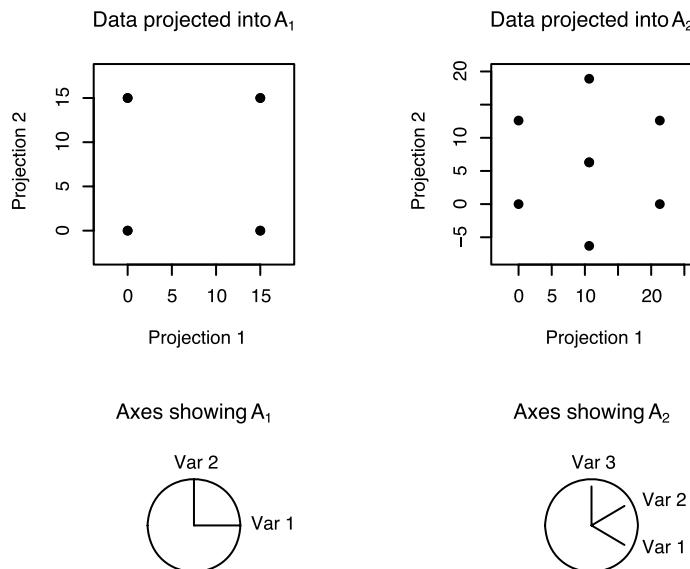


Figure 2.2. Two 2-D data projections. The two *top plots* are the data projections, and the two *bottom plots* are illustrations of the projection coefficients

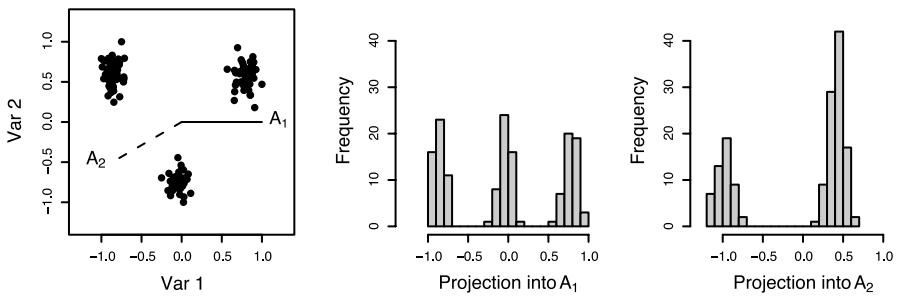


Figure 2.3. Two 1-D projections of 2-D data

What Structure Is Interesting?

2.1.2

When we use tours, what are we looking for in the data? We search for data projections that are *not* bell-shaped and, hence, not normally distributed, for example, clusters of points, outliers, nonlinear relationships, and low-dimensional substructures. All of these can be present in multivariate data but hidden from the viewer who only chooses a few static projections. Figures 2.4 and 2.5 show some examples.

In Fig. 2.4 a scatterplot matrix of all pairwise plots is shown at left, and a tour projection is shown at right. The pairwise plots show some linear association between three variables, particularly between the variables TEMP and PRESS, and TEMP and

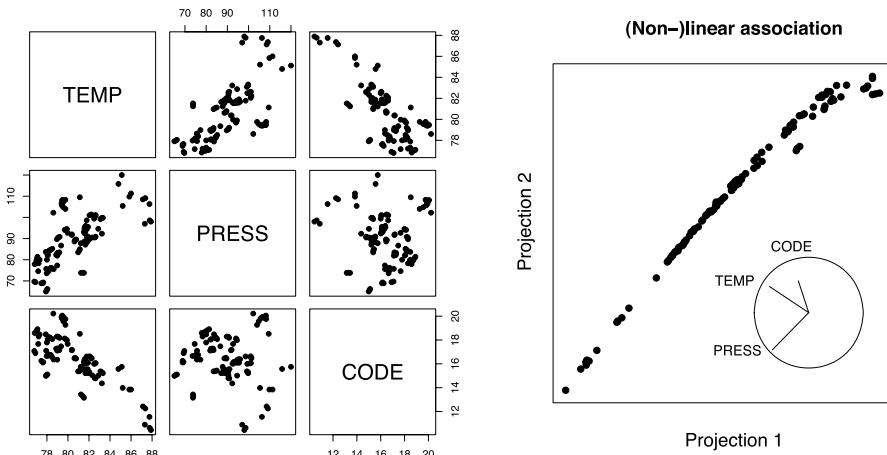


Figure 2.4. The three variables show some association in the scatterplot matrix (all pairwise marginal projections in *left plot*), but they are revealed to be almost perfectly related by a tour (*right plot*)

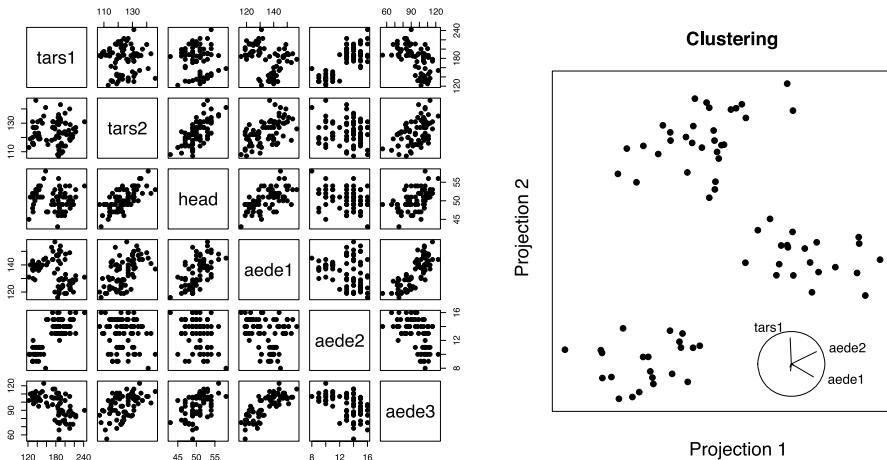


Figure 2.5. The six variables show some clustering in the scatterplot matrix (all pairwise marginal projections in *left plot*), but the three clusters are much better separated in a tour projection (*right plot*)

CODE. However, viewing the data in a tour reveals that the three variables are really perfectly related, with perhaps a slight nonlinear association. The projection of the data revealing the perfect relationship is:

$$\mathbf{A} = \begin{bmatrix} -0.720 & 0.470 \\ -0.668 & -0.671 \\ -0.191 & 0.573 \end{bmatrix} \begin{array}{l} \text{TEMP} \\ \text{PRESS} \\ \text{CODE} \end{array}$$

In Fig. 2.5 (left) the pairwise scatterplots suggest there is some clustering of the data points in this six-variable dataset. The tour projection (right) reveals three well-separated clusters. The projection revealing the clusters is:

$$\mathbf{A} = \begin{bmatrix} -0.035 & 0.801 \\ -0.023 & -0.215 \\ 0.053 & -0.032 \\ 0.659 & -0.398 \\ 0.748 & 0.378 \\ -0.043 & -0.097 \end{bmatrix} \begin{array}{l} \text{tars1} \\ \text{tars2} \\ \text{head} \\ \text{aedel} \\ \text{aede2} \\ \text{aede3} \end{array}$$

which is primarily a combination of three of the six variables: tars1, aedel, aede2.

Tours

2.2

Most of us are familiar with 3-D rotation, which is something we can do in the real world. We can take an object and rotate it with our hands or walk around an object to view it from all sides. Views of p -dimensional data can be computed in analogous ways, by rotating the entire p -dimensional data (Wegman, 1991; Carr et al., 1996; Tierney, 1991) or by moving a $d (< p)$ -dimensional plane through the space and projecting the data onto it. The latter approach is like looking at the data from different sides.

Movement of a projection plane is achieved by selecting a starting plane and a target plane and computing intermediate planes using a geodesic interpolation. A geodesic is a circular path, which is generated by constraining the planar interpolation to produce orthonormal descriptive frames. This is the method used in GGobi. It is more complicated to compute but it has some desirable properties, primarily that within-plane spin is eliminated by interpolating from plane to plane, rather than frame to frame. The frame that describes the starting plane is carried through the sequence of intermediate planes, preventing the data from rotating within the plane of view. That is, we avoid doing a rotation of the data as in Fig. 2.6. This type of within-plane rotation is distracting to the viewer, akin to viewing a scene while standing on

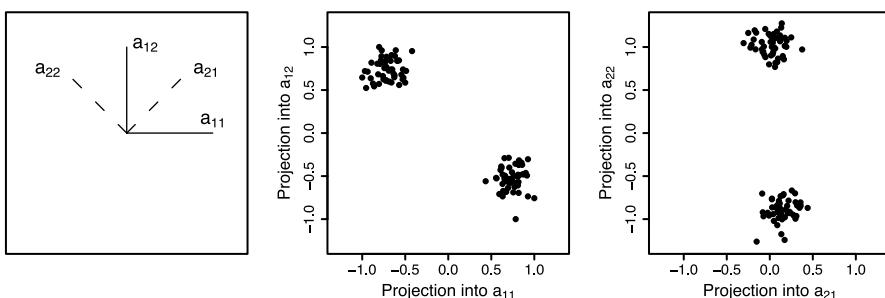


Figure 2.6. Two different frames that describe the same plane, and the resulting rotated views of the data

a wobbly platform. Planar rotations are discussed in detail in Asimov (1985), more simply in Buja and Aasimov (1986a), and very technically in Buja et al. (2005), and in Asimov and Buja (1994), and Buja et al. (1986b) as well.

Differences in the method of selecting the target plane yield different types of tours. The grand tour uses a random selection of target plane. The guided tour quantifies the structure present in each projection and uses this to guide the choice of target plane. Manual controls let the user choose the target direction by manipulating the values of projection coefficients.

2.2.1 Terminology: Plane, Basis, Frame, Projection

It is conventional to use a p -dimensional orthonormal basis:

$$\begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & & \\ \vdots & & \ddots & \\ 0 & & & 1 \end{bmatrix}_{p \times p}$$

to describe p -dimensional Euclidean space. A d -dimensional plane in p -space can be defined by an infinite number of d -dimensional orthonormal frames. For example, consider the $d = 2$ -dimensional frames:

$$\mathbf{A}_1 = \begin{bmatrix} \mathbf{a}_{11} \\ \mathbf{a}_{12} \\ \vdots \\ \mathbf{a}_{1p} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ \vdots & \vdots \\ 0 & 0 \end{bmatrix} \quad \text{and} \quad \mathbf{A}_2 = \begin{bmatrix} \mathbf{a}_{21} \\ \mathbf{a}_{22} \\ \vdots \\ \mathbf{a}_{2p} \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ 0 & 0 \\ \vdots & \vdots \\ 0 & 0 \end{bmatrix}$$

both of which describe the same 2-D plane. We conventionally use \mathbf{A}_1 as the frame describing the 2-D plane, but we could just as validly use \mathbf{A}_2 . Figure 2.6 illustrates the two frames, which result in the same but rotated projections of the data.

In GGobi tours, we generate a new target basis and use this to define the target plane. But the actual basis used to create the data projection is a within-plane rotation of the target basis that matches the basis describing the starting plane.

2.2.2 Interpolating Between Projections: Making a Movie

A movie of data projections is created by interpolating along a geodesic path from the current (starting) plane to the new target plane. The algorithm follows these steps:

- Given a starting $p \times d$ projection \mathbf{A}_a , describing the starting plane, create a new target projection \mathbf{A}_z , describing the target plane. It is important to check that \mathbf{A}_a and \mathbf{A}_z describe different planes, and generate a new \mathbf{A}_z if not. To find the optimal rotation of the starting plane into the target plane we need to find the frames in each plane that are the closest.

2. Determine the shortest path between frames using singular value decomposition. $\mathbf{A}'_a \mathbf{A}_z = \mathbf{V}_a \Lambda \mathbf{V}'_z$, $\Lambda = \text{diag}(\lambda_1 \geq \dots \geq \lambda_d)$, and the principal directions in each plane are $\mathbf{B}_a = \mathbf{A}_a \mathbf{V}_a$, $\mathbf{B}_z = \mathbf{A}_z \mathbf{V}_z$, a within-plane rotation of the descriptive bases \mathbf{A}_a , \mathbf{A}_z , respectively. The principal directions are the frames describing the starting and target planes that have the shortest distance between them. The rotation is defined with respect to these principal directions. The singular values, $\lambda_i, i = 1, \dots, d$, define the smallest angles between the principal directions.
3. Orthonormalize \mathbf{B}_z on \mathbf{B}_a , giving \mathbf{B}_* , to create a rotation framework.
4. Calculate the principal angles, $\tau_i = \cos^{-1} \lambda_i, i = 1, \dots, d$.
5. Rotate the frames by dividing the angles into increments, $\tau_i(t)$, for $t \in (0, 1]$, and create the i th column of the new frame, \mathbf{b}_i , from the i th columns of \mathbf{B}_a and \mathbf{B}_* , by $\mathbf{b}_i(t) = \cos(\tau_i(t))\mathbf{b}_{ai} + \sin(\tau_i(t))\mathbf{b}_{*i}$. When $t = 1$, the frame will be \mathbf{B}_z .
6. Project the data into $\mathbf{A}(t) = \mathbf{B}(t)\mathbf{V}'_a$.
7. Continue the rotation until $t = 1$. Set the current projection to be \mathbf{A}_a and go back to step 1.

Choosing the Target Plane

2.2.3

Grand Tour

The grand tour method for choosing the target plane is to use random selection. A frame is randomly selected from the space of all possible projections.

A target frame is chosen randomly by standardizing a random vector from a standard multivariate normal distribution: sample p values from a standard univariate normal distribution, resulting in a sample from a standard multivariate normal. Standardizing this vector to have length equal to one gives a random value from a $(p-1)$ -dimensional sphere, that is, a randomly generated projection vector. Do this twice to get a 2-D projection, where the second vector is orthonormalized on the first.

Figure 2.7 illustrates the tour path, using GGobi to look at itself. Using GGobi, we recorded the sequence of 9000 1D projections displayed of 3-D data. This tour path is a set of 9000 points on a 3-D sphere, where each point corresponds to a projection. We use a tour to view the path (top left plot). The starting projection is $\mathbf{A}' = (1 \ 0 \ 0)$, indicated by a large point (●) or solid circle in the display. It is at the center right in the plot, a projection in the first two variables. The corresponding data projection is shown at top right. The grand tour path zigzags around the 3-D sphere. The grand tour can be considered as an interpolated random walk over the space of all planes. With enough time it will entirely cover the surface of the sphere. The bottom row of plots shows two views of a grand tour path of 20 000 1-D projections of 6-dimensional data.

Projection Pursuit Guided Tour

In a guided tour (Cook et al., 1995) the next target basis is selected by optimizing a PP index function. The index function numerically describes what is interesting

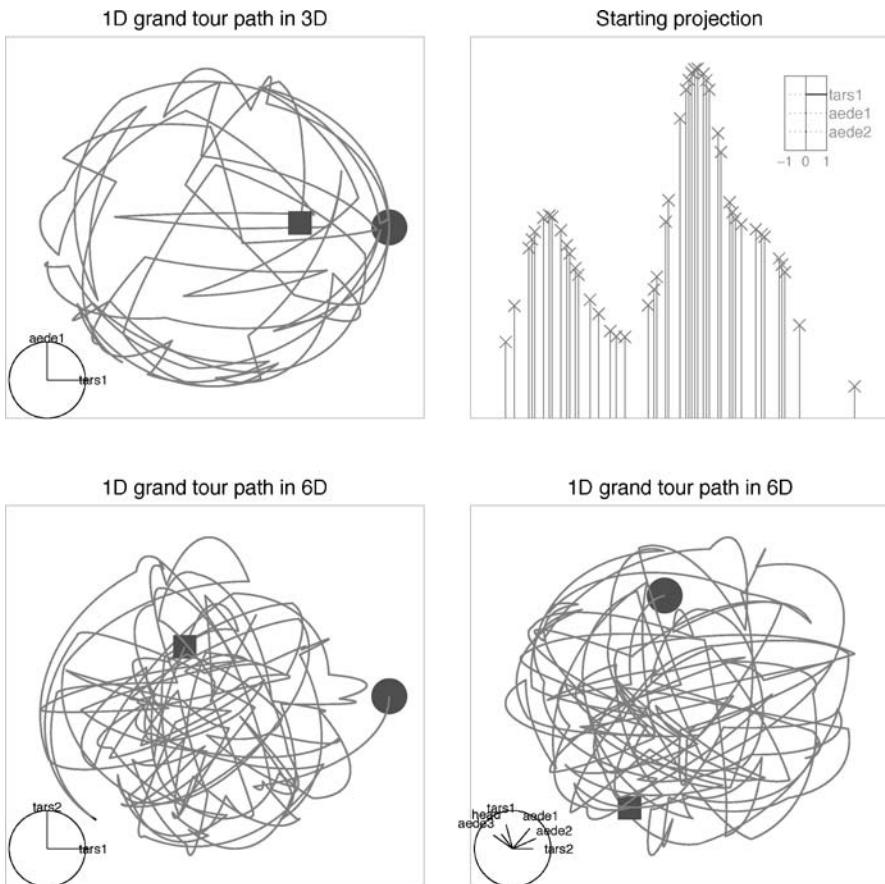


Figure 2.7. Some views of 1-D grand tour paths in 3 dimensions (*top left*) and 6 dimensions (*bottom*). The path consists of a sequence of points on a 2-D and 6-D sphere respectively. Each *point* corresponds to a projection from 3 dimensions (or 6 dimensions) to 1 dimension. The *solid circle* indicates the first point on the tour path corresponding to the starting frame, yielding the 1-D data projection (*top right*) shown for the 3-D path. The *solid square* indicates the last point in the tour path, or the last projection computed

in a projection: higher values correspond to more interesting structure in the projections. Used alone, PP seeks out low-dimensional projections that expose interesting features of the high-dimensional point cloud. In conjunction with the interpolation, a PP guided tour shows many projections to the viewer, in a smooth sequence. Using a PP index function to navigate the high-dimensional data space has the advantage over the grand tour of increasing the proportion of interesting projections visited.

The PP index, $f(\mathbf{XA})$, is optimized over all possible d -dimensional projections of p -dimensional data, subject to orthonormality constraints on \mathbf{A} . The optimization

procedure is an important part of a PP guided tour. The purpose of PP optimization is to find all of the interesting projections, so an optimization procedure needs to be flexible enough to find global and local maxima. It should not doggedly search for a global maximum, but it should spend some time visiting local maxima.

Posse (1990) compared several optimization procedures and suggest a random search for finding the global maximum of a PP index. Cook et al. (1995) used a derivative-based optimization, always climbing the nearest hill, which when merged with a grand tour was a lot like interactive simulated annealing. Klein and Dubes (1989) showed that simulated annealing can produce good results for PP.

Lee et al. (2005) use the modified simulated annealing method. It uses two different temperatures, one for neighborhood definition and the other (cooling parameter) for the probability that guards against getting trapped in a local maximum. This allows the algorithm to visit a local maximum and then jump out and look for other maxima. The temperature of the neighborhood is rescaled by the cooling parameter enabling escape from the local maximum. The optimization algorithm used in GGobi follows these steps:

1. From the current projection, \mathbf{A}_a , calculate the initial PP index value, $I_0 = f(\mathbf{XA}_a)$.
2. Generate new projections, $\mathbf{A}_i^* = \mathbf{A}_a + c\mathbf{A}_i$, from a neighborhood of the current projection where the size of the neighborhood is specified by the cooling parameter, c , and \mathbf{A}_i is a random projection.
3. Calculate the index value for each new projection, $I_i = f(\mathbf{XA}_i^*)$.
4. Set the projection with the highest index value to be the new target, $\mathbf{A}_z = \mathbf{A}_{\max_i I_i}$, and interpolate from \mathbf{A}_a to \mathbf{A}_z .

Figure 2.8 (top two plots) shows a PP guided tour path (1-D in three dimensions). It looks very similar to a grand tour path, but there is a big difference: the path repeatedly returns to the same projection and its negative counterpart (both highlighted by large solid black circles). The middle plot traces the PP index value over time. The path iterates between optimizing the PP function and random target basis selection. The peaks (highlighted by large solid black circles) are the maxima of the PP index, and for the most part, these are at the same projection. The corresponding data projections (approximately positive and negative of the same vector) are shown in the bottom row. The index is responding to a bimodal pattern in the data.

There are numerous PP indices. Here are a few that are used in GGobi. For simplicity in the formula for holes, central Mass, and PCA indices, it is assumed that \mathbf{X} is spheroid using PCA, that is, the mean is zero and the variance–covariance is equal to the identity matrix. This assumption is not necessary for the LDA index.

Holes:

$$I_{Holes}(\mathbf{A}) = \frac{1 - \frac{1}{n} \sum_{i=1}^n \exp(-\frac{1}{2}\mathbf{y}'_i \mathbf{y}_i)}{1 - \exp(-\frac{d}{2})}$$

where $\mathbf{XA} = \mathbf{Y} = [\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_n]^T$ is a $n \times d$ matrix of the projected data.

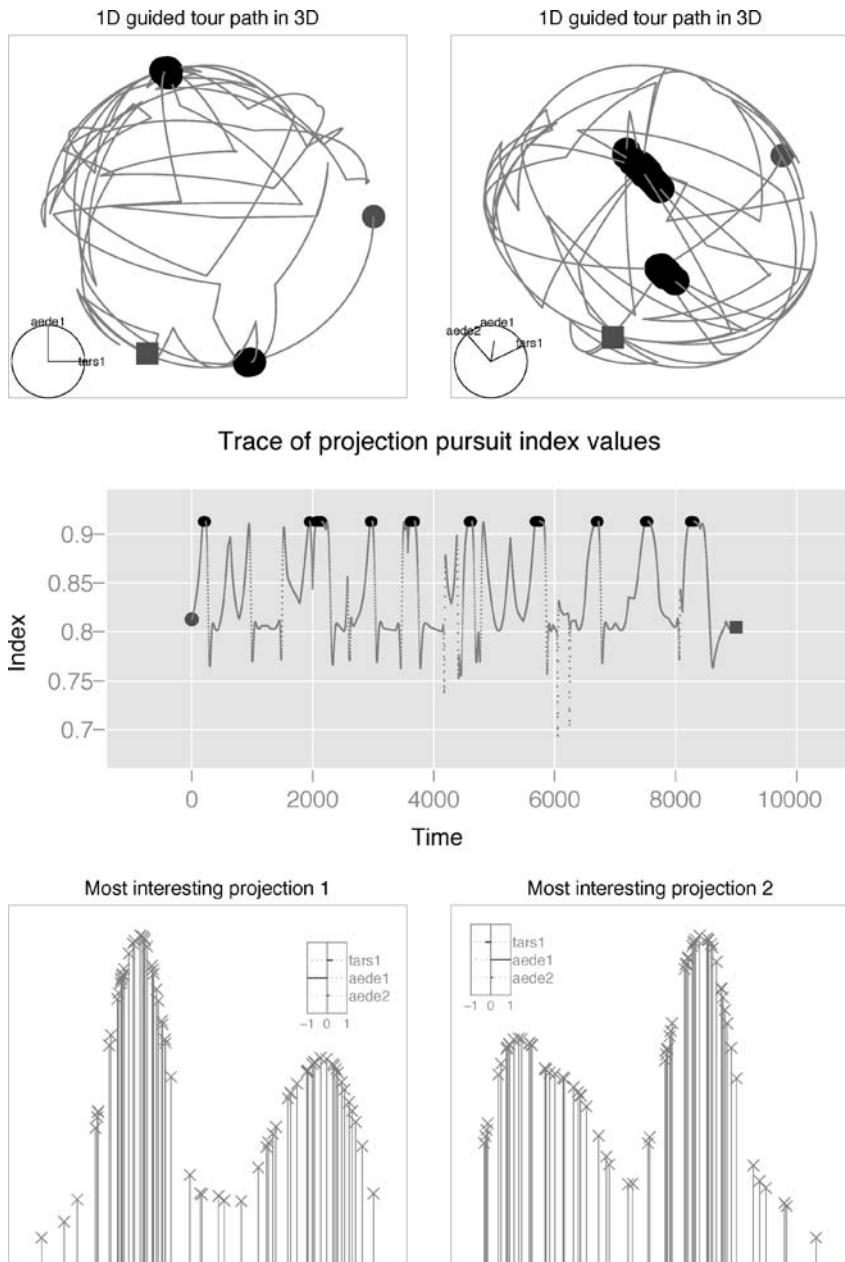


Figure 2.8. Projection pursuit guided tours. *Top:* path of 1-D projections in 3 dimensions. *Middle:* time trace of PP index. *Bottom:* data projections corresponding to PP index maxima. The maxima are highlighted by large solid circles. It is interesting here that the optimization keeps returning the tour to similar neighborhoods in the projection space, corresponding to bimodal densities in the 1-D projections

Central mass:

$$I_{CM}(\mathbf{A}) = \frac{\frac{1}{n} \sum_{i=1}^n \exp(-\frac{1}{2}\mathbf{y}'_i \mathbf{y}_i) - \exp(-\frac{d}{2})}{1 - \exp(-\frac{d}{2})}$$

where $\mathbf{XA} = \mathbf{Y} = [\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_n]^T$ is an $n \times d$ matrix of the projected data.

LDA:

$$I_{LDA}(\mathbf{A}) = 1 - \frac{|\mathbf{A}' \mathbf{W} \mathbf{A}|}{|\mathbf{A}' (\mathbf{W} + \mathbf{B}) \mathbf{A}|}$$

where $\mathbf{B} = \sum_{i=1}^g n_i (\bar{\mathbf{X}}_{i\cdot} - \bar{\mathbf{X}}_{..})(\bar{\mathbf{X}}_{i\cdot} - \bar{\mathbf{X}}_{..})'$, $\mathbf{W} = \sum_{i=1}^g \sum_{j=1}^{n_i} (\mathbf{X}_{ij} - \bar{\mathbf{X}}_{i\cdot})(\mathbf{X}_{ij} - \bar{\mathbf{X}}_{i\cdot})'$ are the “between” and “within” sum-of-squares matrices from linear discriminant analysis, g is the number of groups, $n_i, i = 1, \dots, g$ is the number of cases in each group.

PCA: This is only defined for $d = 1$.

$$I_{PCA}(\mathbf{A}) = \frac{1}{n} \mathbf{Y}' \mathbf{Y} = \frac{1}{n} \sum_{i=1}^n y_i^2$$

where $\mathbf{XA} = \mathbf{Y} = [y_1, y_2, \dots, y_n]^T$ is an $n \times d$ matrix of the projected data.

Figure 2.9 shows the results of using different indices on the same data. The holes index finds a projection where there is a gap between two clusters of points. The central mass index finds a projection where a few minor outliers are revealed. The LDA index finds a projection where three clusters can be seen. The PCA index finds a trimodal data projection.

Manual Controls

Manual controls enable the user to manually rotate a single variable into or out of a projection. This gives fine-tuning control to the analyst. Cook and Buja (1997) has details on the manual controls algorithm. It is similar to a method called spiders proposed by Duffin and Barrett (1994).

Figure 2.10 illustrates the use of manual controls to examine the results of the LDA index (top left plot, also shown at bottom left in Fig. 2.9). In this view there are three very clearly separated clusters of points. The projection is mostly PC1 (a large positive coefficient), with smaller coefficients for PC2 and PC6. The remaining PCs have effectively zero coefficients. We explore the importance of these small coefficients for the three-cluster separation. From the optimal projection given by the LDA index we manually rotate PC6 out of the projection and follow by rotating PC2 out of the projection:

$$\mathbf{A} = \begin{bmatrix} 0.889 \\ 0.435 \\ 0.040 \\ 0.053 \\ 0.033 \\ 0.122 \end{bmatrix} \implies \begin{bmatrix} 0.896 \\ 0.439 \\ 0.040 \\ 0.053 \\ 0.033 \\ 0.004 \end{bmatrix} \implies \begin{bmatrix} 0.938 \\ 0.339 \\ 0.042 \\ 0.055 \\ 0.035 \\ 0.004 \end{bmatrix} \implies \begin{bmatrix} 0.996 \\ 0.026 \\ 0.045 \\ 0.059 \\ 0.037 \\ 0.004 \end{bmatrix}$$

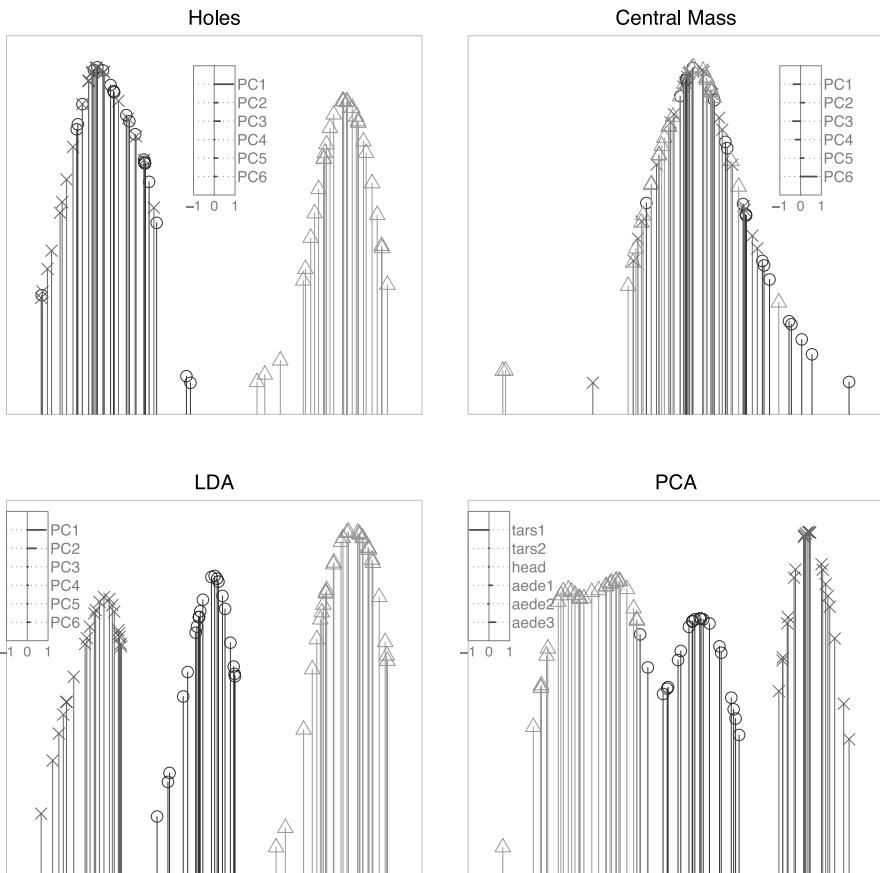


Figure 2.9. One-dimensional data projections corresponding to maxima from four different PP indices computed on the same data. The interesting feature of the data is the separation of the three classes. *Top left:* the holes index finds a projection with a hole in the middle, where one cluster is separated from the other two. *Top right:* the central mass index finds a projection where most points are clumped in the center, revealing a few outliers. *Bottom left:* LDA, using the class information, finds a separation of all three clusters. *Bottom right:* the PCA index finds a projection where all three classes are somewhat distinct

PC6 is rotated out of the projection first (Fig. 2.10, top right). Note that all the coefficients change some because they are constrained by the orthonormality of the p -dimensional data frame. But notice that the coefficient for PC6 is effectively reduced to zero. There is very little change to the projected data, so this variable might be ignored. Next we explore the importance of PC2 by rotating it out of the projection (Fig. 2.10, bottom row). A small change in the coefficient for PC2 results in a blurring of the gap between the two leftmost clusters (bottom left plot). When PC2 is completely removed (bottom right plot), the two leftmost clusters are indistinguishable.

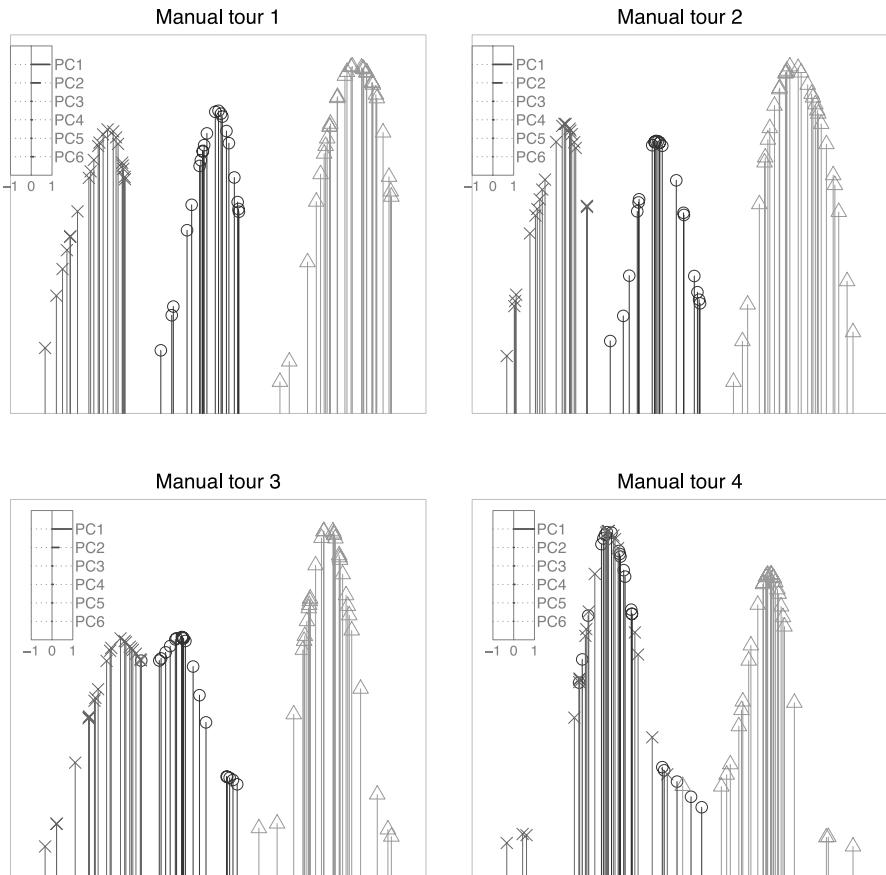


Figure 2.10. Manual controls used to examine the sensitivity of the clustering revealed by the LDA index to PC6 and PC2 is explored. Top left to top right plots: coefficient on PC6 reduced from 0.122 to 0.004 gives a smaller gap between clusters, but the clusters are still separable. Bottom left to bottom right: coefficient for PC2 reduced from 0.439 to 0.026 which removes the cluster structure

But the right cluster is still separated. This suggests that PC2 is important for separating the two leftmost clusters but not important for separating the right cluster.

Precomputed Choices

One of the simplest choices of target planes that creates a smooth transition from scatterplot matrices is the little tour (McDonald, 1982) that interpolates between the frames of a scatterplot matrix of all pairwise marginal projections. Conway et al. (1996) proposes a method for choosing a fixed number of target planes that are approximately equispaced. It chooses the target planes using packing methods on polytopes and determines a shortest (Hamiltonian) path through the set of targets. Neither of these methods is implemented in GGobi.

2.2.4

A Note on Transformations

When analyzing data it is common to transform variables to examine them on different scales. Transformation plays a useful role prior to PP as well. The most common transformation before PP is to “sphere” the data. Spherizing the data means to compute the principal components of the data and to use the resulting variables instead of the original variables. The major reason to do this is that we are not interested in covariance structure. This is adequately captured by PCA. Consequently we commonly remove the covariance from the data before running PP and search for other types of structure in the data. In Fig. 2.9 the labels of the variables in some of the plots PC_1, PC_2, \dots reflect that the data were spherized prior to running the PP guided tour. Sometimes transformations are performed to relieve the data of outliers or skewness. When these occur in single variables, they can be detected and addressed before running PP, but PP is useful for detecting multivariate outliers and nonlinear dependencies in high-dimensional data.

2.2.5

A Note on Scaling

Plots of data are generally constructed by scaling the data using the minimum and maximum data values to fit the data into a plotting space, on a computer screen window, or sheet of paper. Axes are provided so the viewer can convert the points into the original scales.

For high-dimensional data each variable is scaled to a uniform scale using the minimum and maximum, packing the data into a p -dimensional hyperrectangle. These scaled data are projected into a plotting space. It might be interesting to think about scaling the data after a projection is computed, but the effect of this approach is a discontinuity from one projection frame to the next. It would be like watching a movie where the camera lens constantly zooms and pans.

The PP guided tour operates on the unscaled data values. (It may also be important to transform the data by standardizing variables or spherizing before running PP, as discussed in the previous paragraph.) The process of scaling data into a plotting space is called the data pipeline and is discussed in detail in Buja et al. (1988), Sutherland et al. (2000), and in a different sense, in Wilkinson (1999) and Pastizzo et al. (2002).

2.3

Using Tours with Numerical Methods

Tours are useful when used along with numerical methods for certain data analyses, such as dimension reduction and supervised and unsupervised classification. We'll demonstrate with an example from supervised classification.

In supervised classification we seek to find a rule for predicting the class of new observations based on training a classifier using known classes. There are many numerical methods that tackle this problem.

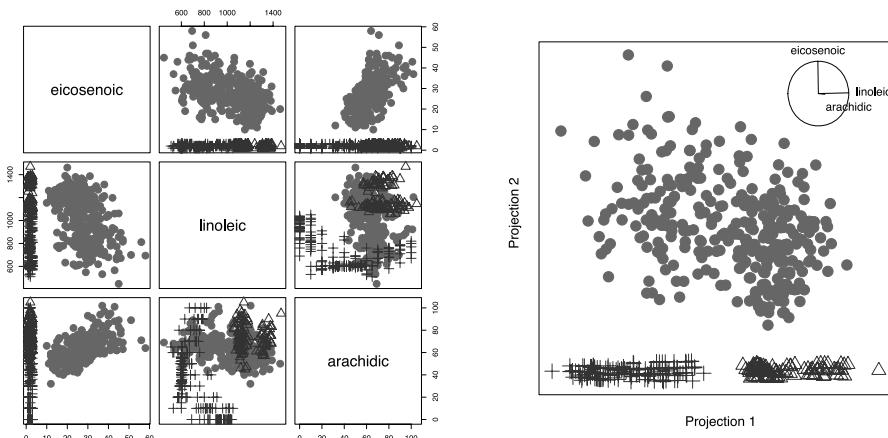


Figure 2.11. *Left plot:* scatterplot matrix of three of the important variables for separating the three classes. A single classification tree usually produces the result to split the three classes based on two variables, linoleic and eicosenoic. *Right:* a projection of linoleic and arachidic, along with eicosenoic, produces a better gap between the classes

For the dataset shown in Fig. 2.11 there are eight variables and three known classes. A classification tree chooses just two of the variables, eicosenoic and linoleic, to separate the three classes. For the training sample eicosenoic separates one class (plotted as circles) from the other two, and linoleic separates the remaining two classes (plusses and triangles). The separation of these last two groups, although difficult to see in the plot of eicosenoic against linoleic, is real (scatterplot matrix at left). There is no gap between the groups of points, but it is possible to draw a line with points from one class on one side of it and the points from the other class on the other side. By using a tour we would have noticed that there is a big gap between the three classes using all eight variables, and also that choosing just three provides a very neat separation. It would be difficult to guess from pairwise plots that arachidic has an important role, but from the tour we can see that when arachidic is combined with linoleic the two classes are much better separated (right plot). The tour projection shows the combination of linoleic and arachidic plotted horizontally that reveals the gap. The tree solution was simple but inadequate, and a small change to the solution provides a much better result.

The tree algorithm was hampered by both variable wise operation and greediness. It did not see the combination of linoleic and arachidic because it could only use one variable at each step. It also stopped immediately when a separation between the classes was found, having no sense of a bigger gap elsewhere. All numerical methods have assumptions or algorithm constraints or complexity that sets limits on the results. A classical method such as linear discriminant analysis assumes that the classes in the data arise from a mixture of normal distributions having equal variance–covariance. Linear discriminant analysis finds a best separating projection similar to the tree solution; one group is well-separated and the other two groups slightly over-

lapped. It is blinded by the assumption that the three classes have equal variance-covariance. Quadratic discriminant analysis does better in making a rule but cannot provide a good picture of the solution. The solution of black-box methods such as forests and neural networks are generally difficult to understand, but by mapping out a picture of the class structure of these data using a tour we can better understand how they have worked and the resulting solution.

Tours can help numerical approaches in many ways: to choose which of the tools at hand works best for a particular problem, to understand how the tools work in a particular problem, and to overcome the limitations of a particular tool to improve the solution.

2.4

End Notes

There are numerous recent developments in tours that should be noted. Huh and Kim (2002) describes a grand tour with trailing tails marking the movement of the points in previous projections. The tour can be constructed in different projection dimensions and constraints. Yang (1999) describes a grand tour with 3-D data projections in virtual environments. The correlation tour described by Buja et al. (1986b), and available in GGobi, runs two independent tours of 1-D projections on horizontal and vertical axes. This paper also describes constraining the tour to special subspaces such as principal components or canonical coordinates. XGobi (Swayne et al., 1998) contained tools for freezing some axes and touring in the constrained complement space, and also a section tour, where points outside a fixed distance from the projection place were erased. Wegman et al. (1998) and Symanzik et al. (2002) discuss a tour on the multivariate measurements constrained on spatial locations, which is similar to the multivariate time series tour discussed in Sutherland et al. (2000), where 1-D projections are shown against a time variable.

In summary, tours support exploring real-valued data. They deliver many projections of real-valued data in an organized manner, allowing the viewer to see the data from many sides.

References

- Asimov, D. (1985). The grand tour: a tool for viewing multidimensional data, *SIAM J Sci Stat Comput* 6(1):128–143.
- Asimov, D. and Buja, A. (1994). The grand tour via geodesic interpolation of 2-Frames, *Visual Data Exploration and Analysis, Symposium on Electronic Imaging Science and Technology*, IS&T/SPIE.
- Buja, A. and Asimov, D. (1986a). Grand tour methods: an outline, *Comput Sci Stat* 17:63–67.
- Buja, A., Asimov, D., Hurley, H. and McDonald, J.A. (1988). Elements of a Viewing Pipeline for Data Analysis, in Cleveland, W.S. and McGill, M.E. (ed), *Dynamic Graphics for Statistics*, Wadsworth, Monterey, CA, pp. 277–308.

- Buja, A., Cook, D., Asimov, D. and Hurley, C. (2005). Computational methods for high-dimensional rotations in data visualization, in Solka, J.L., Rao, C.R. and Wegman, E.J. (ed), *Handbook of Statistics: Data Mining and Visualization*, Elsevier/North Holland, <http://www.elsevier.com>, pp. 391–413.
- Buja, A., Hurley, C. and McDonald, J.A. (1986b). A Data Viewer for Multivariate Data, in Stefanski, I.M. Boardman, T.J. (ed), *Proceedings of the 18th symposium on the interface between computer science and statististics*, Elsevier, Fort Collins, CO, pp. 171–174.
- Carr, D.B., Wegman, E.J. and Luo, Q. (1996). ExplorN: design considerations past and present, *Technical report*, Center for Computational Statistics, George Mason University.
- Conway, J.H., Hardin, R.H. and Sloane, N.J.A. (1996). Packing lines, planes etc., packings in Grassmannian spaces, *Exp Math* 5:139–159.
- Cook, D., Buja, A. (1997). Manual controls for high-dimensional data projections, *J Comput Graph Stat* 6(4):464–480. Also see www.public.iastate.edu/~dicook/research/papers/manip.html.
- Cook, D., Buja, A., Cabrera, J. and Hurley, C. (1995). Grand tour and projection pursuit, *J Comput Graph Stat* 4(3):155–172.
- Duffin, K.L., Barrett W.A. (1994). Spiders: a new interface for rotation and visualization of N-dimensional point sets, *Proceedings of Visualization '94*, IEEE Computer Society Press, Los Alamitos, CA, pp. 205–211.
- Huh, M.Y. and Kim, K. (2002). Visualization of multidimensional data using modifications of the grand tour, *J Appl Stat* 29(5):721–728.
- Klein, R.W. and Dubes, R.C. (1989). Experiments in projection and clustering by simulated annealing, *Pattern Recog* 22(2):213–220.
- Lee, E.K., Cook, D., Klinke, S. and Lumley, T. (2005). Projection pursuit for exploratory supervised classification, *J Comput Graph Stat* p. To appear.
- McDonald, J.A. (1982). Interactive Graphics for Data Analysis, *Technical Report Orion II*, Statistics Department, Stanford University.
- Pastizzo, M.J., Erbacher, R.F. and Feldman, L.B. (2002). Multi-dimensional data visualization, *Behav Res Methods, Instrum Comput* 34(2):158–162.
- Posse, C. (1990). An effective two-dimensional projection pursuit algorithm, *Commun Stat Simulat Comput* 19:1143–1164.
- Sutherland, P., Rossini, A., Lumley, T., Lewin-Koh, N., Dickerson, J., Cox, Z. and Cook, D. (2000). Orca: a visualization toolkit for high-dimensional data, *J Comput Graph Stat* 9(3):509–529.
- Swayne, D., Temple Lang, D., Cook, D. and Buja, A. (2001). GGobi: software for exploratory graphical analysis of high-dimensional data, [Jul/01] Available publicly from www.ggobi.org.
- Swayne, D.F., Cook, D. and Buja, A. (1998). XGobi: interactive dynamic graphics in the X Window system, *J Comput Graph Stat* 7(1):113–130. See also www.research.att.com/areas/stat/xgobi/.
- Symanzik, J., Wegman, E.J., Braverman, A. and Luo, Q. (2002). New applications of the image grand tour, *Comput Sci Stat* 34:500–512.

- Tierney, L. (1991). *LispStat: an object-orientated environment for statistical computing and dynamic graphics*, Wiley, New York.
- Wegman, E.J. (1991). The grand tour in k-dimensions, *Comput Sci Stat* 22:127–136.
- Wegman, E.J., Poston, W.L. and Solka, J.L. (1998). Image grand tour, *Automatic Target Recognition VIII – Proc SPIE*, 3371, pp. 286–294.
- Wilkinson, L. (1999). *The Grammar of Graphics*, Springer, New York.
- Yang, L. (1999). 3D grand tour for multidimensional data and clusters. In: Proceedings of the Third International Symposium on Advances in Intelligent Data Analysis. *Lecture Notes in Computer Science*, vol. 1642, Springer-Verlag, London, UK, pp. 173–186

Multidimensional Scaling

III.3

Michael A.A. Cox, Trevor F. Cox

3.1	<i>Proximity Data</i>	316
3.2	<i>Metric MDS</i>	319
3.3	<i>Non-metric MDS</i>	322
3.4	<i>Example: Shakespeare Keywords</i>	325
3.5	<i>Procrustes Analysis</i>	330
3.6	<i>Unidimensional Scaling</i>	331
3.7	<i>INDSCAL</i>	333
3.8	<i>Correspondence Analysis and Reciprocal Averaging</i>	338
3.9	<i>Large Data Sets and Other Numerical Approaches</i>	341

Suppose dissimilarity data have been collected on a set of n objects or individuals, where there is a value of dissimilarity measured for each pair. The dissimilarity measure used might be a subjective judgement made by a judge, where for example a teacher subjectively scores the strength of friendship between pairs of pupils in her class, or, as an alternative, more objective, measure, she might count the number of contacts made in a day between each pair of pupils. In other situations the dissimilarity measure might be based on a data matrix. The general aim of multidimensional scaling is to find a configuration of points in a space, usually Euclidean, where each point represents one of the objects or individuals, and the distances between pairs of points in the configuration match as well as possible the original dissimilarities between the pairs of objects or individuals. Such configurations can be found using metric and non-metric scaling, which are covered in Sects. 2 and 3. A number of other techniques are covered by the umbrella title of multidimensional scaling (MDS), and here the techniques of Procrustes analysis, unidimensional scaling, individual differences scaling, correspondence analysis and reciprocal averaging are briefly introduced and illustrated with pertinent data sets.

Much of the initial impetus and theory of MDS was developed by mathematical psychologists who published many of their findings in the journal *Psychometrika*. Although its roots are in the behavioural sciences, MDS has now become more widely popular and has been employed in a wide variety of areas of application. This popularity is reflected by its inclusion in many computer-based statistical packages. Books on the subject include those by Borg and Groenen (1997), Cox and Cox (2001) and Young (1987).

3.1 Proximity Data

Proximity means nearness in whatever space is under consideration. The “nearness” of objects, individuals or stimuli needs defining prior to any analysis. In some situations, such as with simple Euclidean distance, this is straightforward. There are two types of basic measure of proximity, similarity and dissimilarity, with these being employed to indicate how similar or dissimilar objects are. The similarity/dissimilarity measured between two objects is a real function, resulting in similarity s_{rs} or dissimilarity δ_{rs} between the r th and s th objects. Usually all measures are taken to be non-negative. The dissimilarity of an object with itself is taken to be zero, while the similarity of an object with itself is the maximum similarity possible, with similarities usually scaled so that the maximum similarity is unity. The choice of proximity measure will depend on the problem under consideration. Sometimes the measure between two individuals is not based on any underlying observations and is totally subjective as with the teacher scoring friendship between pupils. In other situations, similarities (dissimilarities) are constructed from a data matrix for the objects. These are then called similarity (dissimilarity) coefficients. Several authors, for example Cormack (1971), Jardine and Sibson (1971), Anderberg (1973), Sneath and Sokal (1973), Diday and Simon (1976), Mardia et al. (1979), Gordon (1999), Hubalek (1982), Gower

(1985), Gower and Legendre (1986), Digby and Kempton (1987), Jackson et al. (1989), Baulieu (1989) and Snijders et al. (1990), discuss various similarity and dissimilarity measures together with their associated problems. Table 3.1 lists the more popular dissimilarities for quantitative data, where $\mathcal{X} = [x_{ri}]$ denotes the data matrix obtained for n objects on p variables ($r = 1, \dots, n; i = 1, \dots, p$). The vector for the r th object is denoted by $(x)_r$ and so $\mathcal{X} = [\mathbf{x}_r^T]$. The $\{w_i\}$ are weights, and these and the parameter λ are chosen as required.

When all the variables are binary, it is customary to construct a similarity coefficient and then to transform this into a dissimilarity coefficient with a transformation such as $\delta_{rs} = 1 - s_{rs}$. The measure of similarity between objects r and s is based on Table 3.2. The table shows the number of variables, a , out of the total p variables where both objects score “1”, the number of variables, b , where r scores “1” and s scores “0”, etc. Table 3.3 gives a list of similarity coefficients based on the four counts a, b, c, d . Various situations call for particular choices of coefficients. In practice, more than one can be tried, hoping for some robustness against choice. Hubalek (1982) gives a very comprehensive list of similarity coefficients for binary data.

Table 3.1. Dissimilarity measures for quantitative data

Dissimilarity measure	Formula
Euclidean distance	$\delta_{rs} = \{\sum_i (x_{ri} - x_{si})^2\}^{1/2}$
Weighted Euclidean	$\delta_{rs} = \{\sum_i w_i (x_{ri} - x_{si})^2\}^{1/2}$
Mahalanobis distance	$\delta_{rs} = \{(x_{ri} - \mathbf{x}_r)' \Sigma^{-1} (\mathbf{x}_r - \mathbf{x}_s)\}^{1/2}$
City block metric	$\delta_{rs} = \sum_i x_{ri} - x_{si} $
Minkowski metric	$\delta_{rs} = \{\sum_i w_i x_{ri} - x_{si} ^\lambda\}^{1/\lambda} \quad \lambda \geq 1$
Canberra metric	$\delta_{rs} = \sum_i \frac{ x_{ri} - x_{si} }{x_{ri} + x_{si}}$
Divergence	$\delta_{rs} = \frac{1}{p} \sum_i \frac{(x_{ri} - x_{si})^2}{(x_{ri} + x_{si})^2}$
Bray-Curtis	$\delta_{rs} = \frac{1}{p} \frac{\sum_i x_{ri} - x_{si} }{\sum_i (x_{ri} + x_{si})}$
Soergel	$\delta_{rs} = \frac{1}{p} \frac{\sum_i x_{ri} - x_{si} }{\sum_i \max(x_{ri}, x_{si})}$
Bhattacharyya distance	$\delta_{rs} = \sqrt{\sum_i (\sqrt{x_{ri}} - \sqrt{x_{si}})^2}$
Wave-Hedges	$\delta_{rs} = \sum_i \left(1 - \frac{\min(x_{ri}, x_{si})}{\max(x_{ri}, x_{si})}\right)$
Angular separation	$\delta_{rs} = 1 - \frac{\sum_i x_{ri} x_{si}}{[\sum_i x_{ri}^2 \sum_i x_{si}^2]^{1/2}}$
Correlation	$\delta_{rs} = 1 - \frac{\sum_i (x_{ri} - \bar{x}_r)(x_{si} - \bar{x}_s)}{[\sum_i (x_{ri} - \bar{x}_r)^2 \sum_i (x_{si} - \bar{x}_s)^2]^{1/2}}$

Table 3.2. Summary of binary totals

		Object s	
		1	0
Object r	1	a	b
	0	c	d
		$a + c$	$b + d$
			$p = a + b + c + d$

Table 3.3. Similarity coefficients for Binary Data

Coefficient	Formula
Braun, Blanque	$s_{rs} = \frac{a}{\max\{(a+b), (a+c)\}}$
Czekanowski, Sørensen, Dice	$s_{rs} = \frac{2a}{2a + b + c}$
Hamman	$s_{rs} = \frac{a - (b + c) + d}{a + b + c + d}$
Jaccard coefficient	$s_{rs} = \frac{a}{a + b + c}$
Kulczynski	$s_{rs} = \frac{a}{b + c}$
Kulczynski	$s_{rs} = \frac{1}{2} \left(\frac{a}{a+b} + \frac{a}{a+c} \right)$
Michael	$s_{rs} = \frac{4(ad - bc)}{(a+d)^2 + (b+c)^2}$
Mountford	$s_{rs} = \frac{2a}{a(b+c) + 2bc}$
Mozley, Margalef	$s_{rs} = \frac{a(a+b+c+d)}{(a+b)(a+c)}$
Ochiai	$s_{rs} = \frac{\sqrt{(a+b)(a+c)}}{ad - bc}$
Phi	$s_{rs} = \frac{\sqrt{(a+b)(a+c)(b+d)(c+d)}}{a+d}$
Rogers, Tanimoto	$s_{rs} = \frac{a}{a + 2b + 2c + d}$
Russell, Rao	$s_{rs} = \frac{a}{a + b + c + d}$
Simple matching coefficient	$s_{rs} = \frac{a + d}{a + b + c + d}$
Simpson	$s_{rs} = \frac{a}{\min\{(a+b), (a+c)\}}$
Sokal, Sneath, Anderberg	$s_{rs} = \frac{a}{a + 2(b+c)}$
Yule	$s_{rs} = \frac{ad - bc}{ad + bc}$

For categorical data, agreement scores can be used where, for example, if objects r and s share the same category, then $\delta_{rs} = 0$ and $\delta_{rs} = 1$ if they do not. Other, more elaborate, agreement scores can be devised.

When data are mixed, with binary, quantitative and categorical variables, Gower (1971) suggests using a general similarity coefficient,

$$s_{rs} = \frac{\sum_{i=1}^p w_{rsi} s_{rsi}}{\sum_{i=1}^p w_{rsi}}, \quad (3.1)$$

where s_{rsi} is the similarity between the r th and s th objects based on the i th variable alone and w_{rsi} is unity if the r th and s th objects can be compared on the i th variable and zero otherwise. For quantitative variables, Gower suggests $s_{rsi} = 1 - |x_{ri} - x_{si}| / R_i$, where R_i is the range of the observations for variable i . For presence/absence data, Gower suggests $s_{rsi} = 1$ if objects r and s both score “presence,” and zero otherwise, while $w_{rsi} = 0$ if objects r and s both score “absence,” and unity otherwise. For nominal data Gower suggests $s_{rsi} = 1$ if objects r and s share the same categorization, and zero otherwise.

Metric MDS

3.2

Given n objects with a set of dissimilarities $\{d_{rs}\}$, one dissimilarity for each pair of objects, metric MDS attempts to find a set of points in some space where each point represents one of the objects and the distances between points $\{d_{rs}\}$ are such that

$$d_{rs} = f(\delta_{rs}), \quad (3.2)$$

where f is a continuous parametric monotonic function. The function f can either be the identity function or a function that attempts to transform the dissimilarities to a distance-like form. The first type of metric scaling described here is classical scaling, which originated in the 1930s when Young and Householder (1938) showed that, starting with a matrix of distances between all pairs of the points in a Euclidean space, coordinates for the points could be found such that distances are preserved. Torgerson (1952) brought the subject to popularity using the technique for scaling, where distances are replaced by dissimilarities.

The algorithm for recovering coordinates from distances between pairs of points is as follows:

1. Form matrix $\mathcal{A} = [-\frac{1}{2} \delta_{rs}^2]$.
2. Form matrix $\mathcal{B} = \mathcal{H} \mathcal{A} \mathcal{H}$, where \mathcal{H} is the centring matrix $\mathcal{H} = \mathcal{I} - n^{-1} \mathbf{1}_n \mathbf{1}_n^T$, with $\mathbf{1}_n$ a vector of ones.
3. Find the spectral decomposition of \mathcal{B} , $\mathcal{B} = \mathcal{V} \Lambda \mathcal{V}^T$, where Λ is the diagonal matrix formed from the eigenvalues of \mathcal{B} , and \mathcal{V} is the matrix of corresponding eigenvectors.

4. If the points were originally in a p -dimensional space, the first p eigenvalues of \mathcal{B} are nonzero and the remaining $n - p$ are zero. Discard these from Λ (rename as Λ_1), and discard the corresponding eigenvalues from \mathcal{V} (rename as \mathcal{V}_1).
5. Find $\mathcal{X} = \mathcal{V}_1 \Lambda_1^{1/2}$, and then the coordinates of the points are given by the rows of \mathcal{X} .

As an example, rather than use distances between cities and towns in the UK, the cost of rail travel between all pairs of the following mainland terminus rail stations were used: Aberdeen, Birmingham, Blackpool, Brighton, Dover (Priory), Edinburgh, Inverness, Liverpool, London, Newcastle upon Tyne, Norwich, Plymouth, Sheffield, Southampton, Swansea. Figure 3.1 shows a plot of the stations having obtained the coordinates using the above algorithm. This solution is not unique since any translation, rotation or reflection of the configuration of points will give rise to another solution.

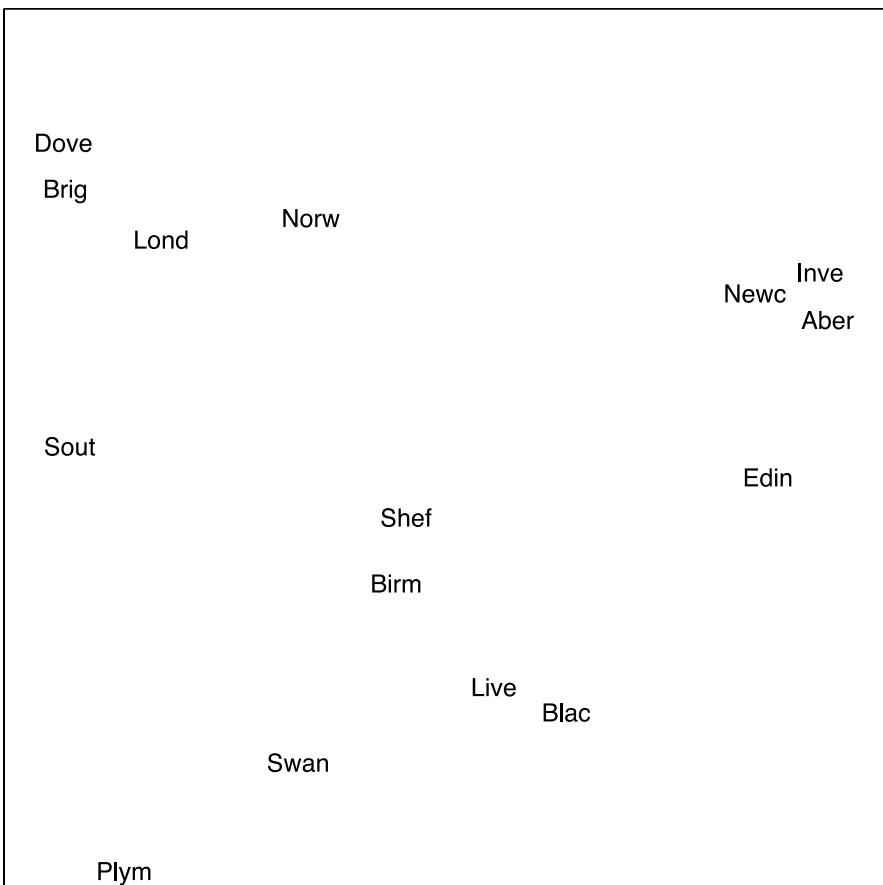


Figure 3.1. A map of rail stations using classical scaling

The plot produces a good representation of the map of the UK. The vertical axis represents West–East, while the horizontal axis runs South–North. It would appear that Newcastle upon Tyne has relocated to Scotland!

Had the exact distances between the rail stations been used in the above (and assuming the UK is in a 2-D Euclidean space!), coordinates would have been found for the stations that would have exactly reproduced the pairwise distances between them. All eigenvalues of \mathcal{B} would have been zero except for the first two. In general, rather than using distances or pseudo-distances between points, classical scaling uses dissimilarities calculated between pairs of objects in place of these distances. The configuration of points obtained in a 2-D space will not usually reproduce the pairwise dissimilarities exactly, but will only approximate them. This implies that nearly all of the eigenvalues of \mathcal{B} are likely to be nonzero, and some might be negative, which will occur if the dissimilarity measure is not a metric. In practice the largest two (positive) eigenvalues and their associated eigenvectors are used for the coordinates of the points. If a 3-D representation is required, then the three largest eigenvalues are used, and so on. A measure of how well the obtained configuration represents the set of pairwise dissimilarities is given by

$$\frac{\sum_{i=1}^p \lambda_i}{\sum_{i=1}^{n-1} |\lambda_i|} \quad \text{or} \quad \frac{\sum_{i=1}^p \lambda_i}{\sum (\text{positive eigenvalues})}. \quad (3.3)$$

Incidentally, if the dissimilarities are calculated as Euclidean distances, then classical scaling can be shown to be equivalent to principal component analysis.

The next example consists of 61 viruses with rod-shaped particles affecting various crops (tobacco, tomato, cucumber and others) recently employed by Ripley (1996) and originally described by Fauquet et al. (1988) and analysed by Eslava-Gomez (1989). There are 18 measurements on each virus, the number of amino acid residues per molecule of coat protein. The whole data set consists of four groups of viruses, Hordeviruses (3), Tobraviruses (6), Tobamoviruses (39) and Furoviruses (13). For brevity the initial four letters of their names will denote the four virus groups. Figure 3.2 shows a classical scaling of the data.

While Tobr and Hord form clear clusters, Furo splits into three clear groups, one of which is similar to Tobr. Similarly Toba forms two groups, one of which is similar to Tobr. The first two eigenvalues have values 6912 and 1956. The sum of all 18 significant eigenvalues is 13 597 out of a potential of 61 values. The first two dimensions correspond to 65% and hence provide a reasonable description of the data.

Another metric scaling approach is to minimize a loss function. For a Sammon map (Sammon 1969), a particular configuration of points with pairwise distances, $\{d_{rs}\}$, representing the dissimilarities $\{\delta_{rs}\}$, has loss function

$$S = \sum_{r < s} \delta_{rs}^{-1} (d_{rs} - \delta_{rs})^2 / \sum_{r < s} \delta_{rs}. \quad (3.4)$$

A configuration is found that has minimum loss using an appropriate optimization method such as a steepest descent method. Other loss functions have also been suggested and used. Figure 3.3 shows a Sammon map for the virus data.

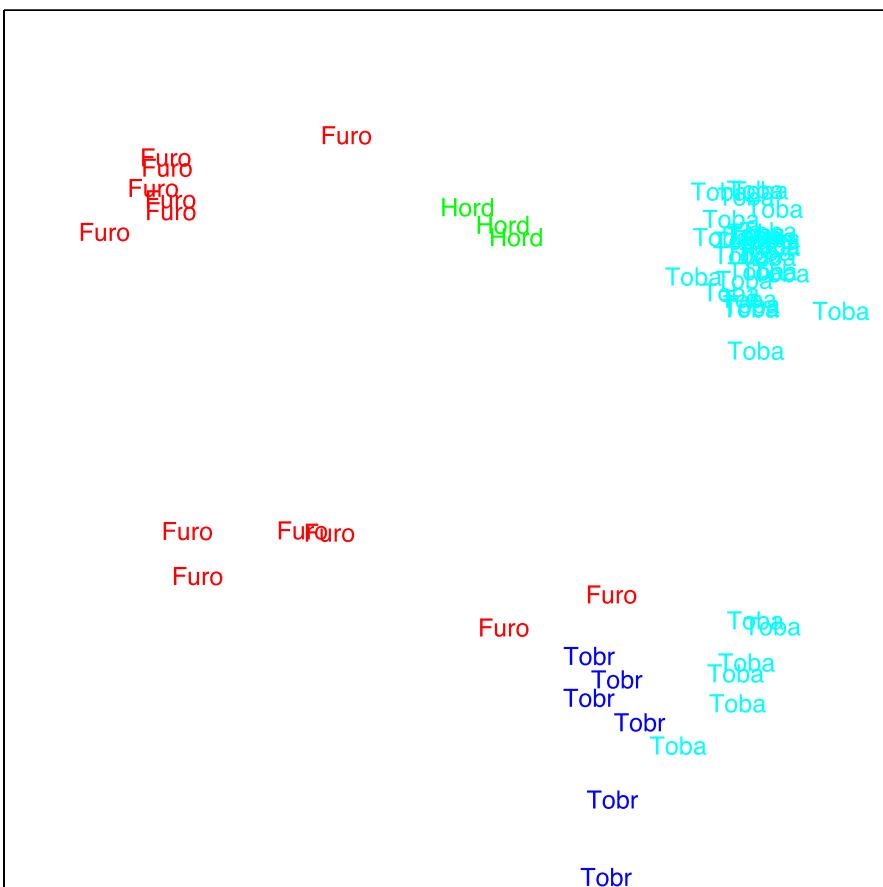


Figure 3.2. Classical scaling of virus data: Hord – Hordeviruses, Tobr – Tobraviruses, Toba – Tobaviruses, Furo – Furoviruses

While the Sammon mapping produces a structure similar to that obtained using classical scaling, the clusters are less clear cut. This differing view probably arises because the mapping is an iterative procedure and is hence dependent on the initial vector selected and the number of iterations performed.

3.3 Non-metric MDS

A non-metric approach to MDS was developed by Shepard (1962a, b) and further improved by Kruskal (1964a, b). In summary, suppose there are n objects with dissimilarities $\{\delta_{rs}\}$. The procedure is to find a configuration of n points in a space, which is usually chosen to be Euclidean, so that each object is represented by a point

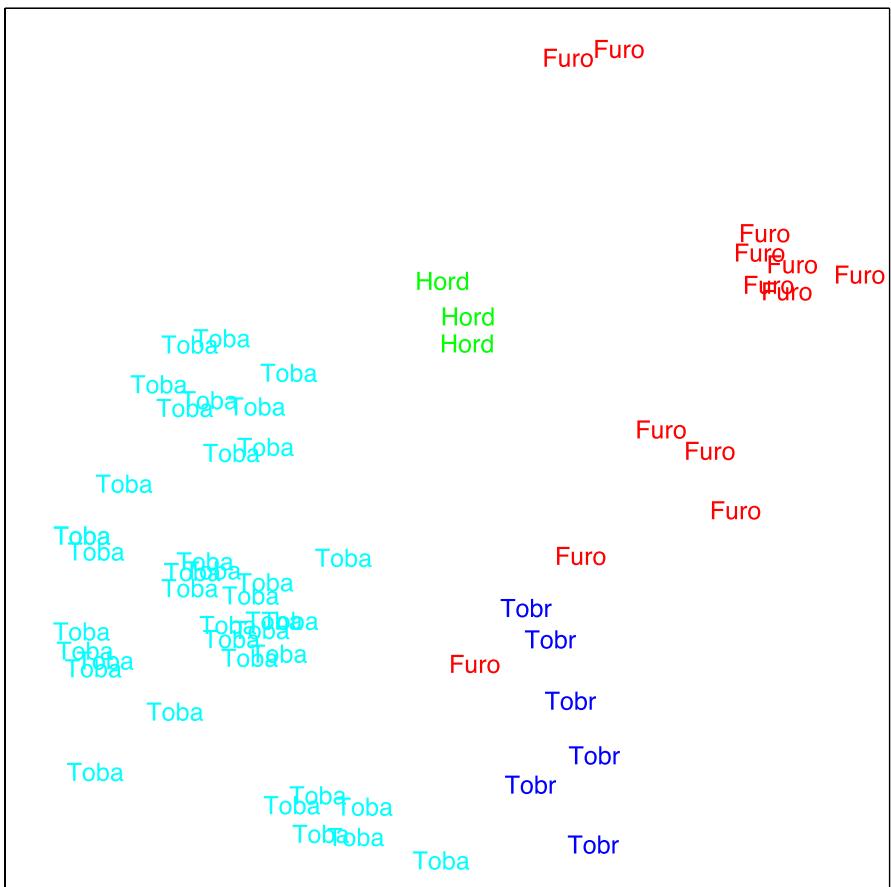


Figure 3.3. Sammon mapping of the virus data: Hord – Hordeviruses, Tobr – Tobraviruses, Toba – Tobaviruses, Furo – Furoviruses

in the space. A configuration is sought so that distances between pairs of points $\{d_{rs}\}$ in the space match “as well as possible” the original dissimilarities $\{\delta_{rs}\}$. Here matching means the rank order of $\{d_{rs}\}$ matches the rank order of $\{\delta_{rs}\}$ as best as possible. The matching of the distances $\{d_{rs}\}$ to the dissimilarities $\{\delta_{rs}\}$ for a particular configuration is measured by the STRESS (S), where

$$S = \sqrt{\frac{\sum_{rs} (\delta_{rs} - \hat{d}_{rs})^2}{\sum_{rs} d_{rs}^2}}. \quad (3.5)$$

Here, $\{\hat{d}_{rs}\}$ is the primary monotone least-squares regression of $\{d_{rs}\}$ on $\{\delta_{rs}\}$, also known as isotonic regression. Details of this regression are not entered into here, but an example can be seen in Fig. 3.5, the Shepard plot. Further details can be found

in Cox and Cox (2001), Borg and Groenen (1997) and elsewhere. A configuration is found that minimizes S , usually using a gradient descent approach.

The rail data used for classical scaling were analysed using non-metric MDS. Figure 3.4 shows the configuration obtained, and again the solution is arbitrary up to translation, rotation and reflection. The STRESS associated with the optimum solution is 10 %. It should be noted that 1000 randomly selected starting points were employed to ensure that the true optimum has been obtained.

A Shepard plot may also be utilized to assess the procedure. This is simply a plot of d_{rs} and \hat{d}_{rs} against δ_{rs} and is shown in Fig. 3.5 for the rail station data. It shows how well the distances within the configuration match the original dissimilarities according to rank order. It makes the monotonic least-squares regression fit particularly clear by joining the δ_{rs} and \hat{d}_{rs} pairs.

The next section gives a more detailed example and shows how the quality of the fit of the model can be investigated.

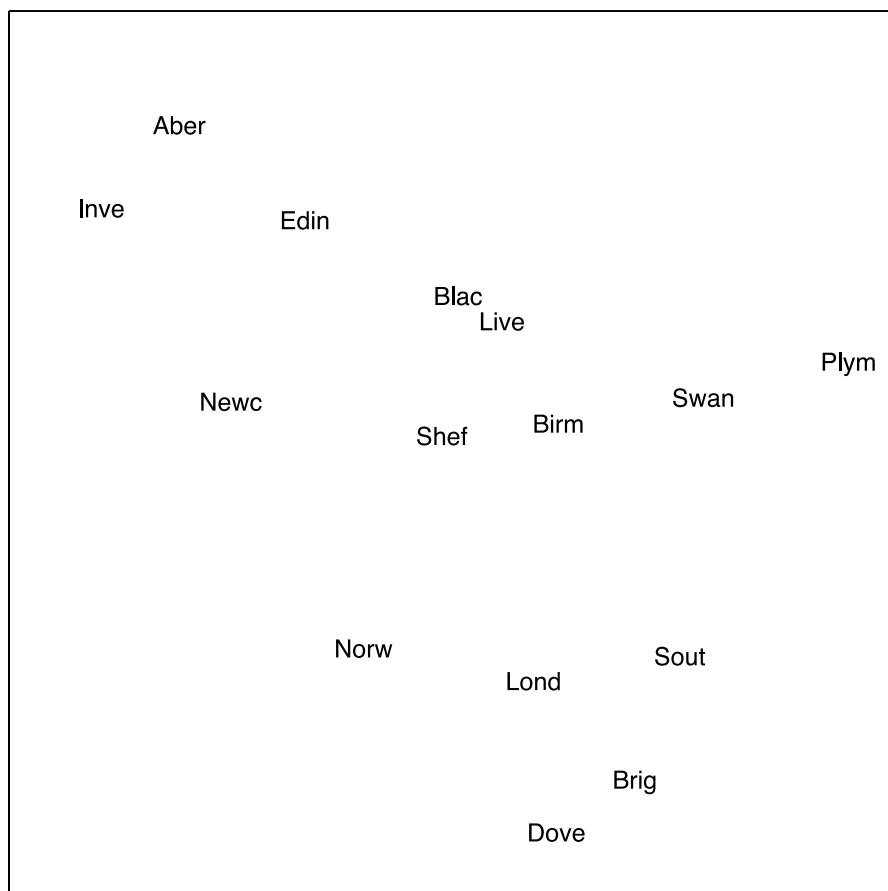


Figure 3.4. A map of rail stations from non-metric MDS

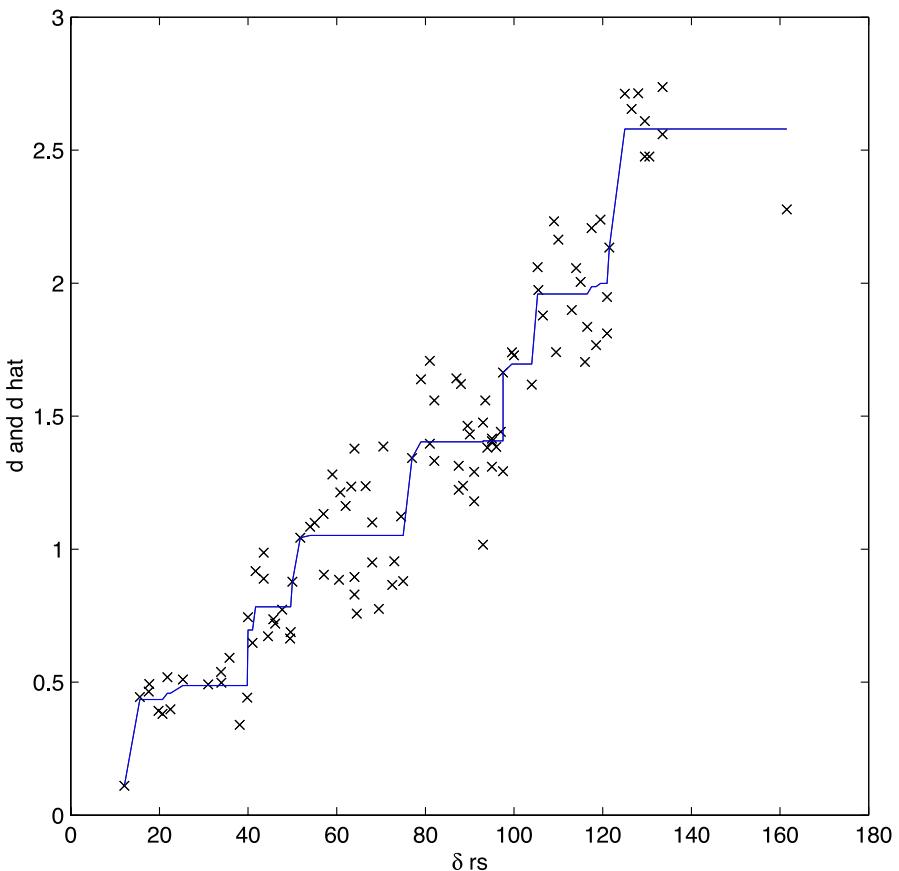


Figure 3.5. The shepard plot for the rail station data

Example: Shakespeare Keywords

3.4

Cox (2005) uses classical scaling on frequency counts of keywords from 20 Shakespeare plays. A similar analysis was carried out but using non-metric scaling with dissimilarity defined by Euclidean distance calculated from the frequencies. Figure 3.6 shows the configuration for the keywords, but where “lord” and “king” have been excluded from the analysis since their inclusion forced all other words into a single cluster producing zero STRESS. The STRESS obtained after exclusion was 11%. It is difficult to plot the configuration in such a small page area, and the only clearly visible words are related to characters plus “god” and “dead.” Figure 3.7 shows the configuration obtained for plays where the role of keywords and plays has been interchanged. The STRESS for this configuration is 10%. It would appear that Hamlet is closest to the historical plays, while the tragedies and comedies are hard to separate.

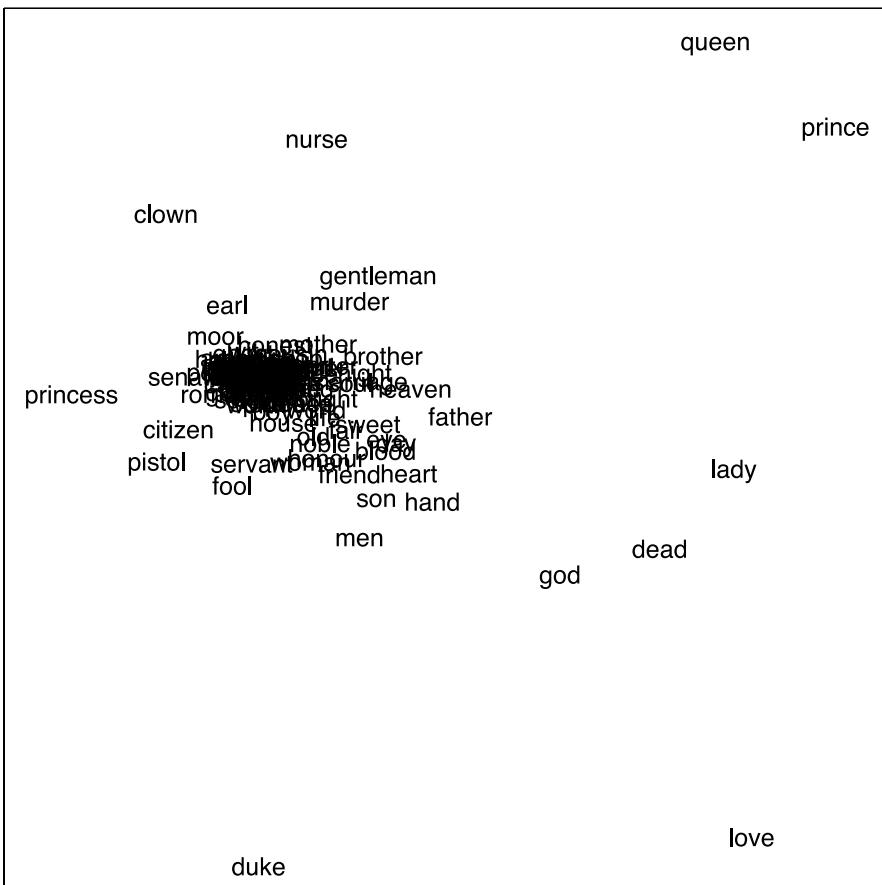


Figure 3.6. Non-metric MDS of Shakespearean keywords

In practice, the number of dimensions necessary for an informative solution is often investigated using a plot of STRESS against number of dimensions (scree plot). This is shown in Fig. 3.8 for the Shakespeare plays and indicates that three or more dimensions may have been preferable. However, then there is the eternal problem of displaying the configuration in just two dimensions.

Although not often carried out in practice, it is possible to investigate the MDS analyses further by looking at outlying or ill-fitting points. Figure 3.9 shows the Shepard plot of d_{rs} and \hat{d}_{rs} against δ_{rs} for the play data. Two points appear removed from the main cluster of points showing large values of $d_{rs} - \hat{d}_{rs}$. These are (tim, h41) and (ham, h41). Figure 3.10 shows a histogram of the values of $d_{rs} - \hat{d}_{rs}$, showing a normal type distribution with a few outliers.

Table 3.4 gives the mean squared differences of $d_{rs} - \hat{d}_{rs}$ for each play (i.e. averaged over s for each r). Henry IV, Part 1 appears discordant with the other plays.

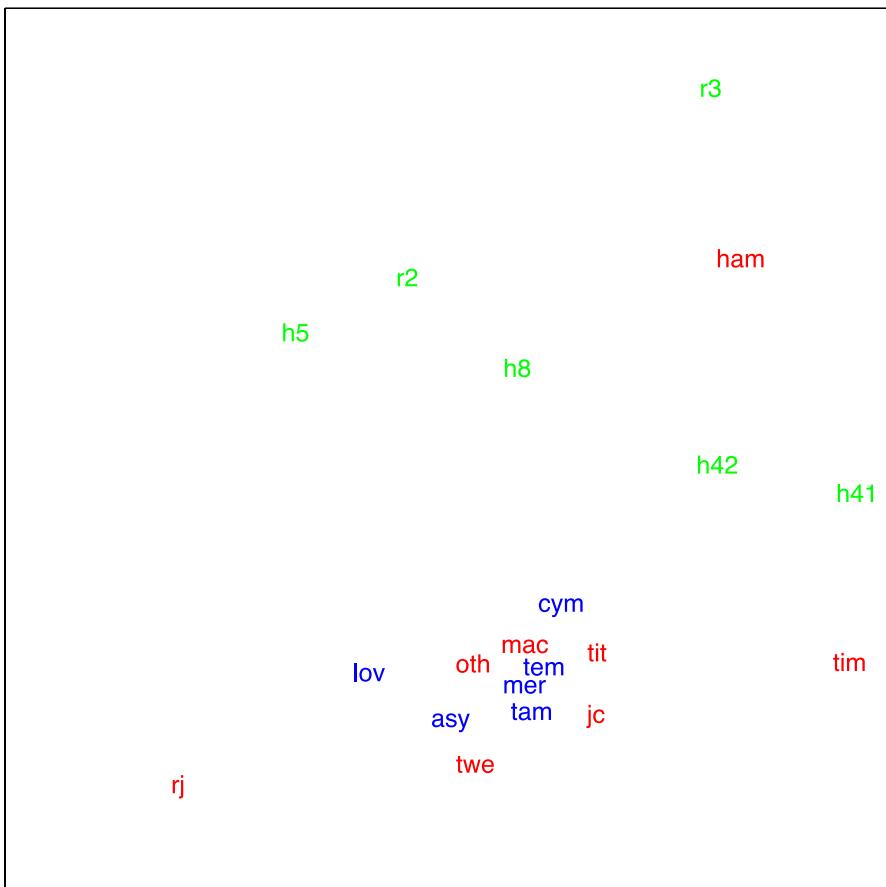


Figure 3.7. Nonmetric MDS of Shakespearean plays: (plays colored *blue* are comedies) asy – As You Like It, cym – Cymbeline, lov – Love's Labours Lost, mer – Merchant of Venice, tam – Taming of the Shrew, tem – The Merchant of Venice; (plays colored *green* are historical plays) h41 – Henry IV, Part 1, h42 – Henry IV, Part 2, h5 – Henry V, h8 – Henry VIII, r2 – Richard II, r3 – Richard III; (plays colored *red* are tragedies) ham – Hamlet, jc – Julius Caesar, mac – Macbeth, oth – Othello, rj – Romeo and Juliet, tit – Titus Andronicus, twe – Twelfth Night

To find the effect of each δ_{rs} on the fitting of the configuration, each δ_{rs} can be left out of the analysis in turn and the STRESS re-calculated. Also, the resulting configuration each time can be matched to the original and the resulting value of the Procrustes analysis (Sect. 5) noted. The lowest STRESS that was obtained was 8.4 % when the dissimilarity between Timon of Athens and Henry IV, Part 1 was removed. The next lowest STRESS was 8.6 % when the dissimilarity between Hamlet and Henry IV, Part 1 was removed. For all other dissimilarities, the STRESS was back to approximately the original value. A similar exercise was carried out removing whole plays at a time. Table 3.5 shows the STRESS values obtained each time a play was removed.

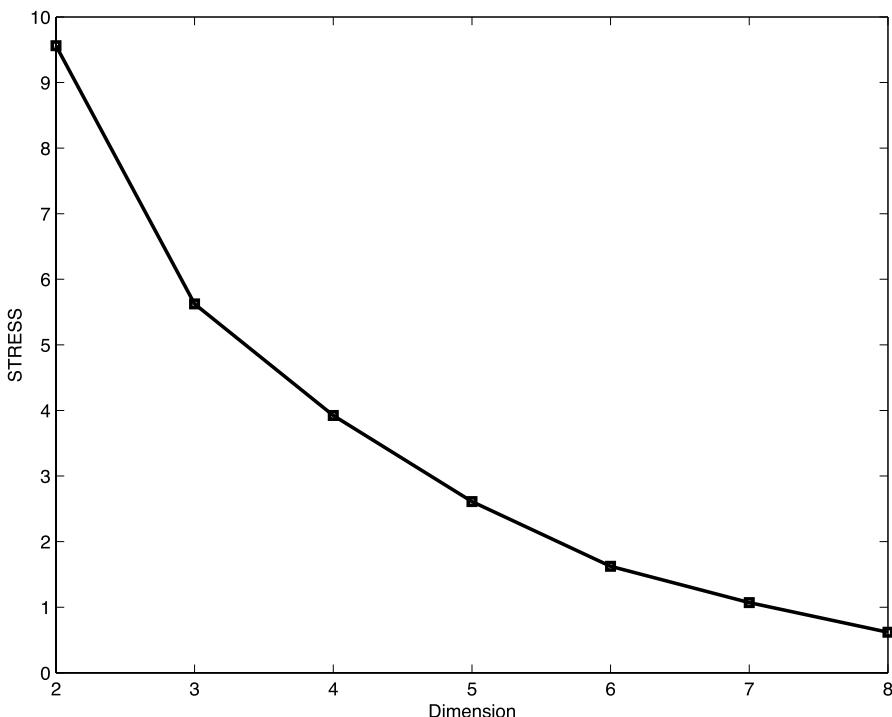


Figure 3.8. Screen plot for Shakespearean data

Table 3.4. Mean $(d_{rs} - \hat{d}_{rs})^2 \times 1000$ for each Shakespearean play

play	mean	play	mean
tam	4.5	r2	14.0
mer	5.3	r3	14.4
tem	7.2	twe	15.3
mac	7.4	lov	19.5
tit	8.8	h5	19.9
asy	9.4	cym	29.0
h8	9.7	rj	29.8
h42	9.8	ham	42.1
oth	10.7	tim	46.5
jc	11.6	h41	69.9

Again if Henry IV, Part 1 is removed, then the STRESS is reduced, showing this play is the worst fitting one. At the other end of the scale, removing Richard III increases the STRESS to 11 %, showing that that play fits the model well.

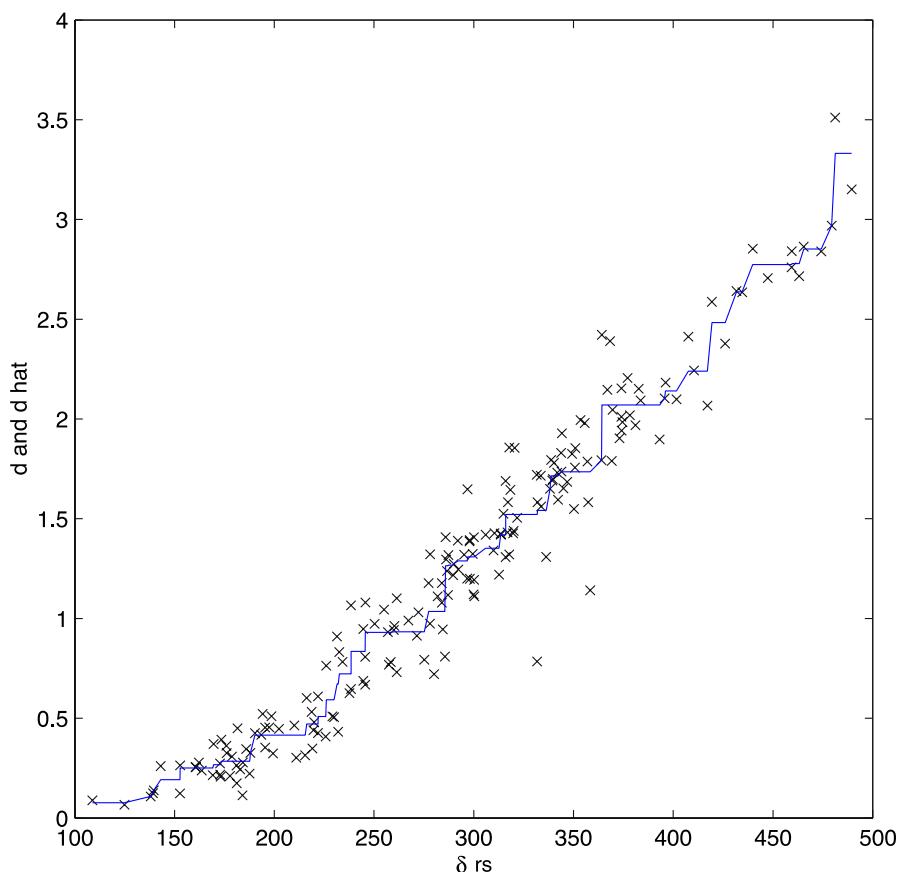
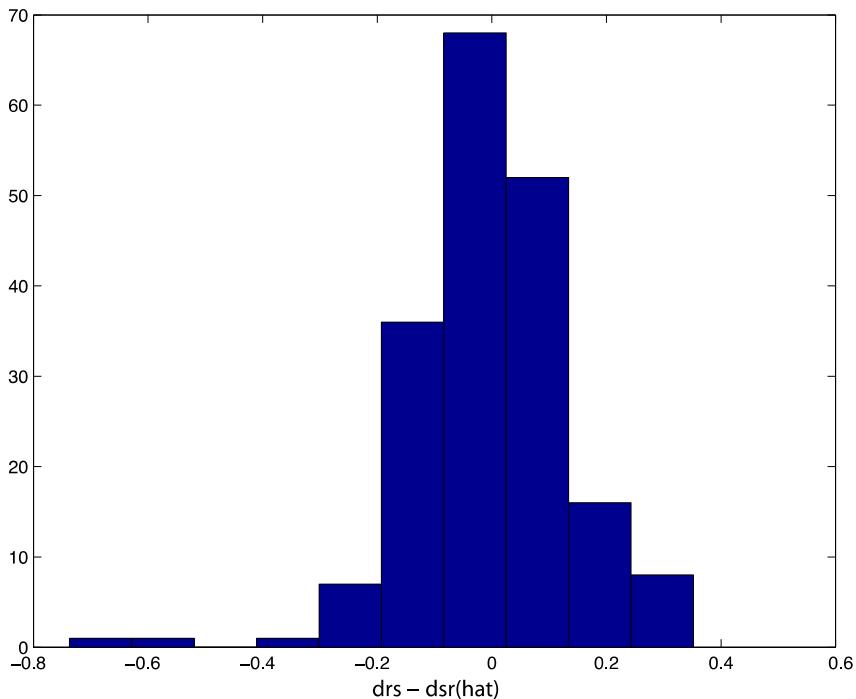


Figure 3.9. Shepard plot of Shakespearean data

Table 3.5. STRESS $\times 100$ obtained when each play is removed in turn

Play	STRESS	Play	STRESS
h4l	7.24	ort	9.52
tim	8.44	h42	9.58
cym	8.61	mac	9.60
ham	8.69	h8	9.61
rj	9.16	tem	9.62
lov	9.23	asy	9.64
h5	9.30	mer	9.70
twe	9.38	r2	9.71
jc	9.48	tam	9.76
tit	9.51	r3	10.76

Figure 3.10. Histogram of $dr_s - \hat{dr}_s$ for Shakespearean data

3.5

Procrustes Analysis

The classical scaling analysis and the non-metric scaling of the terminal train station data produced different, but similar, configurations of points. Since arbitrary translations, rotations and reflections of these configurations give equally valid solutions. In order to make a clear visual comparison of the two, we need to match one configuration with the other. This is achieved using Procrustes analysis. Procrustes analysis finds the isotropic dilation, translation, reflection and rotation that best match one configuration to another. A detailed account of this and allied methods is given by Gower and Dijksterhuis (2004). (According to Greek mythology Procrustes was an innkeeper living near Athens who would subject his guests to extreme measures to make them fit his beds. If they were too short, he would stretch them, or if they were too long, he would cut off their legs.)

Suppose a configuration of n points in a q -dimensional Euclidean space, with coordinates given by the n by q matrix \mathcal{X} , is to be matched to another configuration of points in a p -dimensional Euclidean space ($p \geq q$), with coordinates given by the n by p matrix \mathcal{Y} . Note, it is assumed that the r th point in the X space is in a one-to-one correspondence with the r th point in the Y space. First $p - q$ columns of zeros are added to the end of matrix \mathcal{X} in order to give the matrices the same dimensions.

A measure of the discrepancy between the two configurations is given by the sum of squared distances, R^2 , between corresponding points in the two spaces, i.e.

$$R^2 = \sum_{r=1}^n (\mathbf{y}_r - \mathbf{x}_r)^T (\mathbf{y}_r - \mathbf{x}_r), \quad (3.6)$$

where $\mathcal{X} = [\mathbf{x}_1, \dots, \mathbf{x}_n]^T$, $\mathcal{Y} = [\mathbf{y}_1, \dots, \mathbf{y}_n]^T$ and \mathbf{x}_r and \mathbf{y}_r are the coordinate vectors of the r th point in the two spaces.

The points in the X space are dilated, translated, rotated and reflected to new coordinates, \mathbf{x}' , where

$$\mathbf{x}'_r = \rho \mathcal{A}^T(\mathbf{x})_r + \mathbf{b}, \quad (3.7)$$

ρ is a dilation, \mathcal{A} is an orthogonal matrix giving a rotation and possibly a reflection and \mathbf{b} is a translation. The optimal values of these that minimizes R^2 are summarized in the following procedure:

1. (Optimum translation) Place the centroids of the two configurations at the origin.
2. (Optimum rotation) Find $\mathcal{A} = (\mathcal{X}^T \mathcal{Y} \mathcal{Y}^T \mathcal{X})^{1/2} (\mathcal{Y}^T \mathcal{X})^{-1}$ and rotate \mathcal{X} to $\mathcal{X}\mathcal{A}$.
3. (Optimum scaling) Scale the \mathcal{X} configuration by multiplying each coordinate by $\rho = \text{tr}(\mathcal{X}^T \mathcal{Y} \mathcal{Y}^T \mathcal{X}) / \text{tr}(\mathcal{X}^T \mathcal{X})$.
4. Calculate the Procrustes statistic

$$R^2 = 1 - \{\text{tr}(\mathcal{X}^T \mathcal{Y} \mathcal{Y}^T \mathcal{X})^{1/2}\}^2 / \text{tr}(\mathcal{X}^T \mathcal{X}) \text{tr}(\mathcal{Y}^T \mathcal{Y}). \quad (3.8)$$

The value of R^2 can be between 0 and 1, where 0 implies a perfect matching of the configurations. The larger the value of R^2 , the worse the match.

Procrustes analysis was used on the cost of rail travel data. Figure 3.11 shows the non-metric scaling result (Fig. 3.4) matched to the metric (Fig. 3.1). In this case the Procrustes statistic is 0.06, showing that the point configurations are remarkably similar.

Extensions to basic Procrustes analysis of matching one configuration to another include weighting of points and axes, the allowance of oblique axes and the matching of more than two configurations; see Cox and Cox (2001) or Gower and Dijksterhuis (2004) for a detailed account of the area.

Unidimensional Scaling

3.6

When the space in which the points representing objects or individuals has only one dimension, the scaling technique becomes that of unidimensional scaling. The loss function to be minimized is

$$S = \sum_{r < s} (\delta_{rs} - |x_r - x_s|)^2. \quad (3.9)$$

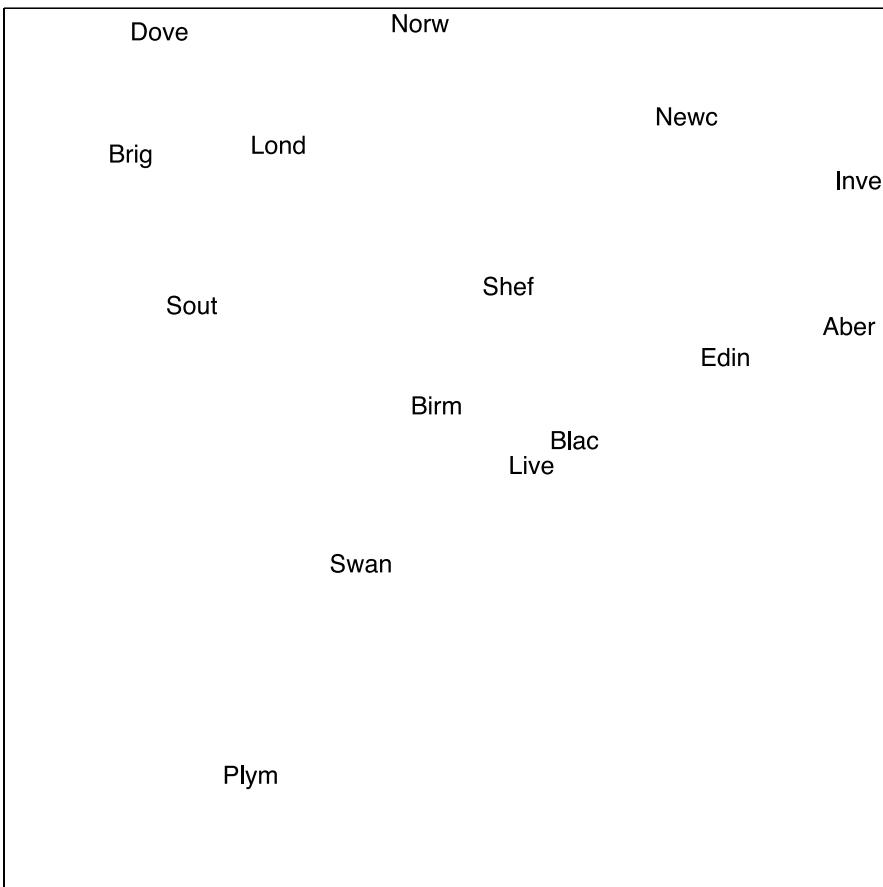


Figure 3.11. Procrustes rotation of metric onto non-metric scaling for rail cost

Minimizing S can be difficult because of the possible large number of local minima. Various algorithms have been proposed, for example Guttman (1968), Hubert and Arabie (1986, 1988) and Lau et al. (1998). Here we use the algorithm by Guttman (1968) for the following example.

The data used to illustrate this method are ratings for World War II politicians and are the lower triangle given by Everitt and Dunn (1983). Figure 3.12 shows the resulting unidimensional scaling obtained.

For clarity, the points have been plotted on a diagonal, which represents a linear axis starting from Mao Tse Tung at 6.75 and ending at Mussolini at 5.67. The countries the leaders represent have also been added. What is interesting is that Stalin is most closely identified with Hitler and Mussolini as opposed to his UK/US World War II allies.

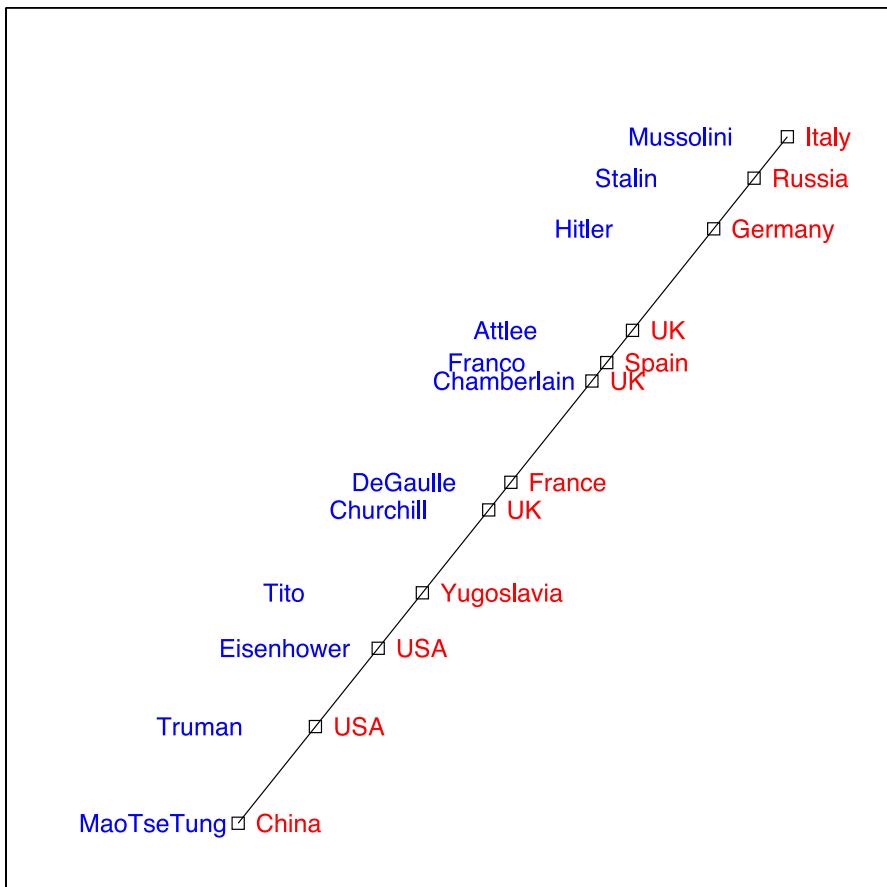


Figure 3.12. Unidimensional scaling of World War II political leaders

INDSCAL

3.7

Suppose data consist of several sets of dissimilarities between objects, the same objects in each case. For example, several panellists assess the dissimilarities between all pairs of a set of products. MDS could be applied to each panelist's data resulting in many configurations. A better approach might be to combine the dissimilarities in some manner.

Carroll and Chang (1970) proposed a metric model comprising two spaces: a group stimulus space and a subject's (or individual's) space, both chosen to be of the same dimension. Points in the group stimulus space represent the objects or stimuli and form an "underlying" configuration. The individuals are represented as points in the subject's space. The coordinates of each individual are the weights required to give the weighted Euclidean distances between the points in the stimulus space, the values

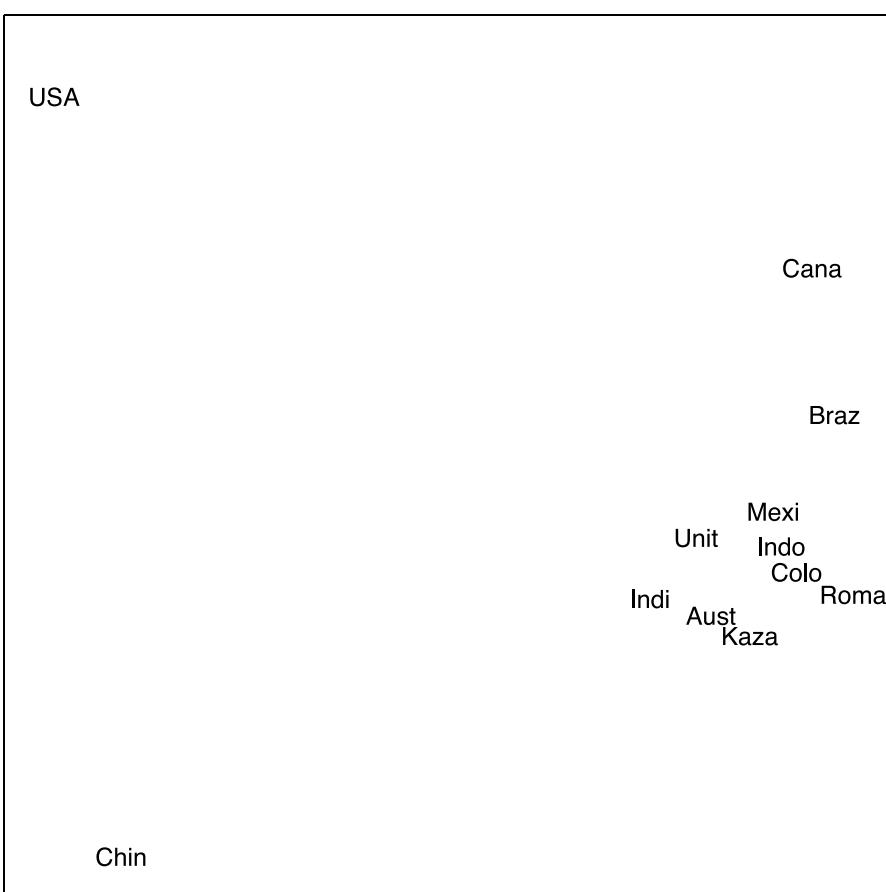


Figure 3.13. INDSCAL for BP data by country: Aust – Australia, Indo – Indonesia, Braz – Brazil, Kaza – Kazakhstan, Cana – Canada, Mexi – Mexico, Chin – China, Roma – Romania, Colo – Colombia, Unit – United Kingdom, Indi – India, USA – USA

Table 3.6. Energy source codes for the BP data

Energy source	code
Coal – Consumption in millions of tonnes of oil equivalent	ColC
Coal – Production in millions of tonnes of oil equivalent	ColP
Hydro – Consumption in millions of tonnes of oil equivalent	Hydr
Natural Gas – Consumption in millions of tonnes of oil equivalent	GasC
Natural Gas – Production in millions of tonnes of oil equivalent	GasP
Nuclear – Consumption in millions of tonnes of oil equivalent	NucC
Oil – Consumption in millions of tonnes	OilC
Oil – Production in millions of tonnes	OilP

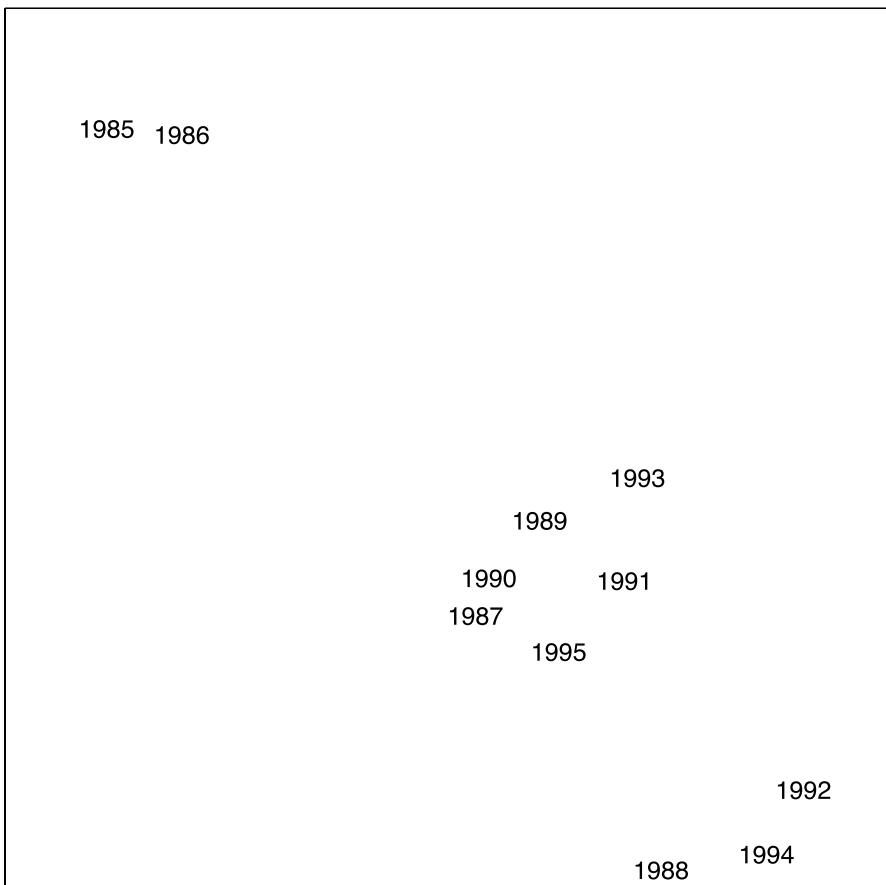


Figure 3.14. INDSCAL for BP data by year

which best represent the corresponding dissimilarities for that individual. Hence the acronym INDSCAL – INdividual Differences SCALing.

Let there be n objects under study and N subjects producing the dissimilarities. Let the dimension of the spaces be p and the points in the group stimulus space be denoted by x_{rt} ($r = 1, \dots, n$; $t = 1, \dots, p$). Let the dissimilarity between objects r and s for the i th subject be $\delta_{rs,i}$ and the points in the subjects' space have coordinates w_{it} ($i = 1, \dots, N$; $t = 1, \dots, p$). Then the weighted Euclidean distance between the r th and s th points for the i th subject is

$$d_{rs,i} = \left[\sum_{t=1}^p w_{it} (x_{rt} - x_{st})^2 \right]^{1/2}. \quad (3.10)$$

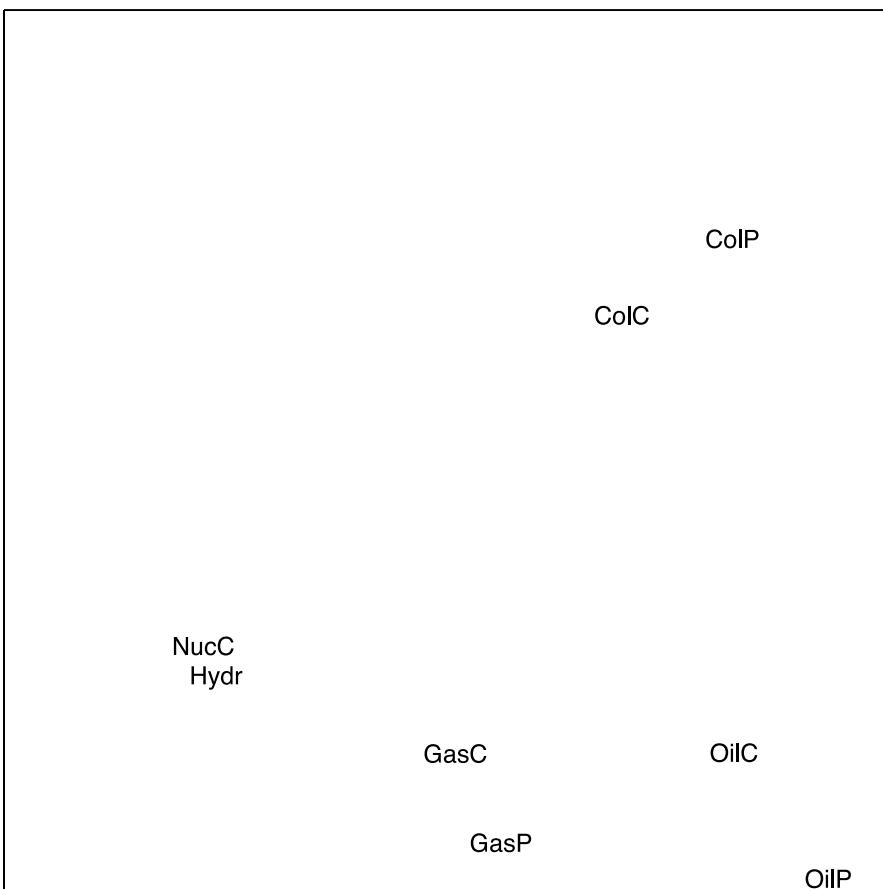


Figure 3.15. INDSCAL for BP data by data source

The individual weights $\{w_{it}\}$ and coordinates $\{x_{rt}\}$ are sought which best match $\{d_{rs,i}\}$ to $\{\delta_{rs,i}\}$. Carroll and Chang (1970) give an algorithm which uses a recursive least-squares approach to do this.

The data used to illustrate INDSCAL are from the 1995 edition of the BP Statistical Review of World Energy. The review incorporates additional elements from the BP Review of World Gas. The review is a compendium of statistics on the primary forms of energy (BP 1996).

The data are for all years from 1985 to 1995, with energy sources as shown in Table 3.6. Data are available for both production and consumption.

Initially dissimilarities were generated for countries and each year, averaging over the energy sources. The INDSCAL analysis results are given in Figs. 3.13 and 3.14. Figure 3.13 shows the “group stimulus space” and Fig. 3.14 the “subjects space”.

Clearly China and the USA are exceptional. The USA is the largest consumer/producer of gas and oil, and also the largest consumer of nuclear. In coal (both produc-

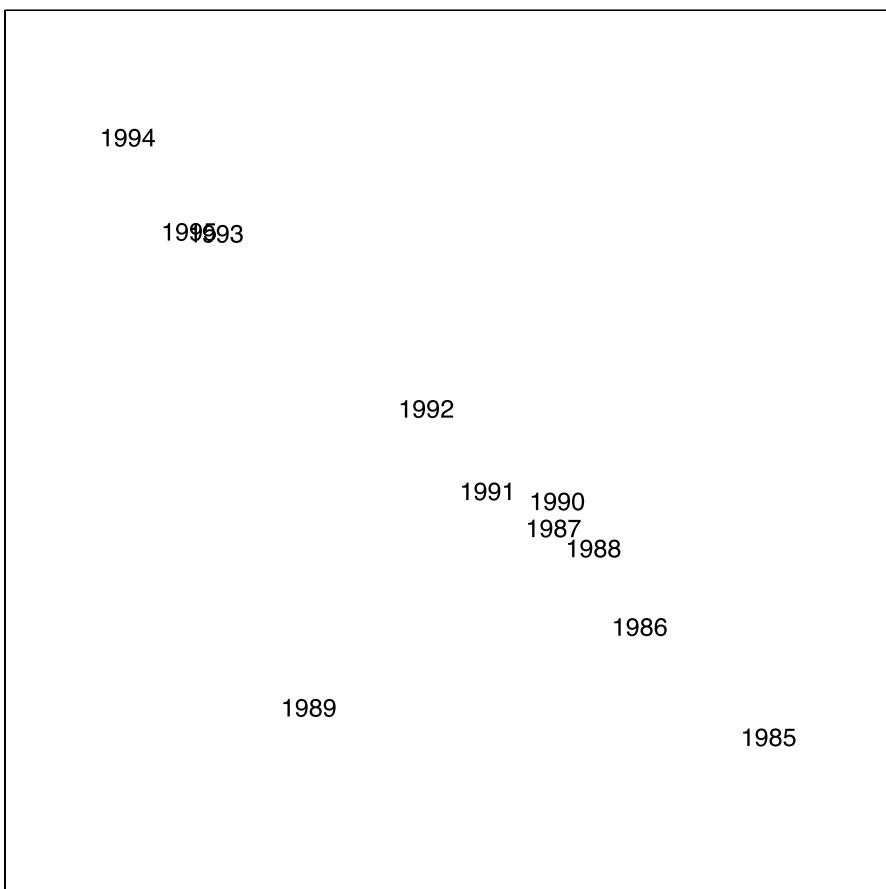


Figure 3.16. Example plots from INDSCAL for BP data by year

tion and consumption) the USA and China are very close and significantly higher than the other countries considered.

The years 1985 and 1986 are exceptional; these correspond to a marked percentage increase in the consumption/production of coal by Indonesia.

A similar analysis may be conducted based on averaging over the countries. The resulting plots are shown in Figs. 3.15 and 3.16.

Clearly the production and consumption of each energy source are very close in the plot showing consumption is highly linked to production. What is surprising is the coincidence of nuclear and hydroelectric energy.

A Procrustes statistic was employed to compare Figs. 3.14 and 3.16, giving a value of 0.59, suggesting the plots are dissimilar. In Fig. 3.16 the years, with slight adjustments, are in sequential order 1994, 1995, 1993, 1992, 1991, 1990, 1987, 1988, 1986, 1985, the exception being 1989.

3.8 Correspondence Analysis and Reciprocal Averaging

Correspondence analysis represents the rows and columns of a two-way contingency table as points in Euclidean spaces of dimension usually chosen as two. It can also be used on any data matrix that has non-negative entries and can also be extended to higher way tables, but this aspect is not covered here. The technique will be illustrated using the contingency table displayed in Table 3.7, which records the number of papers in a research database which employ specific keywords in their descriptive fields (title, keywords or abstract) over an 11-year period.

First distances are measured between the rows of the table. Ordinary Euclidean distance, treating the 11 columns as 11 dimensions for the rows, is not appropriate, since doubling the size of the sample for any one row will alter the Euclidean distance dramatically. For contingency tables where only the overall total is fixed, this may be not be a problem, but it will be for tables where row totals can be chosen arbitrarily. To overcome this problem, χ^2 distances are used. The χ^2 distance, $d_{ii'}$, between the i 'th and i' 'th rows is defined by

$$d_{ii'}^2 = \sum_{j=1}^p \frac{1}{c_j} \left(\frac{x_{ij}}{r_i} - \frac{x_{i'j}}{r_{i'}} \right)^2, \quad (3.11)$$

where x_{ij} is the entry in the table in the i 'th row and j 'th column, r_i is the i 'th row sum and c_j is the j 'th column sum. A space is now found where points in the space represent the rows of the table and where Euclidean distance between points equals the χ^2 distance between the corresponding rows of the table. Greenacre (1984) gives a comprehensive account of how this space is found, but only a brief summary can be given here.

Let \mathcal{X} denote the table that has been normalized to have overall total equal to unity. Let \mathcal{D}_r be the diagonal matrix of the row sums of \mathcal{X} , and let \mathcal{D}_c be the diagonal matrix of column sums of \mathcal{X} . Let the generalized singular value decomposition of \mathcal{X} be given by

$$\mathcal{X} = \mathcal{A}\mathcal{D}_\lambda\mathcal{B}^T, \quad (3.12)$$

Table 3.7. References employing keywords

Keywords	94	95	96	97	98	99	00	01	02	03	04	Total
Reciprocal av.	0	0	4	0	3	3	0	0	0	3	3	16
Correspond. anal.	144	171	186	219	237	246	241	243	278	314	310	2589
Ind. diff. scal.	8	4	6	3	5	4	4	4	3	1	3	45
Classical scaling	5	6	7	5	2	10	7	7	4	5	6	64
Procrustes anal.	15	16	29	20	36	43	41	32	40	70	67	409
Mult. scaling	75	117	125	107	109	137	145	147	166	191	195	1514
Total	247	314	357	354	392	443	438	433	491	584	584	4637

where $\mathcal{A}^T \mathcal{D}_r^{-1} \mathcal{A} = \mathcal{B}^T \mathcal{D}_c^{-1} \mathcal{B} = \mathcal{I}$. Then the space and coordinates for the row points are given by $\mathcal{D}_r^{-1} \mathcal{A} \mathcal{D}_\lambda$. Note there is a “trivial dimension” which has to be ignored which corresponds to the singular value of unity with singular vector a vector of ones. The Euclidean distances between the points in the $\mathcal{D}_r^{-1} \mathcal{A} \mathcal{D}_\lambda$ space equal the corresponding χ^2 distances between the rows of the table. However, this space has dimension equal to one less than the number of columns. As an approximation, only the first two singular values and corresponding singular vectors are used (ignoring the trivial dimension). Similarly, a space for the columns can be found as $\mathcal{D}_c^{-1} \mathcal{B} \mathcal{D}_\lambda$, and distances between points in this space equal the χ^2 distances between corresponding columns in the table.

Figure 17 shows the space for the rows of the contingency table, and Fig. 3.18 the space for the columns of the table.

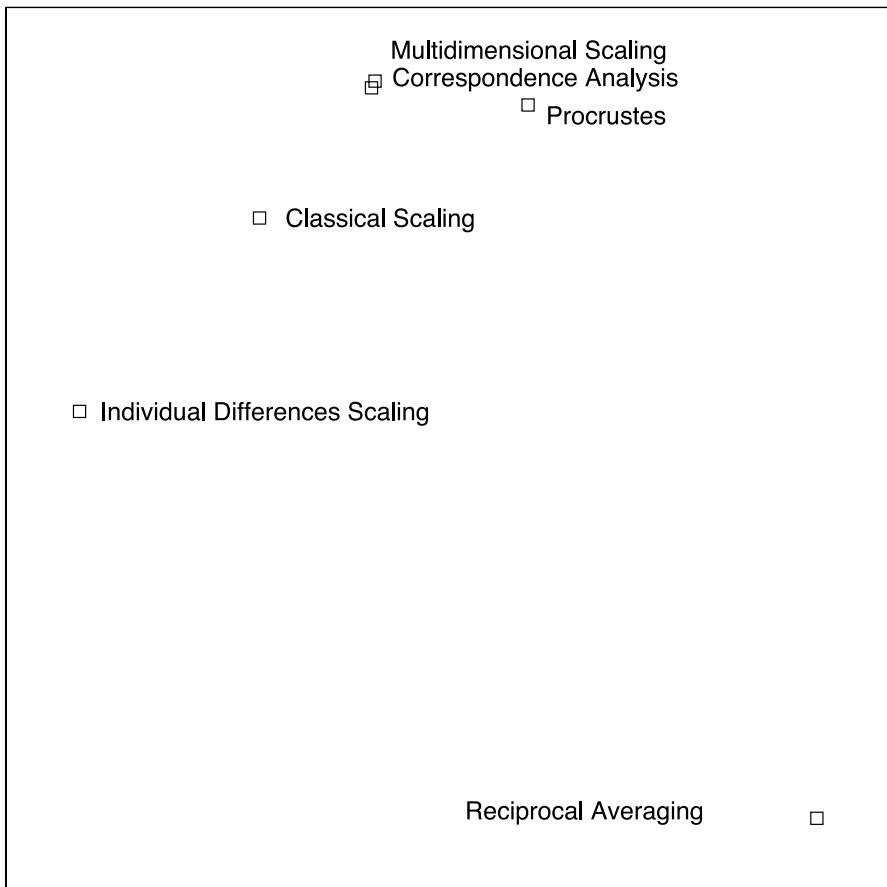


Figure 3.17. Keyword plot for references

The keyword (row) plot shows the similar popularity of correspondence analysis and multidimensional scaling and how Procrustes analysis is related to them, while the difference in use of reciprocal averaging, which is used sporadically, becomes clear. Sometimes the row and column spaces are combined into one space since they have both arisen from the singular value decomposition of \mathcal{X} . Distances between row points and column points are not defined, although row points close to column points will have some association.

The year plot is consistent with a steady increase in the use of correspondence analysis, Procrustes analysis and multidimensional scaling. Note that the years to the lower right of the plot are the years that reciprocal averaging makes an entry into the table.

Reciprocal averaging is used for binary data and is essentially the same as correspondence analysis, although the construction appears different. It can be easily explained using an ecological example. Suppose n different species of plants are each

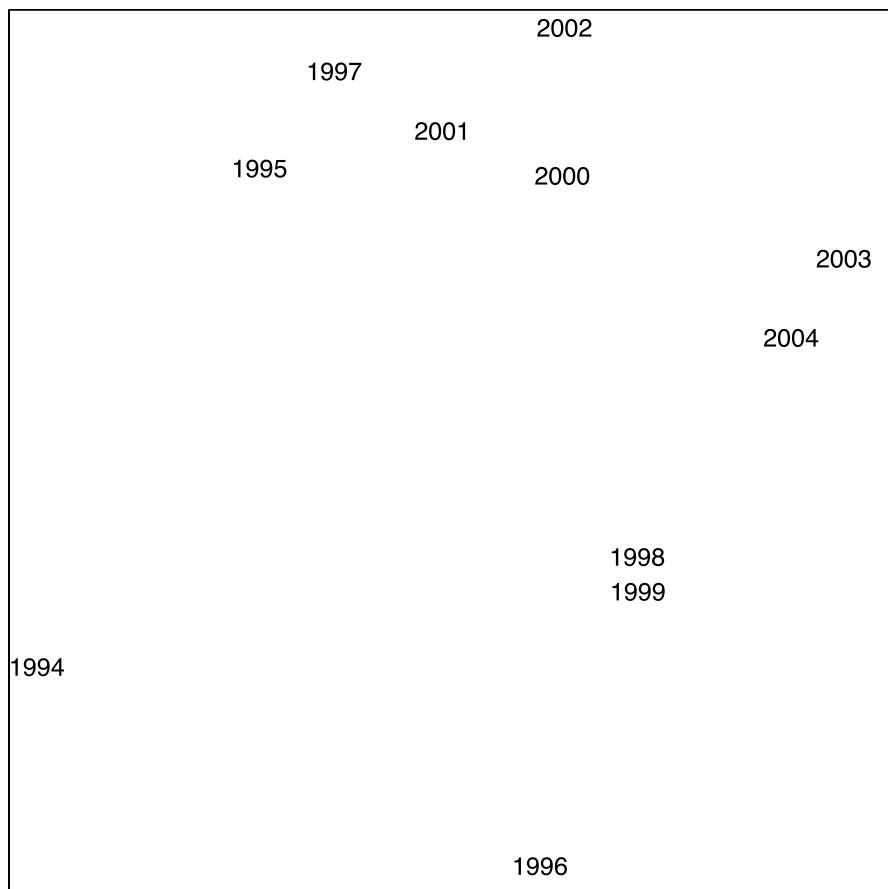


Figure 3.18. Year plot for references

planted at p different sites which vary in exposure to the weather. Let $x_{ij} = 1$ if the i th species survives at the j th site, and zero if it does not. Let u_i be a hardiness score for the i th species and v_j an exposure score for the j th site. It is assumed that the exposure score at the j th site is proportional to the mean hardiness score of the species at that site. Thus

$$v_j \propto \sum_i u_i x_{ij} / \sum_i x_{ij}. \quad (3.13)$$

Similarly, it is assumed that the hardiness score of species i is proportional to the mean exposure score of the sites occupied by that species. Thus

$$u_i \propto \sum_j v_j x_{ij} / \sum_j x_{ij}. \quad (3.14)$$

Let $r_i = \sum_j x_{ij}$, $c_j = \sum_i x_{ij}$. Reciprocal averaging solves the equations

$$\rho u_i = \sum_j v_j x_{ij} / r_i \quad (i = 1, \dots, n), \quad (3.15)$$

$$\rho v_j = \sum_i u_i x_{ij} / c_j \quad (j = 1, \dots, p), \quad (3.16)$$

where ρ is a scaling parameter, to obtain the hardiness and exposure scores.

Reciprocal averaging was used on the Shakespeare data used in Sect. 4, but where it has been turned into a binary format scoring 0/1 if a word is absent/present in each of the plays. The resulting plots are shown in Figs. 3.19 and 3.20.

As in Fig. 3.6 it is the personal words (princess, bishop, clown, moor etc.) that convey most information about the play.

When examining the plays, losing the detail (the word count for each play) has clearly affected the detail displayed, although certain similarities within the plots can be seen, for instance the cluster asy, cym, tam, tem, mer seen in Fig. 3.20 is within a larger cluster within Fig. 3.7.

Large Data Sets and Other Numerical Approaches

3.9

Behind most MDS techniques there is a need for accurate and efficient algorithms for minimizing functions, but many MDS programs and algorithms cannot cope with very large data sets, as they suffer from high computational complexity. They cannot feasibly be applied to data sets over a few thousand objects in size. However, methods have been proposed to overcome this problem, for example Fast Spring Model-Visualisation (FSMvis). FSMvis adopts a novel hybrid approach based upon stochastic sampling, interpolation, and spring models. Following Morrison et al. (2003) the mechanics of the spring model are described.

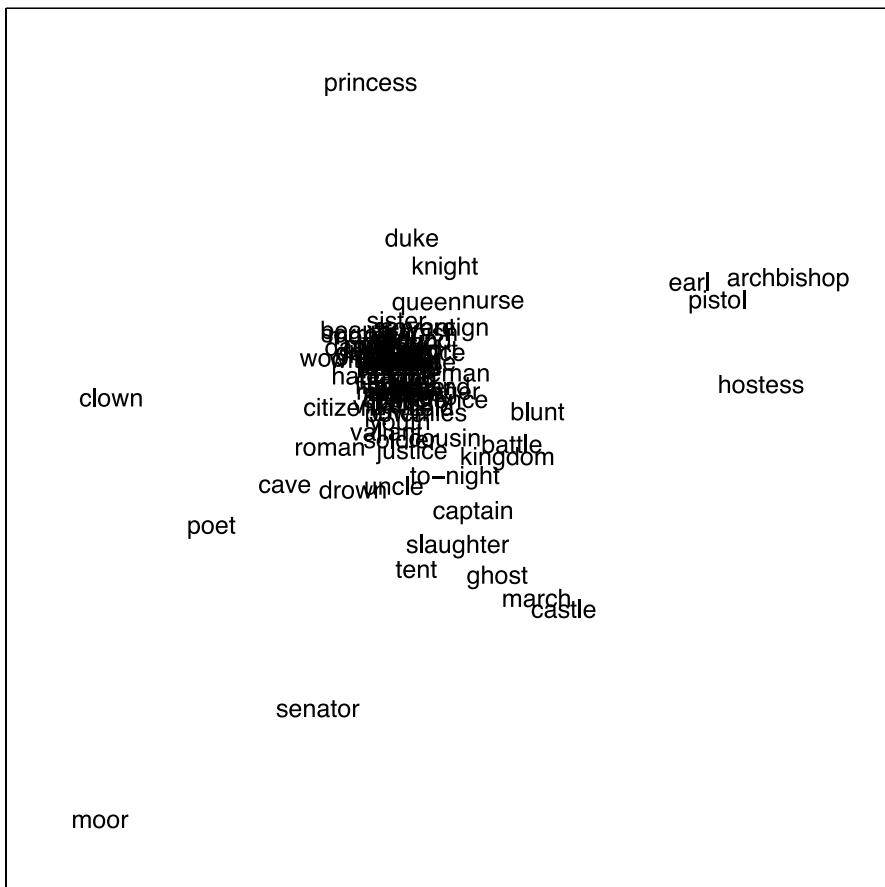


Figure 3.19. Reciprocal averaging of Shakespearean keywords

The concept of the “spring model” comes from work by Eades (1984) on a heuristic approach for graph drawing. Eades described an approach for the layout of a general graph through the physical analogy of a system of steel rings connected by springs. A graph consists of vertices and edges, and here each vertex represents one of the objects under consideration. The graph may be represented by a mechanical system on replacing the vertices by rings and the edges by springs. Each relaxed spring length or “rest distance” is set to be the dissimilarity measured between the corresponding objects. Initially the vertices/rings are placed in random positions and the springs connecting them are either stretched or compressed. When the system is released, the forces exerted by the springs move the system to equilibrium, and presumably to a state of minimal energy. The algorithm employed is iterative, with each iteration refining the layout of the graph.

This procedure was applied to dissimilarities calculated for a population of women who were at least 21 years old, of Pima Indian heritage and living near Phoenix, AZ,

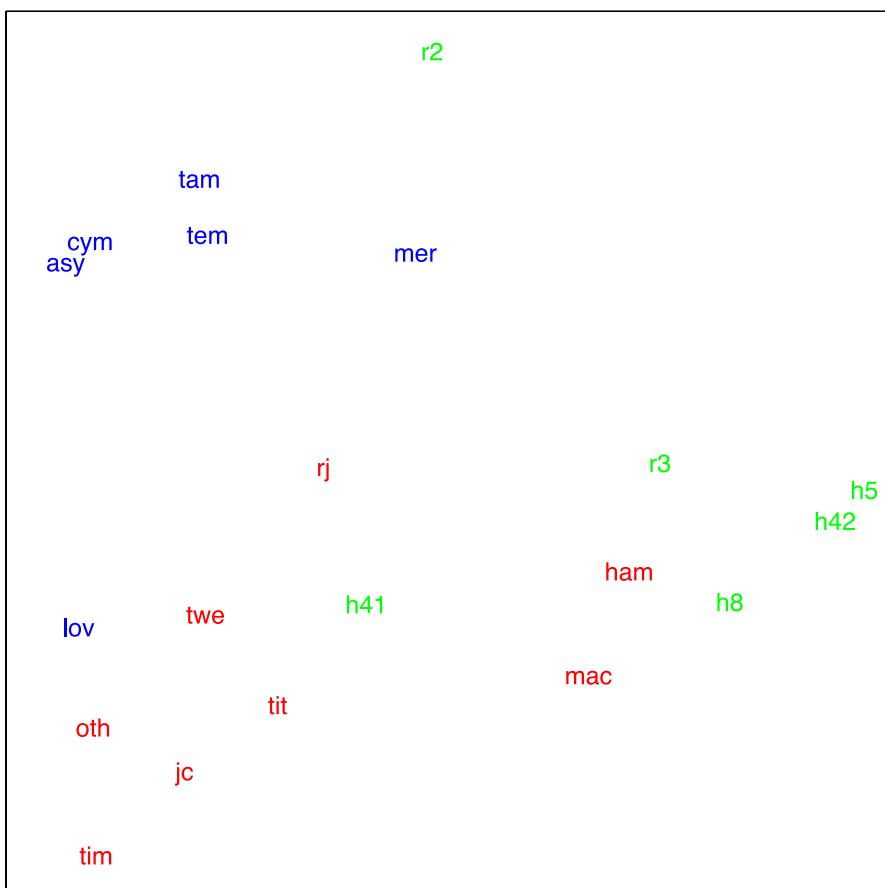


Figure 3.20. Reciprocal averaging of Shakespearean plays

and who were tested for diabetes according to World Health Organization criteria. The results are displayed in Fig. 3.21. The data were collected by the US National Institute of Diabetes and Digestive and Kidney Diseases (Hettich et al., 1998). The raw data consisted of 8 measurements on 768 individuals. The method successfully partitions the 768 individuals into the 2 groups.

Agrafiotis et al. (2001) present a family of algorithms that combine non-linear mapping techniques using neural networks. The method employs an algorithm to project a small random sample and then “learns” the underlying transform using one or more multilayer perceptrons. This approach captures the non-linear mapping relationship as an explicit function and allows the scaling of additional patterns as they become available, without the need to reconstruct the entire map. The approach is particularly useful for extracting low-dimensional Cartesian coordinate vectors from large binary spaces, such as those encountered in the analysis of large chemical data sets. Molecular similarity is used to analyse chemical phenomena and can

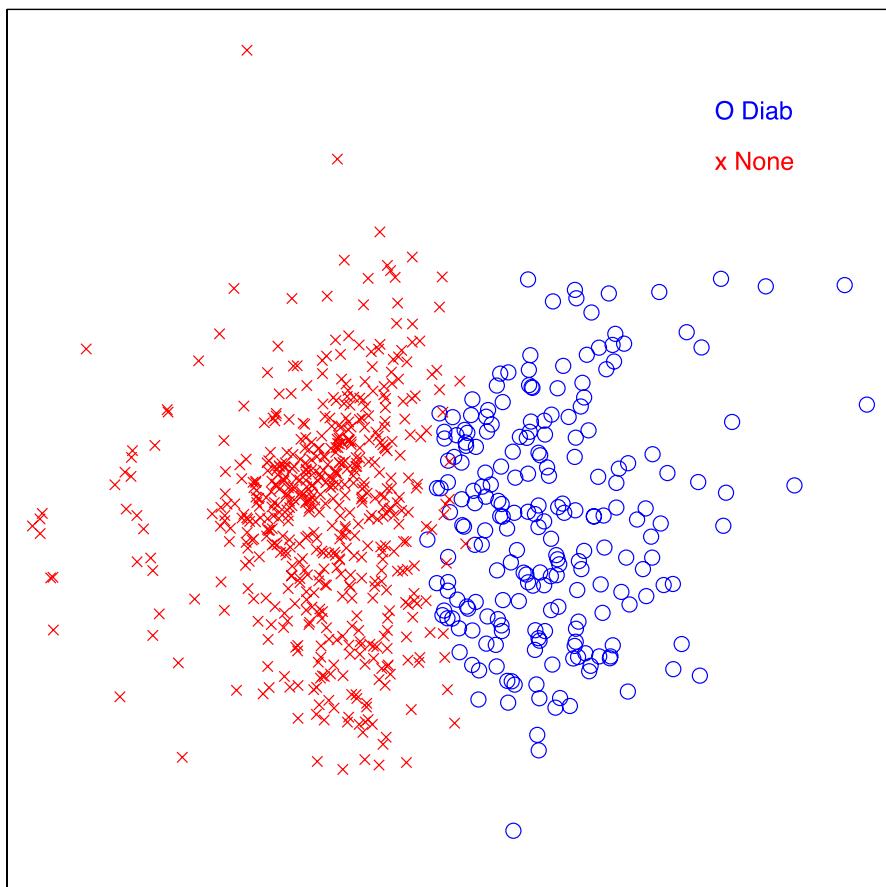


Figure 3.21. Example plot from FSMvis for Pima Indian data

aid in the design of new chemical entities with improved physical, chemical and biological properties. As a guide the authors report analysing one billion items in around 4 h.

Genetic algorithms are an approach to optimization suggested by the biological process of evolution driven by natural selection. The aim is to derive a parameter set that minimizes the difference between a model's expected values and those observed from the data. For detailed reviews see Charbonneau (1995) and Schmitt (2001). The procedure employed for our comparisons uses the algorithm developed by Charbonneau and Knapp (2005). As expected, when tested on the rail journey cost data, the results were indistinguishable. In principle this approach admits larger data sets, but not necessarily of sufficient size to make the approach worthwhile.

Simulated annealing is a suitable approach for large-scale optimization problems. It is claimed to be ideal for locating an ideal global minimum located among a number of local minima. This method has been employed to address the famous travelling

salesman problem. The approach borrows its philosophy from theoretical physics. By analogy the system is raised to a high temperature in which the individuals move freely with respect to each other. As cooling occurs this mobility is lost. Some alignment occurs between the individuals, and the system approaches a minimum energy state. To try to achieve the desired global minimum, the cooling must occur slowly. This procedure has been encapsulated within published algorithms (Goffe et al., 1994; Corana et al., 1987). The approach differs from the conventional gradient descent method in that an occasional step away from an apparent minimum might be used to assist in escaping from local minima.

The majorization approach to minimization was first proposed by de Leeuw (1977) for use with MDS. Essentially, a complicated function $f(x)$ is replaced by a more manageable auxiliary function $g(x, y)$ such that for each x in the domain of f , $f(x) \leq g(x, y)$, for a particular y in the domain of g , and also so that $f(y) = g(y, y)$. The function g is called the majorizing function. An initial value x_0 is used and then $g(x, x_0)$ is minimized with respect to x . Let the value of x , which gives rise to the minimum, be x_1 . Then $g(x, x_1)$ is minimized with respect to x , and so on until convergence.

References

- Agrafiotis, D.K., Rassokhin, D.N. and Lobanov, V.S. (2001). *J Comp Chem* 22:488–500
- Anderberg, M.R. (1973). *Cluster Analysis for Applications*. Academic, New York
- Baulieu, F.B. (1989). *J Classificat* 6:233–246
- Borg, I. and Groenen, P.G. (1997). *Modern Multidimensional Scaling*. Springer, New York
- BP (1996). <http://www.bp.com>
- Carroll, J.D. and Chang, J.J. (1970) Analysis of individual differences in multidimensional scaling via an n -way generalization of “Eckart–Young” decomposition. *Psychometrika* 35:283–319
- Charbonneau, P. (1995). *Astrophys J Suppl Ser* 101:309–334
- Charbonneau, P. and Knapp, B. (2005). <http://downloadhaoucaredu/archive/pikaia/>
- Corana, A., Marchesi, M., Martini, C. and Ridella, S. (1987). *ACM Trans Math Softw* 13:262–280
- Cormack, R.M. (1971). *J R Stat Soc A* 134:321–367
- Cox, T.F. (2005). *An Introduction to Multivariate Data Analysis*. Hodder Arnold, London
- Cox, T.F. and Cox, M.A.A. (2001). *Multidimensional Scaling*. Chapman & Hall/CRC, Boca Raton, FL
- de Leeuw, J. (1977). Applications of convex analysis to multidimensional scaling. In: Barra, J.R., Brodeau, F., Romier, G., van Cutsem, B. (eds) *Recent Developments in Statistics*. North Holland, Amsterdam
- Diday, E. and Simon, J.C. (1976). Clustering analysis. In: Fu, K.S. (ed) *Communication and Cybernetics 10 Digital Pattern Recognition*. Springer, Berlin Heidelberg New York

- Digby, P.G.N. and Kempton, R.A. (1987). *Multivariate Analysis of Ecological Communities*. Chapman and Hall/CRC, London
- Eades, P. (1984). *Congressus Numerantium* 42:149–160
- Eslava-Gomez, G. (1989). *Projection pursuit and other graphical methods for multivariate Data*. DPhil Thesis, University of Oxford, Oxford
- Everitt, B.S. and Dunn, G. (1983). *Advanced Methods of Data Exploration and Modelling*. Heinemann, London
- Fauquet, C., Desbois, D., Fargette, D. and Vidal, G. (1988). Classification of furoviruses based on the amino acid composition of their coat proteins. In: Cooper, J.I., Asher, M.J.C. (eds) *Viruses with Fungal Vectors*. Association of Applied Biologists, Edinburgh
- Goffe, W.L., Ferrier, G.D. and Rogers, J. (1994). *J Econometr* 60:65–99
- Gordon, A.D. (1999). *Classification*, 2nd edn. Chapman and Hall/CRC, London
- Gower, J.C. (1971). *Biometrics* 27:857–874
- Gower, J.C. (1985). Measures of similarity, dissimilarity and distance. In: Kotz, S., Johnson, N.L., Read, C.B. (eds) *Encyclopedia of Statistical Sciences*. vol 5. Wiley, New York
- Gower, J.C. and Legendre, P. (1986). *J Classificat* 3:5–48
- Gower, J.C. and Dijksterhuis, G.B. (2004). *Procrustes Problems*. Oxford University Press, Oxford
- Greenacre, M.J. (1984). *Theory and Application of Correspondence Analysis*. Academic, London
- Guttman, L. (1968). *Psychometrika* 33:469–506
- Hettich, S., Blake, C.L. and Merz, C.J. (1998). *UCI Repository of machine learning databases*. <http://www.ics.uci.edu/~mlearn/MLRepository.html>
- Hubalek, Z. (1982). *Biol Rev* 57:669–689
- Hubert, L. and Arabie, P. (1986). Unidimensional scaling and combinatorial optimisation. In: de Leeuw, J., Heiser, W.J., Meulman, J., Critchley, F. (eds) *Multidimensional Data Analysis*. DSWO, Leiden
- Hubert, L. and Arabie, P. (1988). Relying on necessary conditions for optimization: unidimensional scaling and some extensions. In: Bock, H.H. (ed) *Classification and Related Methods of Data Analysis*. North Holland, Amsterdam
- Jackson, D.A., Somers, K.M. and Harvey, H.H. (1989). *Am Nat* 133:436–453
- Jardine, N. and Sibson, R. (1971). *Mathematical Taxonomy*. Wiley, London
- Kruskal, J.B. (1964a). *Psychometrika* 29:1–27
- Kruskal, J.B. (1964b). *Psychometrika* 29:115–129
- Lau, K.N., Leung, P.L. and Tse, K.K. (1998). *J Classificat* 15:3–14
- Mardia, K.V., Kent, J.T. and Bibby, J.M. (1979). *Multivariate Analysis*. Academic, London
- Morrison, A., Ross, G. and Chalmers, M. (2003). *Inf Visual* 2:68–77
- Ripley, B.D. (1996). *Pattern Recognition and Neural Networks*. Cambridge University Press, Cambridge
- Sammon, J.W. (1969). *IEEE Trans Comput* 18:401–409
- Schmitt, L.M. (2001). *Theor Comput Sci* 259:1–61
- Shepard, R.N. (1962a). *Psychometrika* 27:125–140

-
- Shepard, R.N. (1962b). *Psychometrika* 27:219–246
- Sneath, P. and Sokal, R. (1973). *Numerical Taxonomy*. Freeman, San Francisco
- Snijders, T.A.B., Dormaar, M., van Schuur, W.H., Dijkman-Caes, C. and Driessen, G. (1990). *J Classificat* 7:5–31
- Torgerson, W.S. (1952). *Psychometrika* 17:401–419
- Young, G. and Householder, A.S. (1938). *Psychometrika* 3:19–22
- Young, F.W. and Hamer, R.M. (eds) (1987). *Multidimensional Scaling: History, Theory and Applications*. Lawrence Erlbaum, Hillsdale, NJ

Huge Multidimensional Data Visualization: Back to the Virtue of Principal Coordinates and Dendrograms in the New Computer Age

III.4

Francesco Palumbo, Domenico Vistocco, Alain Morineau

4.1	<i>Introduction</i>	351
4.2	<i>The Geometric Approach to the Statistical Analysis</i>	352
	Distance and Metric Space	353
	OECD Countries Dataset.....	354
4.3	<i>Factorial Analysis</i>	355
	Principal Component Analysis	356
4.4	<i>Distance Visualization in \mathbb{R}^P</i>	360
	Hierarchical Clustering	362
4.5	<i>Principal Axis Methods and Classification: a Unified View</i>	365
4.6	<i>Computational Issues</i>	365
	Partitioning Methods	366
	Mixed Strategy for Very Large Datasets	368

4.7	<i>Factorial Plans and Dendrograms: the Challenge for Visualization</i>	371
	The Use of Visual Variables for Designing Useful Representations	372
	Interactivity Within and Between Graphics.....	375
4.8	<i>An Application:</i>	
	<i>the Survey of Italian Household Income and Wealth</i>	377
4.9	<i>Conclusion and Perspectives</i>	383

Introduction

4.1

Many papers refer to Tukey's (1977) treatise on exploratory data analysis as the contribution that transformed statistical thinking. In actual fact, new ideas introduced by Tukey prompted many statisticians to give a more prominent role to data visualization and more generally to data. However, J.W. Tukey in 1962 had already begun his daring provocation when at the annual meeting of the Institute of Mathematical Statistics he gave his talk entitled "The Future of Data Analysis" (Tukey, 1962).

At the same time, the French statistician J.P. Benzécri brought his paradigm to the attention of the international scientific community in his paper "L' Analyse des Données" (Benzécri, 1976). As with Tukey's ideas, it appeared totally revolutionary with respect to the "classical" statistical approaches for two reasons: *i*) the absence of any *a priori* model and *ii*) the prominent role of graphical visualization in the analysis of output. Unfortunately most of Benzécri's papers were written in French. Michael Greenacre, in the preface to his well-known book *Theory and Application of Correspondence Analysis* (Greenacre, 1984), wrote: "*In 19'80 I was invited to give a paper on correspondence analysis at an international conference on multidimensional graphical methods called 'Looking at Multivariate Data' in Sheffield, England. [There] ... I realized the tremendous communication gap between Benzécri's group and the Anglo-American statistical school.*"

These simultaneous and independent stimuli for statistical analysis mainly based on visualization did not occur by chance but as a consequence of extraordinary developments in information technology. In particular, technological innovations in computer architecture permitted the storage of ever larger volumes of data and allowed one to obtain even higher-quality graphical visualization (on screen and paper). These two elements contributed to giving a prominent role to data visualization. The growth of data volume, on the other hand, determined the need for preliminary (exploratory) analyses; graphical methods quickly proved their potential in this kind of analysis. The performance of graphics cards permitted one to obtain more detailed visualization, and the developments in dynamic and 3-D graphics have opened new frontiers.

A posteriori we can state that at that time statisticians became conscious of the potential of graphical visualization and of the need for exploratory analysis. However, it appears quite strange that these two *giants* of the statistics world, Tukey and Benzécri, are very rarely mentioned together in data analysis papers. Their common starting point was the central role of data in statistical analysis; both of them were strong believers that, in the future, the amount of available data would increase tremendously, although the current abundance of data might be more than even they expected!

In light of this historical background, the title of the present contribution should appear more clear to the reader. Our idea is to present visualization in the modern computer age following the precepts of data analysis theorists. Moreover, note that the basic principles of data analysis are inspired by the elementary notions of geometry.

A key element in the success of data analysis is the strong contribution of visualization: it exploits the human capability to *perceive* the 3-D space. On the other hand, the role of the geometric approach in mathematics has a centuries-old story. Let us take into account that many theorems were first enunciated in geometric notation and mathematically formalized many, perhaps hundreds of years later. The Pythagorean theorem is a well-known example.

Our perception of the real world is the result of a geometric space characterized by orthogonal axes, the concept of distance, and the effects of light. The combination of the first two elements defines a metric space. Cartesian spaces permit one to visualize positions of a set of dimensionless points. Exploiting capabilities of current graphics cards acting on brightness, points are enriched by point markers, characterized by different sizes, shapes, and colors, that add information, helping the user interpret results more easily and quickly.

Our mathematics based on the decimal system is clearly the result of having ten fingers; similarly, it is obvious that our geometry, Euclidean geometry, is based on a system of orthogonal axes due to our perception of the horizon line. As the binary and hexadecimal numerical systems represent possible alternatives to the decimal system, similarly there exist different geometries based on nonorthogonal systems where parallel lines converge in a finite space. However, even if alternative geometries exist, Euclidean geometry remains the only geometry that we apply in the solution of real-world problems.

The concepts of *far* and *close* are native concepts. It is not necessary to be a mathematician to understand them. Distance represents the measure of closeness in space.

This contribution will introduce the concept of factorial space and of dendograms; it intends to furnish guidelines for giving a correct representation of displayed data. It will also show how it is possible to obtain enhanced representation where, thanks to modern graphics cards, it is possible to obtain millions of colors, transparencies, and man-machine interactions.

4.2 **The Geometric Approach to the Statistical Analysis**

There are several reasons to revisit principal coordinates and dendograms. When these methods appeared 30 years ago, the capabilities of computer devices were very poor. This lack of capability in obtaining satisfactory graphical representations led to ingenious and original solutions. Text ASCII printers were used to draw factorial plans using the (base) ASCII character set, and 132-column printers were preferred to 80-column printers to obtain wider and clearer representations. In some papers from that time we also find hand-drawn factorial plans and dendograms. Nevertheless, the data analysis approach prospered despite these technical difficulties and even more data analysis methods appeared in the specialized literature. The reasons for this success, presumably, lie behind the very easy interpretation keys. In fact, interpretation is based on the notion of distance, which is a human concept.

Nowadays many graphical tools permit one to enhance basic information with many additional elements that transform a 2-D representation into a higher-dimensional representation.

Wainer and Velleman (2001) noted that some concepts of geometry are inconsistently imported in the statistical literature. However, in spite of differences in definitions and notations, statisticians and mathematicians use visualization techniques for the same goal: to reveal (hidden) relationships in the data structure (also known as fitting in statistics).

In the next subsection we separately introduce the notions of Cartesian space, distance, and metric space. These are the foundations of principal coordinates and dendrograms. Then, in the following sections, we present innovative visualization techniques that can be realized by combining the *old* and the *new*.

Distance and Metric Space

4.2.1

The method statisticians use to collect multivariate data is the *data matrix*. An $X_{n,p}$ data matrix represents a set of n multidimensional observations, described by a set of p variables. From a geometric point of view, the generic row vector \mathbf{x}_i (where $i = 1, \dots, n$) represents a point in the \mathbb{R}^p Cartesian space. Of course the alternative representation consists in representing p points (variable) in the n -dimensional space.

According to Euclidean geometry, Cartesian space refers to a couple of ordered and oriented orthogonal axes, which admits a definition of a unit measure. In the year 300 B.C. approximately, the Greek mathematician Euclid formalized the mathematical knowledge of his time. His book *Elements* is considered the second most popular book in history, after the Holy Bible. The greatness of his contribution is demonstrated by the fact that most of our current knowledge in geometry is in so-called *Euclidean geometry*.

However, we must wait for Descartes for the formalization of the isomorphism between algebraic and geometric structures. In fact, what we call a Cartesian space was first introduced by Newton several years later.

The interest and the long story behind the definition of Cartesian space demonstrates the importance of such a mathematical concept.

Given a set Ω , any function $d : \Omega \times \Omega \rightarrow \mathbb{R}^+$ satisfying the following three properties is a distance:

- $d(\mathbf{X}_i, \mathbf{X}_j) = 0 \Leftrightarrow \mathbf{X}_i = \mathbf{X}_j$
- $d(\mathbf{X}_i, \mathbf{X}_j) = d(\mathbf{X}_j, \mathbf{X}_i)$ (*symmetry*)
- $d(\mathbf{X}_i, \mathbf{X}_j) \leq d(\mathbf{X}_i, \mathbf{X}_h) + d(\mathbf{X}_h, \mathbf{X}_j)$,
where $\{\mathbf{X}_i, \mathbf{X}_h, \mathbf{X}_j\} \in \Omega$ (*triangular inequality*)

We will call Ω a *metric space* if its elements define a distance.

Looking at the properties of distance, we can easily understand the reasons that induced statisticians to turn to indexes having the properties of distance. It is important to note that the most commonly used methods satisfy special cases of minimal distance.

The need to visualize over Cartesian axes these distances appeared when, thanks to the capabilities of modern computers (and more specifically to the advent of the personal computer), statisticians began to treat simultaneously many variables and thousands of statistical units.

Some examples of the problems arising when plotting p -dimensional data will be illustrated with the help of the *OECD Countries* dataset. The last sections of the paper will show possible solutions when dealing with huge datasets.

The human mind can conceive, but not imagine, graphical representations in orthogonal spaces having more than three dimensions.

To overcome this limitation the use of *dendograms* permits one to visualize the distances among a set of statistical units belonging to an \mathbb{R}^p space ($\forall p \geq 1$).

The assumption of a system of coordinates permits one to define the concept of distance. Many common statistical indexes are rigorously bound to the notion of distance.

4.2.2

OECD Countries Dataset

The data consist of six short-term macroeconomic indicators provided by the Organisation for Economic Cooperation and Development (OECD). In particular, the performance indicators are:

GDP: gross domestic product

LI: leading indicator (a composite indicator based on other indicators of economic activity)

UR: unemployment rate

IR: interest rate

TB: trade balance

NNS: net national savings

The above indicators are observed on 20 OECD countries and are listed in Table 4.1. After considering many available and well-known datasets, we decided to use the Vichi and Kiers (2001) dataset. In spite of its small dimensionality, OECD data are distinguished by features that can be explored using a visual approach. Illustrative examples from this dataset will allow us to appreciate the capabilities and limits of multidimensional data analysis. Distortion-free distances displaying any dataset variables are limited to 2-D Cartesian space: *only* in this case do distances on the map correspond to actual distances. Any representations involving more than two variables imply distance distortion and information loss.

Methods presented in the following sections share the same goal: *to visualize the correspondence within a set of variables and difference within a set of multivariate statistical units in terms of distance*. Firstly, attention is devoted to factorial methods that permit one to linearly combine a set of variables in a subset of *latent* variables; secondly a section is dedicated to hierarchical clustering methods and to other clustering methods allowing for the representation of the difference between (multivariate) statistical units in terms of distance.

Table 4.1. Six macroeconomic performance indicators of 20 OECD countries (percentage change from the previous year, September 1999/2000)

Country	Label	GDP	LI	UR	IR	TB	NNS
Australia	A-lia	4.80	8.40	8.10	5.32	0.70	4.70
Canada	Can	3.20	2.50	8.40	5.02	1.60	5.20
Finland	Fin	3.90	-1.00	11.80	3.60	8.80	7.70
France	Fra	2.30	0.70	11.70	3.69	3.90	7.30
Spain	Spa	3.60	2.50	19.00	4.83	1.20	9.60
Sweden	Swe	4.10	1.10	8.90	4.20	7.00	4.00
United States	USA	4.10	1.40	4.50	5.59	-1.40	7.00
Netherlands	Net	2.90	1.60	4.20	3.69	7.00	15.80
Greece	Gre	3.20	0.60	10.30	11.70	-8.30	8.00
Mexico	Mex	2.30	5.60	3.20	20.99	0.00	12.70
Portugal	Por	2.80	-7.50	4.90	4.84	-8.70	14.00
Austria	A-tria	1.10	0.60	4.70	3.84	-0.60	9.40
Belgium	Bel	1.40	-0.10	9.60	3.64	4.50	12.40
Denmark	Den	1.00	1.50	5.30	4.08	3.30	5.00
Germany	Ger	0.80	-2.00	9.50	3.74	1.50	7.70
Italy	Ita	0.90	-0.40	12.30	6.08	4.30	8.20
Japan	Jap	0.10	5.40	4.20	0.74	1.20	15.10
Norway	Nor	1.40	0.90	3.30	4.47	7.10	15.10
Switzerland	Swi	1.10	2.10	3.80	1.84	4.40	13.20
United Kingdom	UK	1.20	4.90	6.40	7.70	-0.50	4.80

Although we are aware that many innovative visualization methods appear every year in the specialized literature, we will not discuss the capabilities of these methods here; instead we will focus our attention on those visualization methods in which the concept of distance plays a central role.

Factorial Analysis

4.3

In statistics, factorial analysis (FA) refers to a set of methods that permit one to reduce the dimension of a data matrix with respect to a least-squares criterion (Mizuta, 2004). The geometric formalization of the problem was one of the keys to the great success and quick dissemination of the methods.

Given a generic data matrix X of order $n \times p$, the aim of the methods is to replace p original variables with a set of q ordered and orthogonal factors that are obtained as a linear combination of the original variables, where $q < p$. Factors are ordered according to the information they carry. Orthogonality ensures consistent representations based on Cartesian space and allows us to split the whole variability into an additive linear model based on independent variables.

Of the factorial methods, principal component analysis (PCA) is probably the best, most used, and most implemented in statistical software packages. PCA deals with

quantitative multivariate observations. The method is conceptually founded on the singular value decomposition approach proposed by Eckart and Young (1936) for approximating a matrix with another one of lower rank.

Let us denote by \mathbf{X} the generic $n \times p$ data matrix, where the general term x_{ij} ($i = 1, \dots, n; j = 1, \dots, p$) indicates the value assumed by the i th statistical unit for the j th variable. From a geometric point of view, rows and columns of \mathbf{X} represent points in the \mathbb{R}^p and in the \mathbb{R}^n space, respectively. Without loss of generality, we assume that columns of data matrix \mathbf{X} have been standardized in matrix \mathbf{Y} having the same order of \mathbf{X} .

PCA allows one to find a q -dimensional subspace, where ($q \ll p$) holds position, such that the distances among the n row vectors were approximated in this subspace in a satisfactory manner. PCA problems allow for more than one formulation. The following section offers only a brief overview of the geometric formulation of PCA and mainly reflects the French approach to data analysis.

Readers interested in the PCA formulation problem are referred to, e.g., Hardle and Simar (2006); Jolliffe (2002); Lebart et al. (1984, 1995) and Mardia et al. (1979).

4.3.1 Principal Component Analysis

PCA was originally proposed by Hotelling (1933) as a method for determining the major axes of an ellipsoid derived from a multivariate normal distribution. Although a common interpretation of PCA as one specific type of factor analysis is widespread, data analysts have a different view of the method. They use PCA as a technique for describing a dataset without imposing any assumption about distribution or without starting from an underlying statistical model (Benzécri, 1976; Lebart et al., 1984, 1995). In this framework the point of view is then geometrically oriented and PCA aims to identify a subspace through the optimization of a given algebraic criterion. Among the potential criteria useful for fitting a set of n points to a subspace, the classical least squares is undoubtedly the most widespread method.

The problem, in a nutshell, consists in determining the unknown $p \times p$ matrix $\mathbf{U} = [\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_p]$ that indicates the maximum variance directions. The vector $\psi_j = \mathbf{Y}\mathbf{u}_j$ represents the coordinates of the n row points over the axis \mathbf{u}_j ($\forall j = 1, \dots, p$). The unknown matrix \mathbf{U} is determined solving the following eigenanalysis problem:

$$\mathbf{Y}'\mathbf{Y} = \mathbf{U}\Lambda\mathbf{U}' . \quad (4.1)$$

Notice that the previous equation can be alternatively expressed as

$$\mathbf{Y}'\mathbf{Y}\mathbf{U} = \mathbf{U}\Lambda , \quad (4.2)$$

where Λ is a square diagonal matrix of order p having as general element λ_j , and \mathbf{U} must satisfy the constraint $\mathbf{U}'\mathbf{U} = \mathbf{I}$. It is straightforward to say that λ_j and \mathbf{u}_j are respectively the generic eigenvalue and eigenvector of the square symmetric matrix $\mathbf{Y}'\mathbf{Y}$. Notice that $\{\lambda_1, \lambda_2, \dots, \lambda_p\}$ are ranked in decreasing order and \mathbf{U} defines an orthonormal basis.

Taking into account Eq. 4.2 and premultiplying the left and the right members of the equation by \mathbf{Y} , we obtain the following expression:

$$\mathbf{YY}'\mathbf{YU} = \mathbf{YU}\Lambda. \quad (4.3)$$

Let us denote $\mathbf{V} = \mathbf{YU}$; (4.3) then becomes

$$\mathbf{YY}'\mathbf{V} = \mathbf{V}\Lambda. \quad (4.4)$$

The above expression identifies in \mathbb{R}^n an orthogonal space in which to represent the p variables.

We call principal components the coordinates of n points in the orthogonal basis \mathbf{U}_q . They are defined as $\Psi = \mathbf{YU}_q$. Equivalently, variable coordinates are obtained by $\Phi = \mathbf{Y}'\mathbf{V}_q$. Matrices \mathbf{U}_q and \mathbf{V}_q are obtained by the first $q \leq p$ columns of \mathbf{U} and \mathbf{V}' , respectively.

FA output interpretation is a rather ticklish question. Very frequently we stumble upon analyses limiting interpretation to the graphical representation of points over the first or first two factorial plans. Even if this is the most evident aspect of the analysis, there is nothing worse than ignoring the rest of the output. Distances projected over factorial plans represent approximations of real distances. A correct interpretation, hence, should combine graphical and analytical results.

Unbiased output interpretation assumes knowledge of the following two basic principles: (i) Principal components are orthogonal by construction, so any orthogonal space defined by two or more axes represents an additive model. The additivity of the model allows us to determine the explained inertia associated to each factorial subspace as the sum of the respective eigenvalues. (ii) Original variables are centered, so the axes' origin corresponds to the average statistical unit. Then the distance from the origin reveals the deviation with respect to the mean vector.

These are very important and remarkable properties.

Taking into account these properties, let us consider the output of the OECD countries. Figures 4.1 and 4.2 show respectively the configuration of the variables and statistical units with respect to the first two factors of the OECD data table.

Standardization implies all variables are plotted inside a hypersphere having a *radius* equal to one; angles between variables and factors express the correlation. The first factor has a strength positive association with variables GDP and UR and negative with the NNS indicator. Indicators IR and TB have respectively direct and inverse relation with factor 2 and are almost uncorrelated with factor 1. We remark that the LI indicator does not fit the first factorial plan. Taking into account the variable positioning with respect to the factors, we interpret the characteristics of the statistical units according to their positioning. For example, the position of Spain in the lower right area implies that this country has a UR indicator value significantly greater than the variable mean. Looking at the data table, we notice that Spain has the greatest value (19.00).

Figure 4.3 shows the OECD countries with respect to the first three factorial axes. Notice the position of Portugal with respect to factor 3; it contributes to the orientation of the third factor and has a coordinate equal to -3.46.

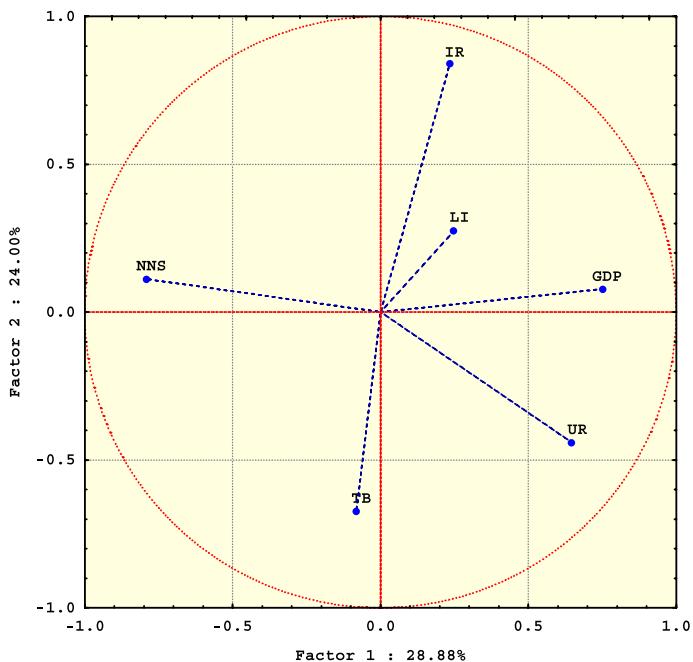


Figure 4.1. Correlation circle with respect to F1 and F2

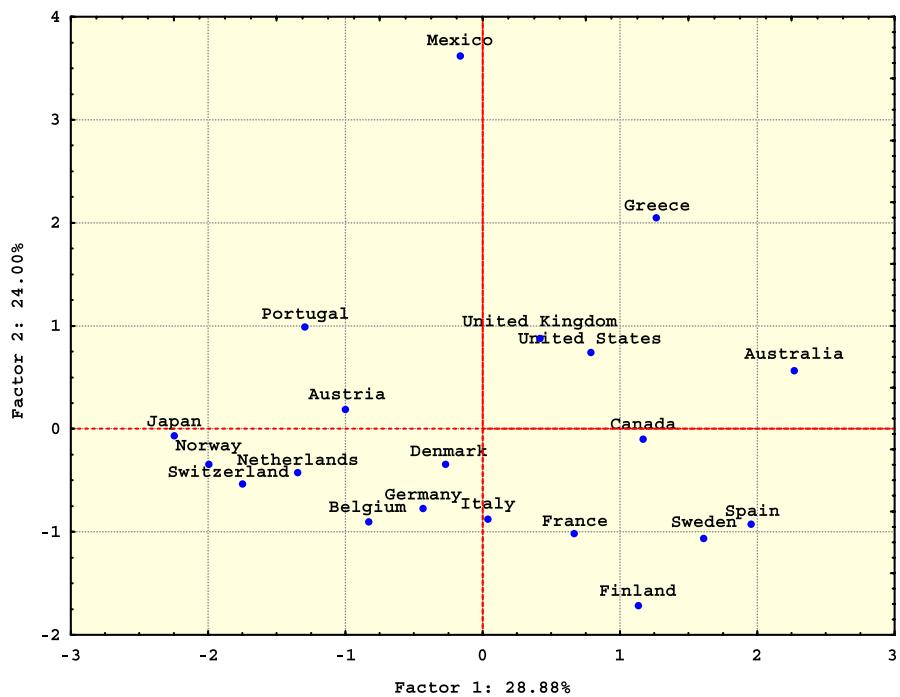


Figure 4.2. OECD countries with respect to the first two principal components

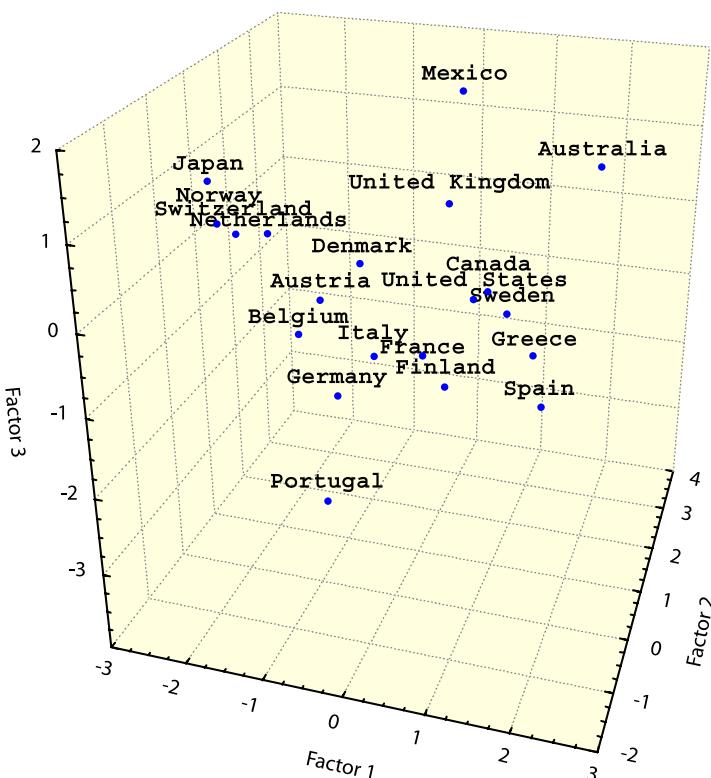


Figure 4.3. OECD countries with respect to the first three principal components

When the number of variables and statistical units remains moderate, statisticians very familiar with data analysis have a good opportunity to make correct interpretations taking into account these aspects. When the number of units and/or variables increases, it becomes hard and difficult to obtain a good interpretation. However, basic geometric principles still hold, and we move throughout the interpretation on the basis of the concepts “close” and “far”. Close points identify units with similar figures, while far points refer to units with clearly different characteristics.

Supplementary units and supplementary variables

There is a very important property pertaining to all FAs that is rarely commented upon in textbooks on multivariate analysis: the possibility of representing supplementary units and/or supplementary variables. Having the means to produce interactive graphical representations, this functionality allows one to add and remove subsets of statistical units and variables in graphical visualization in order to evaluate their position with respect to the others. In exploratory analysis, deletion of extreme points allows the analyst to evaluate the axes’ stability.

Let us consider the two following matrices: \mathbf{Y}^r and \mathbf{Y}^c . We assume that they are respectively of order $n^* \times p$ and $n \times p^*$. Rows of \mathbf{Y}^r are centered and reduced with respect to means and standard deviations of the matrix $\mathbf{X}_{n,p}$, and \mathbf{Y}^c contains standardized variables.

Row vectors in \mathbf{Y}^r can be projected as supplementary units and column vectors in \mathbf{Y}^c as supplementary variables according to the following *formulae*:

$$\Psi^r = \mathbf{Y}^r \mathbf{U}_q,$$

$$\Phi^c = \mathbf{Y}^{c'} \mathbf{V}_q,$$

where \mathbf{U} and \mathbf{V} are defined in (4.2) and (4.4), respectively.

Anybody with a little practice in FA knows that performing the analysis on the results of a huge dataset will *lose appeal*, whereas choosing a subset as a reference group and projecting other units as supplementary points ensures a much more interesting interpretation. The same holds if we turn our attention to the representation of variables. This procedure is advantageous to avoid heavy computational efforts on the whole dataset. Clearly we should be able to recognize in the graphical displays groups of active and supplementary units. Looking at the factorial plots in Fig. 4.2 and 4.3, we notice that there are two extreme points: Mexico and Portugal. Deleting these points and projecting them as supplementary, the plots in Fig. 4.4 illustrate the changes in the variable relationships. As a direct consequence of the deletion, the total inertia associated to the first factorial plan increases from 52.88 % to 59.96 %.

Effects depending on the deletion of Portugal and Mexico are clearly evident on the first principal plan (Fig. 4.4): Portugal moved through the axes' origin; the vertical axis presents a counterclockwise rotation, with no influence by Mexico.

Thus far we have focused our attention on the main features of FA, and we have provided the reader with guidelines on how to evaluate correct interpretations of a factorial plan. In the following sections we will show how to exploit FA's data analysis capabilities to produce useful exploratory analyses with the help of interactive tools.

4.4 Distance Visualization in \mathbb{R}^p

Previous sections showed how FA allows us to graphically evaluate the differences among statistical units in terms of distances. Obviously, we can represent 2-D or 3-D space, so that this task can be exploited taking into account no more than three factors simultaneously. The question is: how can we evaluate the differences in terms of the distance in \mathbb{R}^p spaces when $p > 3$? Even if we cannot graphically represent distances over spaces having more than three dimensions, we can compute distances in \mathbb{R}^p (with $p > 3$) using the Pythagorean theorem. The unresolved issue remains how to visualize these distances. Hierarchical clustering approaches furnish a good solution.

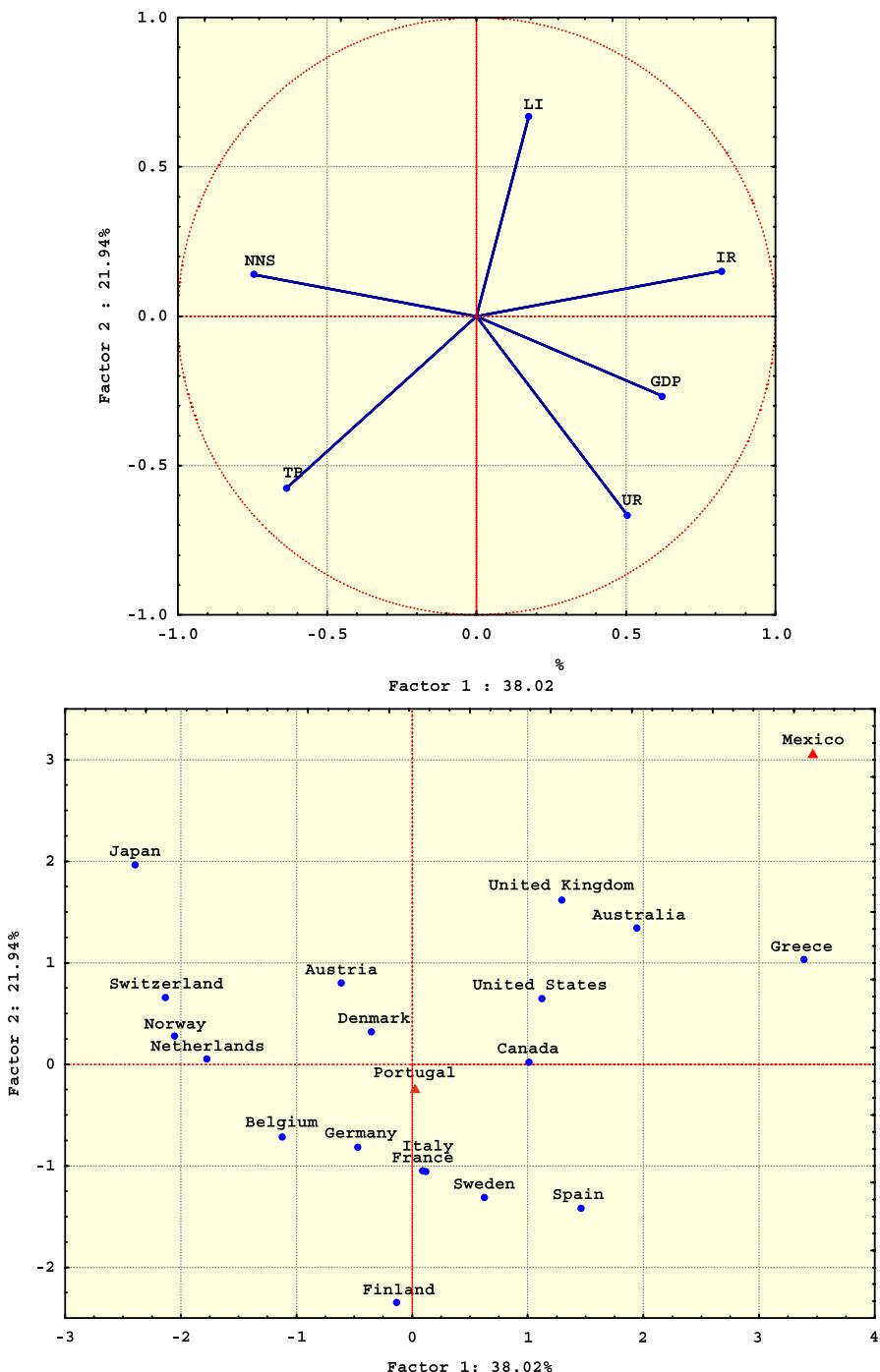


Figure 4.4. Variable correlation and OECD countries with respect to the first two principal components (Portugal and Mexico supplementary)

Given a set Ω of n statistical units described by p variables, the goal of cluster analysis is to partition data into k homogeneous groups or clusters (Gordon, 1999). Groups are defined to have high homogeneity within clusters and high heterogeneity among clusters. This is accomplished by grouping units that are similar according to some appropriate distance criterion. The choice of the distance criterion plays therefore a leading role in the group's definition.

Unlike factorial methods, clustering methods involve a data matrix reduction in the sense of the rows.

There are two main approaches in cluster analysis: nonhierarchical (partitioning) and hierarchical methods. Methods directly producing a partition of the nts objects into k clusters belong to the former approach. Many visualization techniques have been proposed to graphically represent partitions obtained using nonhierarchical algorithms. These displays permit one to understand the extent to which clusters differ; in fact, they are based on the use of color maps, flags, bubbles, networks, etc. However, these representations are completely loose with respect to metric space (Dhillon et al., 1998; Yang, 1999; Kohonen, 2000). Hierarchical methods define instead a unique sequence of nested partitions that can be visualized through dendograms. Dendograms represent the solution to our problem; they represent distances on flat surfaces.

Aside from the clustering method used, the effective use of classification requires interpretation rules to evaluate the heterogeneity of the obtained groups (in terms of observed variables).

Each group is described by continuous and nominal variables of the original dataset. Differences of conditional group means indicate the differences of groups and proper tests can be used to rank the variables according to their discriminant class power. Taking into account the nominal variables, useful measures to evaluate class heterogeneity are based on the difference between the proportion of each modality in classes with the respect to the proportion in the whole dataset.

After a description of hierarchical clustering methods, we focus our attention on partitioning methods.

4.4.1 Hierarchical Clustering

Given a set of n statistical units and a measure of distance or proximity, the main steps of a hierarchical classification algorithm are as follows:

Step 1: the $n \times n$ square symmetrical distance matrix $D^{(0)}$ is computed,

Step 2: the two points having the minimum distance are aggregated into a cluster; aggregated points are then treated as a new metapoint,

Step 3: the distance matrix is then updated to a $D^{(1)}$ matrix having order $(n - 1) \times (n - 1)$ by taking into account the new metapoint and disregarding the aggregated original units.

The previous steps are iterated until all cases are grouped into a single cluster of size n .

With respect to the distances to compute, the points to classify are reduced by one unit at each step. There are n singletons to classify at the first step and $n - i + 1$ groups and singletons at the i th step. Only one group of size n remains at the end of the algorithm.

The final output is mainly affected by two algorithm parameters: the distance measure and the linkage or amalgamation criterion to determine the distance between a point and a group or between two groups. There exist several distance measures according to the nature of the analyzed data.

Various linkage or amalgamation criteria permit one to determine the two most similar clusters (groups) to be merged into one new group:

SINGLE LINKAGE (nearest neighbor): the distance between two clusters is determined by the distance of the two closest objects in the different clusters.

COMPLETE LINKAGE (furthest neighbor): the distance between two clusters is determined by the distance of the two furthest objects in the different clusters.

AVERAGE LINKAGE: the distance between two clusters is defined as the average distance between all pairs of objects in the two different clusters.

CENTROID LINKAGE: the distance between two clusters is determined as the difference between centroids. The *centroid* of a cluster is the average point in the multidimensional space defined by the dimensions.

WARD'S METHOD: this method is based on the *minimum variance criterion* approach to evaluating the overall heterogeneity increase when collapsing two clusters. In short, the method aggregates clusters according to the minimum resulting sum of squares.

The end result of a hierarchical clustering algorithm is a sequence of nested and indexed partitions. The sequence can also be visualized through a tree, also called a dendrogram, which shows how the clusters are related to each other. The index refers to the aggregation criterion and indicates the distance between two subsequent groups (or objects). A dendrogram cut at a given level defines a partition of the data cases into different k groups, where k increases by one at a time as the aggregation index decreases. Figure 4.5 shows an example of a simple dendrogram and the resulting clusters at the shown cutting level.

Choosing the level of the cut, and thus the number of the resulting classes in the partition, can then be done by looking at the dendrogram: the cut has to be made above the low aggregations, which bring together the elements that are very close to one another, and under the high aggregations, which lump together all of the various groups in the population.

When it has been decided where to cut the dendrogram, the next step is to try to find out which variables have participated strongly to merge the cases in each cluster.

The dendrogram can therefore be used to provide visual grouping information, i.e., to read the process of merging the single statistical units into homogeneous clusters, thus playing a complementary role to the numerical algorithms in cluster analysis.

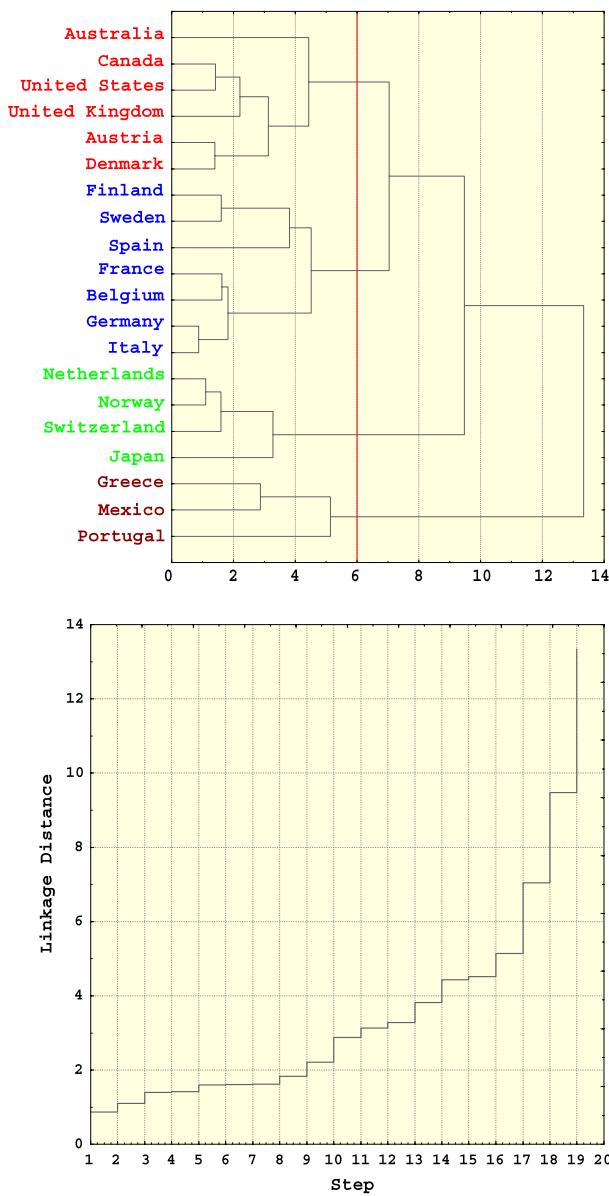


Figure 4.5. A dendrogram and the agglomerative index

Principal Axis Methods and Classification: a Unified View

4.5

The actual knowledge base in numerical analysis and powerful modern PCs allow us to successfully make use of the computational aspects in multidimensional data analysis (MDA). However, there are many analysis strategies that, without loss of efficiency, offer good solutions.

In the previous sections we have shown the centrality of the distance in factorial and clustering methods. This common element has been largely used to perform two-step analysis, namely, using both factorial and cluster analysis. Automatic classification techniques are used to group objects described by a set of variables; they do not make any claim to optimality. Nevertheless, they give relatively fast, economical, and easily interpretable results. PCA and other factorial methods rarely provide an exhaustive analysis of a set of data. Therefore, it is useful to perform a clustering of the observations because this helps to reduce the FA complexity. Additionally, it is of value to use classification analysis to summarize the configuration of points obtained from a principal axis analysis. In other words, a further reduction in the dimensionality of the data is valuable and leads to results that are easier to analyze. So-called “tandem analysis” represents a unified approach in which FA and clustering criteria, both based on the same notion of distance, are simultaneously satisfied in an iterative model (Vichi and Kiers, 2001).

All methods of multivariate descriptive statistical analysis are used in the same situation where the user is faced with a rectangular matrix. This matrix may be a contingency table, a binary matrix (with values of 0 or 1 according to whether an object has a certain attribute), or a matrix of numerical values. The use of automatic classification techniques implies some basic underlying concepts with respect to the purpose of the analysis. Either it is assumed that certain groups must exist among the observations or, on the contrary, the analysis requires a grouping of the observations. In other words, a 2-D continuous visualization of the statistical relationships is not enough. There is also an interest in uncovering groups of individuals or of characteristics.

A given set of results might be reached through different steps and might lead to different interpretations. For example, the problem may be to discover a partition that really exists and that was hypothesized before carrying out the statistical analysis. Conversely, it may be useful to employ partitions as tools or as surrogates in the computations that make it easier to explore the data. In any case, using principal axis methods in conjunction with classification makes it possible to identify groups and to determine their relative positions.

Often partitions or tree structures are used to amplify the results of preliminary principal axis analysis during the exploratory phases of data analysis. There are several families of classification algorithms: agglomerative algorithms, in which the clusters are built by successive pairwise agglomeration of objects and which provide a hierarchy of partitions of the objects; divisive algorithms, which proceed by successive dichotomizations of entire sets of objects and which also provide a hierarchy of parti-

tions; and, finally, algorithms leading to partitions, such as the methods of clustering about moving centers or other minimum variance algorithms.

4.6

Computational Issues

Computational aspects in statistical data mining are treated comprehensively by Wegman and Solka (2005); we refer to their description of computational complexity to better understand the impact of large and massive datasets in the MDA approaches.

Most critical issues become apparent when applying cluster analysis methods. As a matter of fact, the necessary computational effort to attain results in FA depends on the number of variables. Under the common situation in which the number of statistical units is much larger than the number of variables, the computation of a solution can be carried out on the matrix of order $p \times p$, where p indicates the number of columns. Looking at (4.2), it is straightforward to notice that the problem in \mathbb{R}^p has a very feasible computational complexity, of the order $O(p^2)$. With very low computational effort, transition *formulae* (Lebart et al., 1984) permit one to compute the results in \mathbb{R}^n .

Hierarchical clustering algorithms, conversely, are very time consuming, as the computational effort for such algorithms is of the order $O(m^2)$, where m denotes the number of entries in the data matrix. According to Wegman and Solka (2005), using a Pentium IV 1-GHz machine with 1-gigaflop performance assumed, the time required for clustering a dataset with a *medium* number of entries (10^6 bytes) is about 17 min, while about 116 d are required to handle a *large* number of entries (10^8 bytes). When the dataset size rises to 10^{10} bytes (*huge*) it takes 3170 years!

Nonhierarchical clustering algorithms offer good performance with decent computation time even with huge datasets. In the following subsections, we briefly introduce partitioning methods and describe a mixed two-step strategy (nonhierarchical + hierarchical) largely used in a MDA framework.

In Sect. 4.7 we will show how advanced graphical representations can add useful information to a factorial plan and how the human–machine interaction helps us to navigate throughout the data in search of interesting patterns.

4.6.1

Partitioning Methods

Nonhierarchical clustering attempts to directly decompose a dataset into a set of disjoint clusters of similar data items. The partition is obtained through the minimization of a chosen measure of dissimilarity. In particular, taking into account the variance decomposition, the method aims to minimize the ratio

$$Q = \frac{\text{trace}(W)}{\text{trace}(T)}, \quad (4.5)$$

where $\text{trace}(W)$ and $\text{trace}(T)$ denote the *within* groups and *total* variance–covariance matrices, respectively. According to the variance decomposition formula,

minimizing the quantity in (4.5) is equivalent to maximizing the quantity $\text{trace}(\mathbf{B})/\text{trace}(\mathbf{T})$. Notice that $\text{trace}()$ indicates the trace operator, $\text{trace}(\mathbf{T}) = \text{trace}(\mathbf{W}) + \text{trace}(\mathbf{B})$ (Mardia et al., 1979).

The algorithm family known as *k-means* is the most widely known nonhierarchical clustering approach; among the different *k*-means algorithms, that proposed by Forgy is the most widely implemented in specialized software packages (Hartigan, 1975).

Let us assume partitioning of a set of n units characterized by p variables; the main steps of a *k*-means algorithm are the following:

- STEP 1: k provisional group centers are randomly determined: $\mathbf{c}_1^0, \mathbf{c}_2^0, \dots, \mathbf{c}_k^0$.
- STEP 2: a partition $P^0 = \{C_1^0, C_2^0, \dots, C_k^0\}$ of the n objects into k clusters is obtained using the assignment rule: a unit belongs to C_k^0 if it is nearer to \mathbf{c}_k^0 than to all other centers; an object x_i is then assigned to the cluster C_k^0 if $d(x_i, \mathbf{c}_k^0) = \min$.
- STEP 3: k new cluster centers are determined: $\mathbf{c}_1^1, \mathbf{c}_2^1, \dots, \mathbf{c}_k^1$ as the centers of gravity of the clusters of the partition P^0 . New centers are then used to define a new partition $P^1 = \{C_1^1, C_2^1, \dots, C_k^1\}$ constructed according to the same rules used for P^0 .

The previous steps are iterated until convergence to a final partition, i.e., when two succeeding iterations lead to the same partition or when a chosen criterion is obtained that describes the quality of the obtained partition. A further stopping criterion can be based on the number of iterations.

Stable Groups Identification

Although it is based on a relatively slight theoretical basis, the *k*-means classification method has an efficacy largely attested to by empirical results (Milligan, 1996). As a consequence, it is the partitioning method *best adapted to large datasets*. The *k*-means method is used as an adjunct to other methods such as clustering prior to principal axis analysis or directly as a descriptive tool. The method is particularly well adapted to large numerical datasets because *the data are read directly*: the data matrix is saved on auxiliary memory and is read several times in sequential fashion, without requiring large amounts of computer memory.

As the within-group variance can only become smaller between two steps of iterations, the algorithm does converge; however, it is not the convergence itself but *its very high rate of convergence* that justifies using this method in practice.

Generally, obtained partitions depend on the centers chosen at the first iteration. The algorithm could converge toward local optima. The procedure of finding stable groups (Lebart et al., 1984) is a kind of remedy for this situation. Its main advantage is that it elaborates the results obtained in the rigid framework of a single partition by highlighting high-density regions of the object points. The technique consists in performing several partitions starting with different sets of centers and keeping as stable groups the sets of objects that are always assigned to the same cluster.

Let us consider, as a small illustrative example, partitioning 1000 individuals into several homogeneous groups. We perform a first basic partitioning into 6 groups around moving centers (only a few iterations are necessary to ensure stability of the groups). This procedure is repeated three times.

Table 4.2. Partition of 1000 units into 6 homogeneous groups

3 partitions into 6 groups	1	2	3	4	5	6
<i>Partition 1</i>	127	188	229	245	151	60
<i>Partition 2</i>	232	182	213	149	114	110
<i>Partition 3</i>	44	198	325	99	130	204

Table 4.3. Stable groups: distribution of units

Stable groups: cardinalities in decreasing order										
1–10	168	118	114	107	88	83	78	26	22	16
11–20	15	14	12	12	12	11	10	7	7	7
21–30	6	6	4	4	4	4	3	3	3	3
31–40	3	3	3	2	2	2	2	2	2	2
41–50	1	1	1	1	1	1	1	1	1	1
									<i>Total</i>	1000

Table 4.2 shows the cardinality of the 6 groups of the 3 successive partitionings. In fact, this is only the first step to follow in order to pursue stable groups. The three partitionings are then cross-tabulated, resulting in a subdivision of the 1000 objects into $6^3 = 216$ cells. The individuals in each of these 216 cells are those who have always been grouped together in the three partitionings. They constitute the potential *stable groups*. In fact, only 50 groups are not empty, and 40 out of these 50 groups contain less than 15 individuals. The distribution of the individuals is given in Table 4.3. In practice the number of stable groups with substantial cardinality is always much smaller than the number of cells resulting from crossing the basic partitions (in the example, only 7 cells among 216 have more than 70 individuals, compared to 43 cells that have less than 30 individuals, while all the others are empty). In its first phase, the method presented below uses a partitioning technique that is designed for large data tables. The groups obtained from this phase are then clustered through an agglomerative algorithm. This method combines the advantages of both approaches, namely:

1. The ability to treat very large matrices;
2. A detailed description of the main clusters;
3. The ability to make a critical choice with respect to the number of clusters.

4.6.2 Mixed Strategy for Very Large Datasets

When a large number of objects are to be classified, the following classification strategy will be used. The idea is to combine the two approaches presented above: finding a partition and then building a classification tree. The first step is to obtain, at a low cost, a partition of the n objects into k homogeneous groups, where k is far greater than the “expected” number of groups in the population (say $k = 500$ when $n = 500,000$). The second step is an ascending hierarchical classification, where the

terminal elements of the tree are the k groups of the preliminary partition. The next step is to cut the tree at the most appropriate level to obtain an interpretable partition. This level may be chosen visually or automatically determined (some “Cattell criteria”) (Cattell, 1978). The final partition will be a refinement of this crude partition. A schema of the mixed strategy follows (Fig. 4.6).

STEP 1: preliminary partition

The first step is to obtain rapidly a large number of small groups that are very homogeneous. We use the partition defined by the stable groups obtained from cross-tabulating two or three base partitions. Each base partition is calculated using the algorithm of moving centers (k -means) after reading the data directly so as to minimize the use of central memory. The calculations are generally performed on the coordinates of the individuals of the first few principal axes of a principal coordinate analysis. Note that the distance computations are accelerated on these orthogonal coordinates, as noise in the data (distributed within the last coordinates) is eliminated and as principal coordinates may be efficiently computed using any stochastic approximation algorithms.

STEP 2: hierarchical aggregation of the stable groups

Some of the stable groups can be very close to one another, corresponding to a group that is artificially cut by the preceding step. On the other hand, the procedure generally creates several small groups, sometimes containing only one element. The goal of the hierarchical aggregation phase is to reconstitute the groups that have been fragmented and to aggregate the apparently dispersed elements around their original centers.

The tree is built according to Ward's aggregation criterion, which has the advantage of accounting for the size of the elements to classify as weight in the calculations of the loss of variance through aggregation. It is a technique of minimum variance clustering that seeks to optimize, at every step, the partition obtained

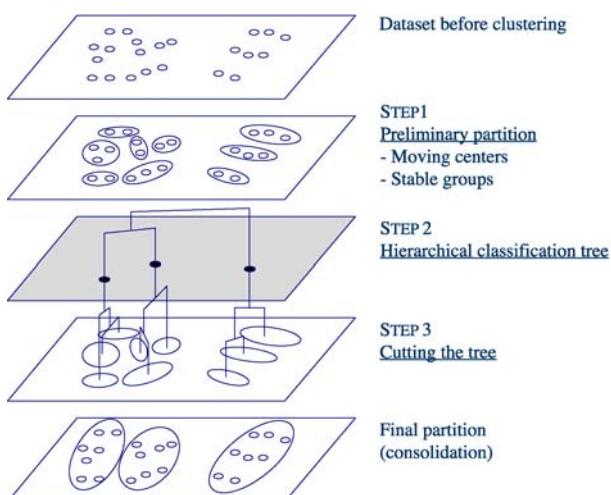


Figure 4.6. Mixed strategy for classifying huge datasets

by aggregating two elements, using criteria linked to variance. The computations are not time consuming when the clustering is performed after a factorial analysis (PCA or MCA) and the objects to be classified are located by their coordinates on the first axes of the analysis.

STEP 3: final partition

The partition of the population is defined by cutting the dendrogram. Choosing the level of the cut, and thus the number of classes in the partition, can be done by looking at the tree: the cut has to be made above the low aggregations, which bring together the elements that are very close to one another, and under the high aggregations, which lump together all the various groups in the population.

Some Considerations on the MIXED strategy

Classifying a large dataset is a complex task, and it is difficult to find an algorithm that alone will lead to an optimal result. The proposed strategy, which is not entirely automatic and which requires several control parameters, allows us to retain control over the classification process. The procedure below illustrates an exploratory strategy allowing the definition of satisfactory partition(s) of data. It is weakly affected by the number of units and can offer good results in a fairly reasonable time. In MDA applications on real datasets, especially in cases of huge databases, much experience is required to effectively tune the procedure parameters (Confais and Nakache, 2004).

A good compromise between accuracy of results and computational time can be achieved by using the following parameters:

1. The number of basic partitionings, which through cross-tabulation define the stable groups (usually two or three basic partitionings);
2. The number of groups in each basic partitioning (approximately equal to the unknown number of “real” groups, usually between 5 and 15);
3. The number of iterations to accomplish each basic partitioning (less than five is usually sufficient);
4. The number of principal coordinates used to compute any distance and aggregation criterion (depending on the decrease of the eigenvalues of principal axis analysis: usually between 10 and 50 for a large number of variables);
5. Finally, the cut level of the hierarchical tree in order to determine the number of final groups (in general, by visual inspection).

Nearest-neighbor-accelerated algorithms for hierarchical classification permit one to directly build a tree on the entire population. However, these algorithms cannot read the data matrix sequentially. The data, which usually are the first principal coordinates of a preliminary analysis, must be stored in central memory. This is not a problem when the tree is built on the stable groups of a preliminary k -means partition (also computed on the first principal axes). Besides working with direct reading, the partitioning algorithm has another advantage. The criterion of homogeneity of the groups is better satisfied in finding an optimal partition rather than in the more constrained case of finding an optimal family of nested partitions (hierarchical tree). In addition, building stable groups constitutes a sort of self-validation of the classification procedure.

Factorial Plans and Dendrograms: the Challenge for Visualization

4.7

The synergy between computer graphics and visual perception permits one to design statistical tools for the interactive visual exploration of statistical data (Unwin et al., 2006). Data visualization is becoming an increasingly important tool in scientific research. Nowadays, the availability of high-performance computing has definitively changed the role of statistical graphics: they are not only a static view of the past but also a dynamic partner and a guide to the future (Wainer and Velleman, 2001). Motivated readers are encouraged to visit the Web site managed by Friendly and Denis (Visited on Oct. 2005), which presents a historical overview of advances in statistical graphics and most of the widely used data representations (Friendly, 2000).

Based on psychometric experiments and anatomical considerations, Wegman has pointed out the human eye's capabilities in visually processing data. More specifically, Wegman's (2003) "recipe" is based on three ingredients: a geometric support, a projecting function of original data into a suitable graphical space, and interaction tools. Wegman's analysis strategy is based on a system of parallel coordinates as visualization support (Inselberg, 1999), a high-dimensional rotation algorithm (Asimov, 1985; Cook et al., 1995; Buja et al., 2005), and *saturation brushing* and *color design* (Wegman, 2003).

Linking original data with graphical representation allows the user to have an innovative view of the data: by querying objects on the graph, the user directly interacts with the data and with the analysis parameters.

According to Wegman's recipe, but using different ingredients, we propose another way to handle data visualization in a statistical context; we like to consider our approach as a little more statistical and a little less computational.

Parallel-coordinate systems are replaced with Cartesian systems (where axes take on a very special statistical meaning because they are factors); the similarity between statistical units in \mathbb{R}^P is evaluated in terms of distances by the use of dendrograms; classical brushing and other more or less classical interactive tools that are presented below ensure a smooth man-machine interaction.

There are many software programs and packages that allow the user to interact with data: users can select groups of units and change the appearance of the plot. With respect to classical interactions, the approach proposed by Wegman allows the user to affect visualization support by changing the analysis parameters.

In other words, the data display changes, either when there are changes in the data (adding or deleting variables or units) or when the analysis parameters are modified. In both cases the visualization satisfies a (statistical) criterion.

Interaction capabilities in the visual data exploration phase complete the interactivity analysis toolkit. Results are presented in some visual forms providing insight into the data, drawing conclusions, and interacting with the data.

A useful task-oriented criterion for pursuing an exploratory approach is the *visual information-seeking mantra* (Shneiderman, 1996; Card et al., 1999). The basic steps for this kind of approach are listed in order of execution below:

- Overview
- Zoom and filter
- Details-on-demand

First, the user needs to get an overview of the data. In the overview phase, the user identifies interesting patterns and subsets and focuses on one or more of them. Focusing consists in a distortion of the overview visualization or in using another visualization technique (Buja et al., 1991). Analyzing the patterns, the user highlights the most important information, using the drill-down functionality. Finally, accessing the (selected) data subsets, the user can retrieve interesting patterns and subsets in order to perform further detailed exploration (Antoch, 2005).

According to Shneiderman's (1996) proposal for designing advanced graphical user interfaces, four more tasks supplement the previous list, namely:

OVERVIEW: the user gains an overview of the data.

ZOOM: the user zooms in on items of interest.

FILTER: the user filters out uninteresting items, by dynamic queries. By allowing users to control the contents of the display, they can quickly concentrate on interesting patterns. At this aim, the software should offer both controls and rapid display update.

DETAILS-ON-DEMAND: the user selects an item or a group of items in order to get details.

RELATE: the user views relationships among data items.

HISTORY: undo and replay support allows the user to progressively refine the exploration; the run actionSs history is stored to retrace the followed steps.

EXTRACTION: allows the user to extract subsets through query parameter setting.

The analysis approach in an exploratory framework mainly focuses on the results of direct visualization. These results can be made dynamic offering user interaction capabilities on the graphical representation. The incorporation of interactive techniques implies that *to any action on a screen corresponds a reaction either in numerical analysis or in visualization* (VITAMIN-S FSR, 2002).

4.7.1 **The Use of Visual Variables for Designing Useful Representations**

In an interview, Bertin (1967) stressed the idea that the use of computers for visualization should not ignore the real objectives of graphics, namely, treating data to get information and communicating, when necessary, the information obtained.

According to the definition of an image as *a sign's structured set* in a visual field, Bertin identifies three dimensions: X and Y define the sign's position in 2-D space, and Z, the third dimension, specifies the informative content. In order to measure the usefulness of any given construction or graphical invention and to avoid useless graphics, Bertin warns us to answer the following three questions:

- Which are the X, Y, and Z components of the data table? (What is it all about?)
- What are the groups in X and Y that Z builds? (What is the information at the general level?)

— What are the exceptions?

A useful representation must provide clear answers to these questions.

In addition, Csinger (1992) refers to the Steven and Weber laws on human perceptive capabilities and highlights two basic behaviors of human perception: (*i*) differences are perceived in terms of relative variations; (*ii*) human perception of variations is biased and the bias is against valuing distances and gradually becomes more favorable toward areas, volumes, and colors.

Traditional (sometimes overused) factorial maps and dendrograms, thanks to their full correspondence to Bertin's principles, are very popular. Moreover, giving the maximum prominence to distances, these representations are very helpful in minimizing biased perceptions.

In addition, to enrich information that can be transmitted through 2-D representations, Bertin introduces seven visual variables: *position, form, orientation, color, texture, value, and size*.

The display in Fig. 4.7 was realized from a real dataset and was chosen because it is a good example to understand how to read this kind of enriched visualization. At this point it is not relevant to know which data are represented: in Sect. 4.8 the actual informative potentiality of enriched representations will be shown in practice. Looking at Fig. 4.7, the reader can appreciate how the inertia (of the first two PCs) is distributed among the statistical units and according to the factorial axes. In particular, units are visualized by means of *pies*: sizes are calculated according to the total contribution of each statistical unit; the slice color denotes the share of contribution to each factor (accordingly colored). Influential units are characterized by the biggest pies and can be easily and quickly identified even if there are thousands of points on the plot.

It is possible to obtain a similar representation using either the absolute contributions or the square cosinus associated with the points. They consist of coefficients computable for each axis allowing one to interpret the axes in terms of the units and the adequacy of the unit representations, respectively (Lebart et al., 1984). In particular:

1. ABSOLUTE CONTRIBUTIONS, which indicate the proportion of explained variance by each variable with respect to each principal axis.
2. SQUARED CORRELATIONS, which indicate the part of the variance of a variable explained by a principal axis.

It is also possible to represent both the above measures using an index for drawing the pies and the other to attribute different brightnesses to the points (a darker point could denote a high value of the index associated to it).

In the case of cluster representation, it is possible to query a given cluster on the factorial plane in order to visualize its internal composition (in terms of both units and variables) through the use of “drill-down” functionalities.

Obviously, clustering methods can be either nonhierarchical or hierarchical. It is clear that in the case of a hierarchical method that exploits linking functionalities, it is also possible to link two different views of the same data (the dendrogram and the factorial plane) in order to obtain more information (Sect. 4.5).

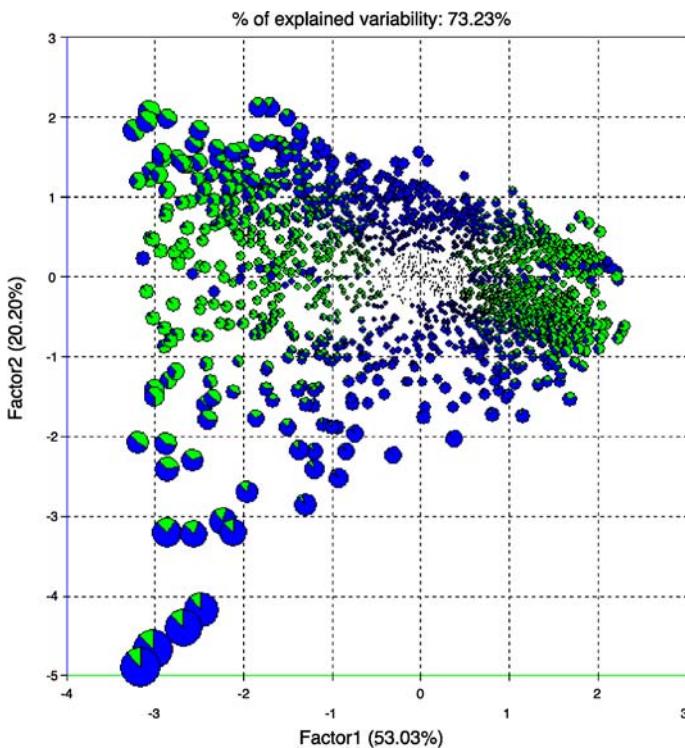


Figure 4.7. PCA unit display: absolute visualization of contributions through point size and color (green and blue refer, respectively, to the first and second factors)

A useful interactive capability in this sense is based on the dendrogram dynamic cutting: changing the cut level of the dendrogram, the user can consequently visualize the resulting partition on the factorial plane. Obtained clusters can be described in terms of descriptive tables and charts; the user can decide to drop a cluster from the analysis or to aggregate two or more homogeneous clusters. As a consequence of the user action, the analysis restarts again and the results change.

Aside from the classical dendrogram, alternative views can be obtained using the tree views that are typical of modern operating system user interfaces. An illustrative example is shown in Fig. 4.8. It is an innovative representation useful for visualizing results of a cluster analysis that also allows the user to query a particular level of the tree.

The user can collapse (expand) a particular cluster in order to hide (to show) the items included in that cluster. Drill-down functionalities allow one to browse a given cluster, while linking functionalities between the tree-view representation and the factorial plane provide the user the capability to select a different cut level in order to visualize the corresponding clusters on the factorial plane.

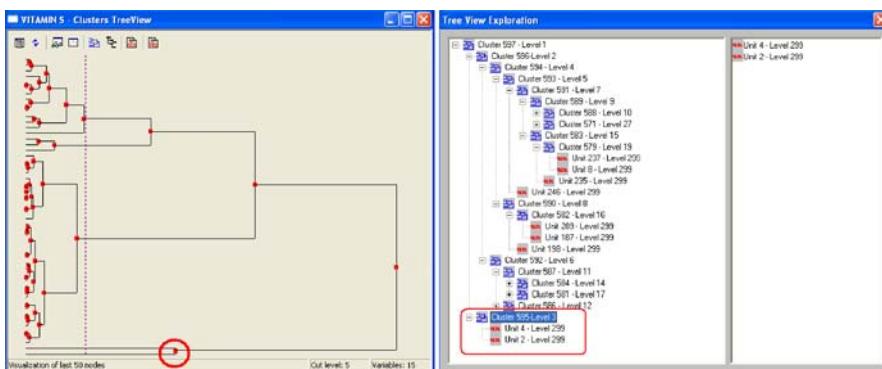


Figure 4.8. A tree view clustering representation

Interactivity Within and Between Graphics

4.7.2

Global views are used to provide context for more detailed views, to help formulate a search, identify patterns, or make a gestalt overview (Cleveland, 1993, 1994; Bounfond, 2000).

Human-machine interaction confers substantial advantages to the power of statistical graphics (Cleveland and McGill, 1988). Interactive manipulation is the capability to directly interact with displays to query them for information, to link them, and to change their shapes, scales, and other characteristics so that a wide range of views can be scanned. Interactive manipulation techniques can be developed and used within each graphic as well as between graphics and data.

Manipulation techniques can be classified into two main areas:

1. INTERACTIVITY WITHIN GRAPHICS, aimed at promoting interactivity on a single graphical representation. Among the available tools are: the ability to show/hide parts of a graph, the ability to use color/size information in order to change the representation according to the point attributes, and the ability to use the mouse pointer to query the graphical representation. Moreover, the user can alter the data table in order to observe changes on the graph. Such changes include the deletion of data, the highlighting of data (the data will stand out against normal data), and the focus of data (in order to show as much detail as possible on the data). Nevertheless, traditional tools can be used, i.e., context-sensitive querying, zooming, flexible scaling, resizing of graphics and of objects, coloring of objects, selecting (by points, areas, etc.), masking, and grouping (Dhillon et al., 1998; Hurley and Buja, 1990).
2. INTERACTIVITY BETWEEN GRAPHICS, aimed at promoting the interactivity among different views of the same data. It is straightforward to say that the previously listed tools can be used also for this kind of interactivity, mainly based on the linking concept: to any action on a view corresponds the same reaction on all the related views of the same data (Symanzik, 2004).

Previously we pointed out that the use of graphical representations could be a starting point for data exploration. In other words, starting from these representations, the main idea of graphical interactivity is to allow the user to visually query the data (Unwin, 1999). That is, once the data have been displayed on graphics, the next step is to allow the user to manipulate the graphics in an interactive fashion, in order to search for patterns in the data. We refer the interested reader to Wilhem's survey on "the paradigm of the linked view", which recently appeared in Rao et al. (2005) (Wilhelm, 2005).

According to their basic aims, a consistent taxonomy of graphical interactive tools is as follows:

FINDING PATTERNS: clusters, outliers, unusual groups, and local densities are examples of features that should be interactively inspected by users. Very important is the inspection of low-dimensional dependencies, which are helpful for dimension reduction.

POSING QUERIES: after the identification of interesting patterns, the user should be able to highlight individual cases as well as subsets of data. The results of such queries should be given in a graphic display.

Different interactive methods are used to carry out the previous tasks. The underlying idea that associates the different usable tools is the data consistency principle: whatever changes the user causes, each available data view (numerical and graphical) is consequently updated.

The analyst can thus act on the screen, generating parameter changes and causing a new execution of the analysis and a subsequent redrawing of the graphical representations. The user can take several different analysis pathways based on the use of one or more interactive tools that allow the user to address different tasks:

- Exploring the original dataset (drill-down functionalities);
- Asking for a different representation of all the information (multiple views) or of the information stored in a selected area of the graph (subset views);
- Dropping points, units, and/or variables from the analysis (deletion): deleting point(s) can help the user to evaluate the sensitivity of the analysis and of subsequent representation to particular units and in order to compare this sensitivity among the different views;
- Aggregating/disaggregating (grouping/ungrouping) units in order to reduce/increase the level of detail of the representation;
- Filtering the representation in order to visualize only part of the information. The filtering criteria can be based both on values of one or more variables (using suitable tools to build and/or query) and on indexes (the user can ask to graph units characterized by given values of one or more indexes). With respect to the last point, in particular, a set of statistical measures can be used to filter the relevant information so as to reduce the amount of represented points according to the importance of the information lying below;
- Focusing on specific characteristics: the user can decide which feature of the data to visualize and the way to display it. This includes the choice of variables, the scale and aspect of the plots, the possibility to animate the plots using real-time controls

in order to have different views of the same set of data, the choice of particular range of values for a numerical variable, the use of tools to obtain a better visual perception of the data (in the case of multiple points in the same plotting location one simple solution is jittering, i.e., a small amount of uniform random noise is added to the data before graphing);

- Asking for a representation of part of the information, namely, starting from unit attributes (subset selection) or querying a given region (selection of an area), that is, the user can concentrate on particular points and regions. These are made to stand out from the background and highlighted using a different color, dimension, or filling in order to look for clusters or behavior different with respect to unselected points;
- Analyzing several linked views of the same data (linking, brushing, slicing). Multiple graphics is an old technique used to propagate information through different plots that display different aspects (dimensions) of the same dataset. When points are selected on a view through an input device, all views of that case are highlighted on each of the other plots simultaneously. In this way the user can analyze if the selected points have some particular features in the other dimensions or views. Linking, brushing, and slicing are based on the idea that by emphasizing the common identity of cases in multiple displays, the analyst can more easily relate several displays to one another. They allow one to look for clusters, to explore the relationships among variables, or to investigate if a particular pattern or position is confirmed on different views of the same data. Moreover, the linking capability can be used not only among different graphical displays but also between a graph and the input data;
- Redrawing the views using only certain selected levels of a strata variable (conditional selection), mainly categorical variables, even if intervals of a quantitative variable can be used. The redrawing of the graphs requires one to rerun the statistical analysis in order to visually concentrate on groups of units characterized by the same level(s) of the strata variable. This tool provides a powerful method to execute conditional analysis.

An Application: the Survey of Italian Household Income and Wealth

4.8

Previous ideas are best transmitted through a typical application. Results shown in the present section have been obtained using the software VITAMIN-S. VITAMIN-S (VI_sual daTA MINing System) is a prototype software realized in the framework of the 1999–2002 IST-2000-26341 project; the project was sponsored by EC fifth FP.

This contribution has been greatly affected by the authors' experience in the development of the VITAMIN-S software. The most innovative aspect of VITAMIN-S

is the connection between MDA techniques and graphic interaction tools (Klinke, 2004). The software was designed as a data-mining tool to work with huge amounts of data.

Data presented in this section come from the Italian Survey of Household Income and Wealth (SHIW), a large-scale household survey run biannually by the Bank of Italy on a panel of about 8000 Italian households (24 000 individuals) spanning about 300 Italian municipalities (Bank of Italy, 2004). The SHIW began in the 1960s with the aim of gathering data on the income and savings of Italian households, and it is widely used in studies on saving behavior by Italian households. It collects detailed information on Italian household demographic characteristics, consumption, income, and balance sheet items. For our analysis, we consider data of the last survey (year 2004). For an exhaustive description of the data and related issues, see D'Alessio et al. (2004).

For the sake of simplicity, among the available indicators we have selected are the following:

ETA: age (years)

Q: working status (1 = employed, 2 = self-employed, 3 = unemployed)

AREA5: geographical area

(1 = northeast, 2 = northwest, 3 = central, 4 = south, 5 = islands)

YL: compensation of employees

YT: pensions and net transfers

YM: net income from self-employment

YC: property income

CD: consumption (durables)

CN: consumption (nondurables)

S: Saving

AR: real assets

AF: financial assets

PF: financial liabilities

BD: consumer durables

Data exploration and feature extraction can be well presented through the visual approach. Unfortunately, static views presented in this section do not allow us to appreciate the human-machine interaction analysis power that comes from MDA.

The following results refer to a two-step analysis (factorial analysis and clustering). Looking at the point configuration on the factorial plans, the user can modify the analysis parameters or add and remove variables or statistical units.

VITAMIN-S software was designed to be used also by users not having a high level of skill in MDA. Indeed, specifying the type of variables and the desired analysis strategy is enough to start.

In this example we classed all variables as continuous except Q and AREA5. In particular we used the AGE variable and the two categorical variables as supplementary. Then we specified factorial analysis plus clustering as a strategy. Once these parameters were established, VITAMIN-S performs the whole analysis and then the user can act on the graphic displays. Any action performed on one display causes methods to

run again (if necessary), and new results updating the current views automatically appear.

Let us start the analysis results interpretation by looking at the representation of variables with respect to the first two factors. In Fig. 4.9 we see the first factorial plan whose total associated inertia is 47.42 %. We observe all indicators, but YT contributes to the first factor. With respect to the second factor, instead, they split into two groups.

After the factorial axes are interpreted according to the variables in the analysis, we shift our attention to the statistical units' configuration on the same factorial space. In Fig. 4.10 we observe the configuration of points on the first factorial plan. As in the most of data-mining applications, we notice that the largest part of the statistical units is massed on the axis origins. This happens because the analysis is strongly affected by a few influence points.

In order to detect which kind of units have the greatest influence, the system allows the user to differentiate points according to the modalities of a categorical variable. In this example, on the right-hand side of Fig. 4.10 points are colored according to the modalities of variable Q (*working status*): blue points correspond to “employed”, cyan indicates “self-employed”, and black indicates “unemployed”.

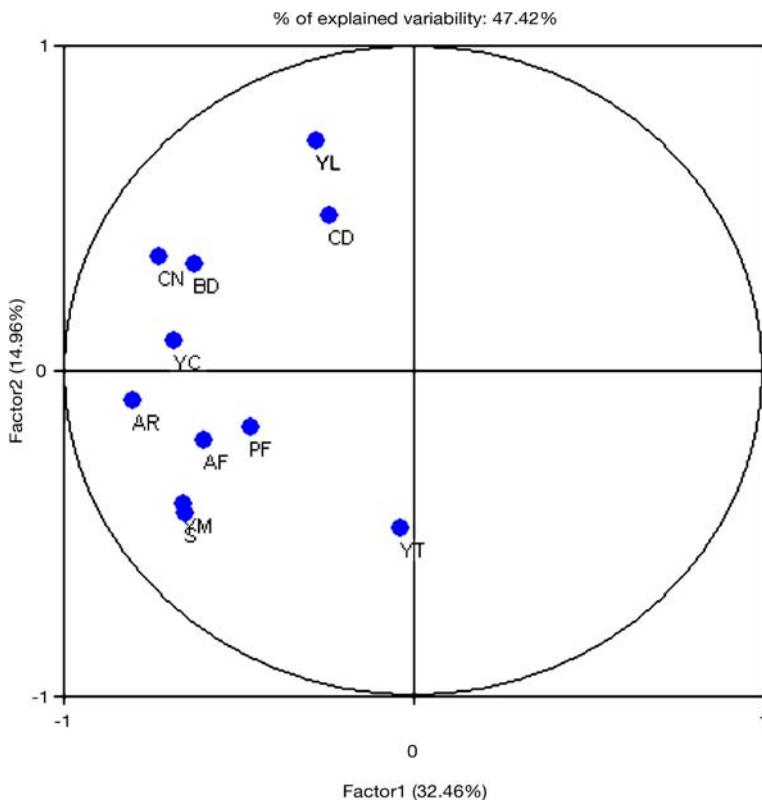


Figure 4.9. Representation of variables: SHIW data

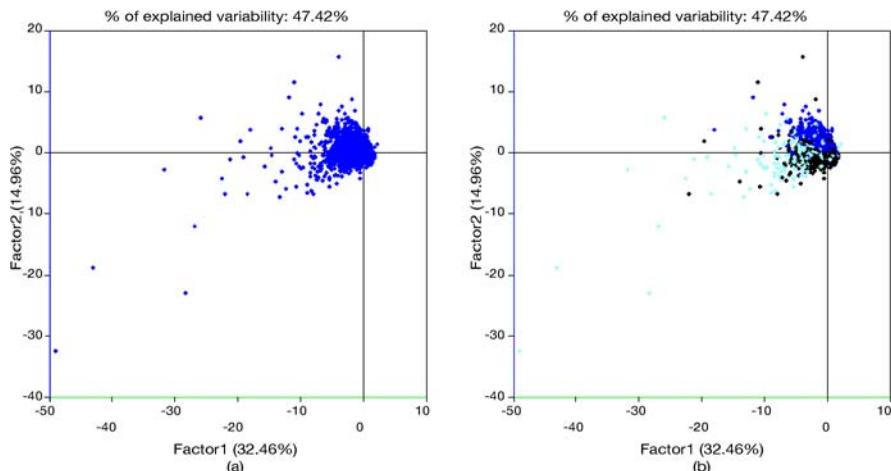


Figure 4.10. Statistical units on the first factorial plan (SHIW data). (a) Classical view. (b) Units distinguished according to categories of Q variable

Some interesting patterns can also be revealed using a scatterplot matrix representation obtained by representing the first factor with respect to the other ones, as shown in Fig. 4.11. The plot, as above, uses highlighting to explore the position of employees (black points) in the different spaces.

Figure 4.12 shows the contribution representation for the units. With respect to the classical representation that uses *dimensionless* points, the same units are represented by *pies* proportional to their contribution. Notice that pies are characterized by two colors (blue and green): green refers to the contribution to the first factor (horizontal axis) and blue to the second one. Configuration of the points in the graphics is enhanced by many useful bits of information for the analyst. To investigate the masking effect, on the right-hand side only the units with a contribution to the first plane of up to 5 % are shown using the filtering capabilities of VITAMIN-S.

The exhibited pattern of the unit plot is typical in data-mining applications: an overwhelming majority of points shows features similar to the average unit. This involves a masking effect in revealing interesting variability. The next step is to make these units supplementary. Removing units or variables is a simple and fast task with VITAMIN-S: one simply selects and cuts the points on the screen. A removal implies that factors are recalculated: this is immediately done in VITAMIN-S and displays are updated in *real time*. The UNDO functionality allows the user to go forward and backward in the different configurations. By evaluating changes in distances among points and absolute contributions the user understands the relationships in the data.

After the deletion, we take a look at the variable and unit configuration in Fig. 4.13. From the variable representation it is evident that a remarkable change occurred: on the first axis two variables move to the right side of the plane while the variables are again split on the second factor.

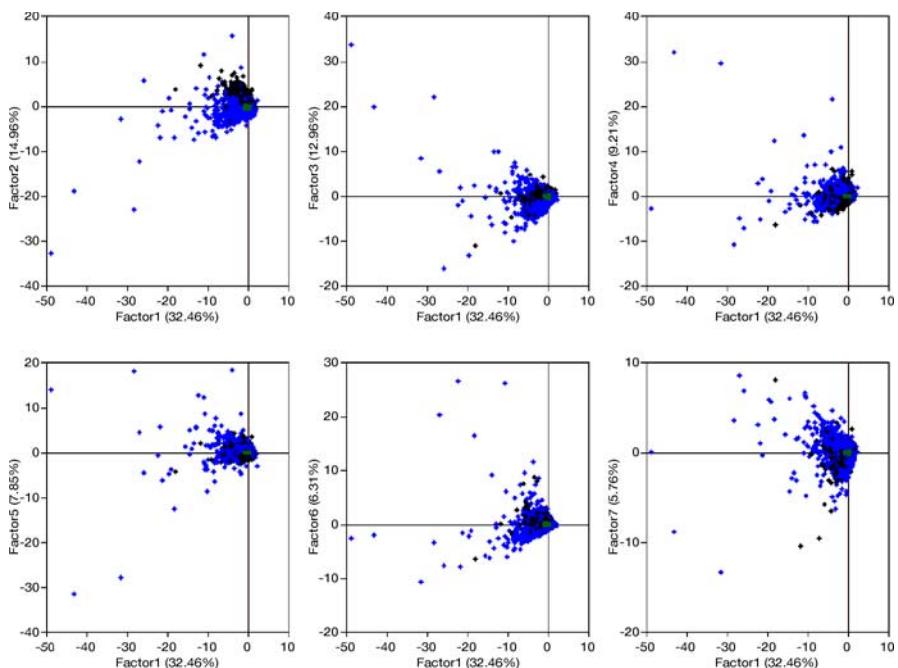


Figure 4.11. Multiple factorial plane representation (SHIW data)

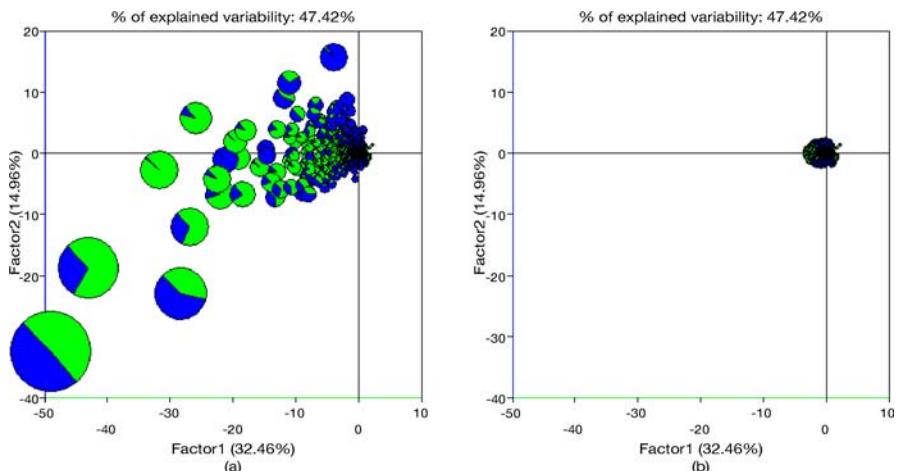


Figure 4.12. (a) Unit representation using contribution. (b) Units with contribution $\leq 5\%$ (SHIW data)

Another typical strategy in data-mining exploration consists in studying the influence of anomalous observations. Using again the filtering capabilities of VITAMIN-S, units with a related contribution greater than 70 % are isolated in Fig. 4.14.

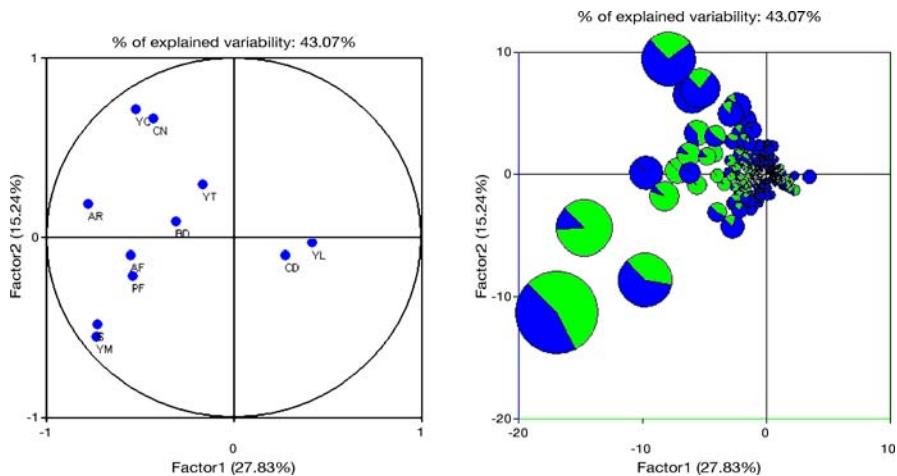


Figure 4.13. Variable and unit representations after the deletion of the less important units

In exploratory MDA, to evaluate the influence of anomalous observations, statisticians remove units having the greatest contributions and, projecting these as supplementary points, compare the two-point configurations.

High-contribution units are then made supplementary using the deletion tool and the resulting planes are shown in Fig. 4.15.

It is evident that a remarkable change occurred: while the variables preserves their position on the first axis with respect to Fig. 4.13, they move on the second factor, that is, they are more influenced by anomalous observations.

The user can reiterate the point deletion step or, if the point configuration is believed to be sufficiently stable, go to the analysis of the classification step.

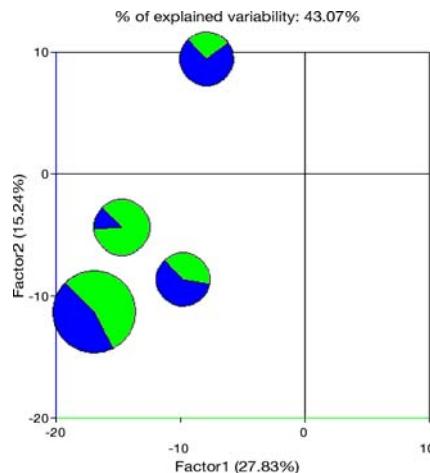


Figure 4.14. Units with a contribution $> 70\%$ (SHIW data)

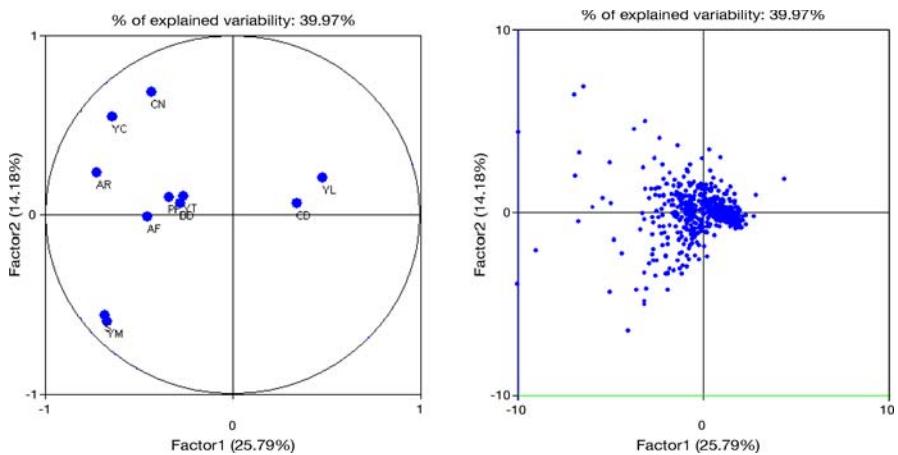


Figure 4.15. Variable and unit representations after the deletion of the most important units

VITAMIN-S was expressly designed to perform exploratory analysis also on large and very large datasets, so it can exploit hierarchical and nonhierarchical cluster analysis algorithms. The current example, consistent with the structure of the contribution, is realized using an agglomerative hierarchical approach: Ward's classification algorithm. As described in Sect. 4.5, cluster analysis assumes as input data the unit coordinates on the factors (the principal components). Finally, the exploratory process ends with a dendrogram display (left-hand side in Fig. 4.16), whereas the analysis process goes on to obtain cluster descriptions. In this phase there are several different data and analysis parameters the user can set. Looking at the dendrogram, we notice the yellow vertical line. Moving the line from right to left, the user can set the classification tree cutting point and the number of partition classes as a consequence. At the same time, on another display there is the cluster display on the factorial plan. The user can act on both displays. In particular, when a cluster on the factorial representation is deleted, the set of active units changes and the whole analysis process works in the background to recalculate all the results of the entire analysis. Results are immediately displayed in the updated views.

Finally, when the user is satisfied with the obtained configurations, the user can click on a command button to save all results, graphical and analytical.

Conclusion and Perspectives

4.9

Recent visualization techniques are based on the highly intensive exploitation of PC graphics cards capabilities. We should expect that these capabilities will further improve in coming years. However, the risk is that these representations, which are very attractive from an aesthetic point of view, will not be readable using a statistical approach. Statistical reading of graphical representations must necessarily be

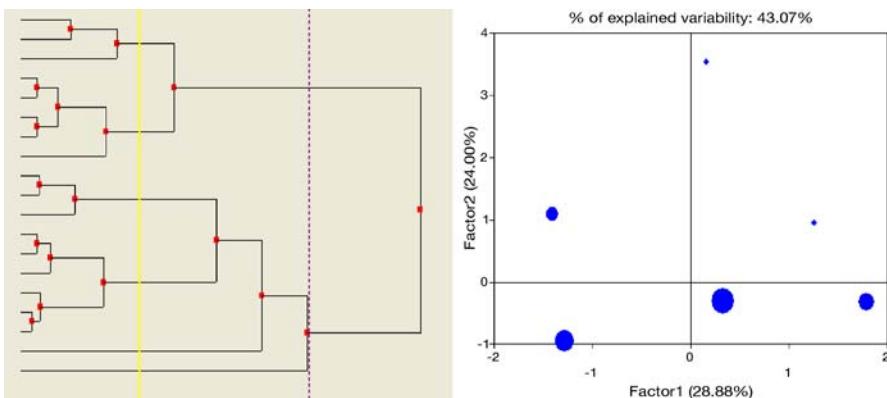


Figure 4.16. Clustering results: dendrogram and cluster displays on the first factorial plan

based on key elements for the correct interpretation (Wilkinson, 2004; Wilkinson et al., 2005). The use of space colors, shapes, and sizes are linked to statistical concepts (Tufte, 1990, 1997b,a, 2001). As in Chernoff's face representations, every detail of the graphic should represent a particular aspect of the data. We can appreciate different and even nicer representations; they could be useful to quickly and easily provide information, but they are not statistical representations. The present contribution has proposed some well-known data representation techniques from MDA in a new light, where human-computer interaction together with modern graphic capabilities can offer simple and highly informative representations.

For the sake of brevity, we did not mention many other statistical methods in the MDA framework that can ensure the same capabilities and are founded on the same paradigm. In the factorial methods framework, it is worth mentioning correspondence analysis (Greenacre, 1984), multidimensional scaling (Cox and Cox, 1994), and Procrustean analysis (Mardia et al., 1979). It is necessary to remark that Euclidean distance is *one* among hundreds of available metrics. The central role of Euclidean distance in geometry captures researchers' attention more than other distances and dissimilarity indexes. As has been pointed out by many authors, the role of distance is prominent and much attention must be paid to this choice.

The choice of classification algorithm would require a wide and deep review of the most recent literature. As a final remark, we would like to stress the fact that the analysis process described in this paper can succeed in exploratory analysis if the same distance, the same *yardstick*, is kept constant through the whole analysis process.

Our example has been realized with the help of VITAMIN-S software; however, using the most recent metalanguages, the same results can be obtained by someone with very little programming experience. Some important computational aspects of exploratory data analysis are treated in the book by Martinez and Martinez (2005).

References

- Antoch, J. (2005). Visual data mining of large data sets using VITAMIN-S system, *Neural Network World* 15(4):283–293.
- Asimov, D. (1985). The grand tour: A tool for viewing multidimensional data, *Journal of Science and Statistical Computing* 6(1):128–143.
- Benzécri, J.-P. (1976). Historie et préhistoire de l'analyse des données, *Callieurs de l'Analyse des Données* 1.
- Bertin, J. (1967). *Semiologie Graphique*, Gauthier-Villars Mouton, Paris. (Also available in English: *Semiology of graphics*, Univ. of Wisconsin Press, 1983).
- Bounford, T. (2000). *Digital Diagrams: How to Design and Present Statistical Information Effectively*, Watson-Guptill Publications.
- Buja, A., Cook, D., Asimov, D. and Hurley, C. (2005). Computational methods for high-dimensional rotations in data visualization, in C.R. Rao, E.J. Wegman and J.L. Solka (eds), *Data Mining and Data Visualization*, Vol. 24 of *Handbook of Statistics*, Elsevier North-Holland, chapter 14, pp. 391–413.
- Buja, A., McDonald, J.A., Michalak, J. and Stuetzle, W. (1991). Interactive data visualization using focusing and linking, *Proceedings of the 2nd conference on Visualization '91*, San Diego, California.
- Card, S.K., Mackinlay, J.D. and Shneiderman, B. (eds) (1999). *Readings in Information Visualization Using Vision to Think*, Morgan Kaufmann, San Francisco, CA.
- Cattell, R. (1978). *The Scientific Use of Factor Analysis in Behavioral and Life Science*, Plenum Press, New York.
- Cleveland, W.S. (1993). *Visualizing Data*, AT&T Bell Laboratories.
- Cleveland, W.S. (1994). *The Elements of Graphing Data*, revised edn, AT&T Bell Laboratories.
- Cleveland, W.S. and McGill, M.E. (eds) (1988). *Dynamic Graphics for Statistics*, Chapman & Hall/CRC.
- Confais, J. and Nakache, J. (2004). *Approche pragmatique de la classification: Arbres hiérarchiques, partitionnements*, Technip, Paris.
- Cook, D., Buja, A., Cabrera, J. and Hurley, H. (1995). Grand tour and projection pursuit, *Journal of Computational and Graphical Statistics* 2(3).
- Cox, T. and Cox, M. (1994). *Multidimensional Scaling*, Chapman&Hall.
- Csinger, A. (1992). The psychology of visualization, *Technical report*, Department of Computer Science , University of British Columbia. Available at http://historical.ncstrl.org/litesite-data/ubc_cs/TR-92-28.ps.gz.
- D'Alessio, G., Faiella, I. and Neri, A. (2004). Italian households budgets in 2002, Bank of Italy, Available at <http://www.bancaditalia.it/statistiche>. Supplements to the Statistical Bulletin (new series).
- Dhillon, I., Modha, D. and Spangler, W. (1998). Visualizing class structure of multidimensional data, *Proceedings of the 30th Symposium on the Interface: Computing Science and Statistics*, Minneapolis, MN.
- Eckart, C. and Young, G. (1936). The approximation of one matrix by another of lower rank, *Psychometrika* 1:211–218.
- Friendly, M. (2000). *Visualizing Categorical Data*, SAS Publishing.

- Friendly, M. and Denis, D. (Visited on Oct. 2005). Milestones in the history of thematic cartography, statistical graphics, and data visualization., Available at <http://www.math.yorku.ca/SCS/Gallery/milestone/>.
- Gordon, A.D. (1999). *Classification*, second edn, Chapman & Hall CRC.
- Greenacre, M. (1984). *Theory and Applications of Correspondence Analysis*, Academic Press, London.
- Hardle, W. and Simar, L. (2006). *Applied Multivariate Statistical Analysis*, e-books, at: <http://www.xplore-stat.de/ebooks/ebooks.html>.
- Hartigan, J.A. (1975). *Clustering Algorithms*, John Wiley & Sons.
- Hotelling, H. (1933). Analysis of a complex of statistical variables into components, *Journal of Educational Psychology* 24.
- Hurley, C. and Buja, A. (1990). Analyzing high-dimensional data with motion graphics, *SIAM Journal on Scientific and Statistical Computing* 11(6):1193–1211.
- Inselberg, A. (1999). Don't panic ...just do it in parallel!, *Computational Statistics* 14:53–77.
- Jolliffe, I.T. (2002). *Principal Component Analysis*, Springer Series in Statistics, 2nd edn, Springer.
- Klinke, S. (2004). Statistical user interfaces, in J.E. Gentle, W. Hardle and Y. Mori (eds), *Handbooks of Computational Statistics. Concepts and Methods*, Springer-Verlag, Heidelberg, pp. 379–401.
- Kohonen, T. (2000). *Self-Organizing Maps*, 3rd edn, Springer.
- Lebart, L., Morineau, A. and Piron, M. (1995). *Statistique exploratoire multidimensionnelle*, Dunod, Paris.
- Lebart, L., Morineau, A. and Warwick, K.M. (1984). *Multivariate Descriptive Statistical Analysis*, Wiley & Sons.
- Mardia, K., Kent, J. and Bibby, J. (1979). *Multivariate Analysis*, Academic Press, London.
- Martinez, W.L. and Martinez, A.R. (2005). *Exploratory data analysis with MATLAB*, Chapman & Hall CRC, Boca Raton.
- Bank of Italy (2004). Survey on household income and wealth, Rome, Italy. Available at <http://www.bancaditalia.it/statistiche>.
- Milligan, G.W. (1996). Clustering validation: results and implication for applied analysis, in P. Arabie, L.J. Hubert and G.D. Soete (eds), *Clustering and Classification*, World Scientific Publishing, Singapore, pp. 341–375.
- Mizuta, M. (2004). Dimension reduction methods, in J.E. Gentle, W. Hardle and Y. Mori (eds), *Handbooks of Computational Statistics. Concepts and Methods*, Springer-Verlag, Heidelberg, pp. 565–589.
- Shneiderman, B. (1996). The eyes have it: A task by data type taxonomy for information visualization, *Proceedings of the IEEE Symposium on Visual Languages '96*, IEEE Press, Los Alamitos, CA, pp. 336–343.
- Symanzik, J. (2004). Interactive and dynamic graphics, in J.E. Gentle, W. Hardle and Y. Mori (eds), *Handbooks of Computational Statistics. Concepts and Methods*, Springer-Verlag, Heidelberg, pp. 293–336.
- Tufte, E.R. (1990). *Envisioning Information*, Graphics Press, Cheshire (Connecticut).

- Tufte, E.R. (1997a). *Visual & Statistical Thinking: Displays of Evidence for Decision Making*, Graphics Press, Cheshire (Connecticut).
- Tufte, E.R. (1997b). *Visual Explanations: Images and Quantities, Evidence and Narrative*, Graphics Press, Cheshire (Connecticut).
- Tufte, E.R. (2001). *The Visual Display of Quantitative Information*, 2nd edn, Graphics Press, Cheshire (Connecticut).
- Tukey, J.W. (1962). The future of data analysis, *Ann. Math. Statist.* 33:1–67.
- Tukey, J.W. (1977). *Exploratory Data Analysis*, Addison-Wesley, Reading, Mass.
- Unwin, A. (1999). Requirements for interactive graphics software for exploratory data analysis, *Computational Statistics* 14:7–22.
- Unwin, A., Theus, M. and Hofmann, H. (2006). *Graphics of Large Datasets. Visualizing a Million.*, Springer Series in Statistics and Computing, Springer.
- Vichi, M. and Kiers, H.A. (2001). Factorial k-means analysis for two-way data, *Computational Statistics & Data Analysis* 37:49–64.
- VITAMIN-S – Functional specification report (2002). VIIsual daTA MINinig System, Deliverable 3.1, DMS University of Naples (Italy) & ATKOSoft S.A. (Athens, Greece). EU FP5 Proj. N. IST-2000-26341.
- Wainer, H. and Velleman, P.F. (2001). Statistical graphics: Mapping the pathways of science, *Annu. Rev. Psychol.* 52:305–335.
- Wegman, E.J. (2003). Visual data mining, *Statistics in Medicine* 22:1383–1397.
- Wegman, E.J. and Solka, J.L. (2005). Statistical data mining, in C.R. Rao, E.J. Wegman and J.L. Solka (eds), *Data Mining and Data Visualization*, Vol. 24 of *Handbook of Statistics*, Elsevier, chapter 1, pp. 1–46.
- Wilhelm, A. (2005). Interactive statistical graphics: the paradigm of linked views, in C.R. Rao, E.J. Wegman and J.L. Solka (eds), *Data Mining and Data Visualization*, Vol. 24 of *Handbook of Statistics*, Elsevier North-Holland, pp. 437–537.
- Wilkinson, L. (2004). The grammar of graphics, in J.E. Gentle, W. Hardle and Y. Mori (eds), *Handbooks of Computational Statistics. Concepts and Methods*, Springer-Verlag, Heidelberg, pp. 337–377.
- Wilkinson, L., Wills, D., Rope, D., Norton, A. and Dubbs, R. (2005). *The Grammar of Graphics*, Statistics and Computing, 2nd edn, Springer.
- Yang, L. (1999). 3D grand tour for multidimensional data and clusters, in D. Hand, J. Kok and M. Berthold (eds), *Advances in Intelligent Data Analysis. IDA-99*, Vol. 1642 of LNCS, Springer-Verlag, p. 173.

Multivariate Visualization by Density Estimation

III.5

Michael C. Minnotte, Stephan R. Sain, David W. Scott

5.1	<i>Univariate Density Estimates</i>	390
	Histograms.....	390
	Improved Binned Density Estimates	393
	Kernel Density Estimates	394
	Kernel Variants	397
	Multiscale Visualization of Density Estimates.....	400
5.2	<i>Bivariate Density Estimates</i>	401
	Bivariate Histograms	402
	Bivariate Kernel Density Estimators.....	404
5.3	<i>Higher-dimensional Density Estimates</i>	406

Density estimation and related methods provide a powerful set of tools for visualization of data-based distributions in one, two, and higher dimensions. This chapter examines a variety of such estimators, as well as the various issues related to their theoretical quality and practical application.

The goal of understanding data leads to the notion of extracting as much information as possible. This begins by understanding how each individual variable varies. If the parametric form is well known, then a few statistics answer the question. However, if the parametric form is unknown, then visual examination of a well-constructed nonparametric density estimate is the recommended route. In this way, features of the density, such as the number and location of modes, can be identified. With multivariate density estimates, we can extend this ability to understand the distributional relationships between variables in two or more dimensions.

5.1 Univariate Density Estimates

Given a univariate sample $\{x_1, \dots, x_n\} \sim f(x)$ of an unknown parametric form, visualization of an estimate of f is an important part of the analysis process for multiple reasons. It allows for direct examination of possibly important structure in f , such as skewness or multiple modes. In addition, it provides for a means of considering assumptions such as that of normality for further analysis. Such visualization can provide an alternative to a formal goodness-of-fit test, particularly for large sample sizes where such tests may reject even quite reasonable models.

5.1.1 Histograms

The form of density estimation most familiar to analysts is the univariate histogram. While it owes its popularity to its simplicity, the histogram can provide a serviceable, if crude, idea of a dataset's distribution.

Construction of a histogram requires a mesh of bin edges, $\{t_0 < t_1 < t_2 < \dots < t_k\}$, covering the range of the data. Define the j th bin as $B_j = [t_j, t_{j+1})$, its width as $h_j = t_{j+1} - t_j$, and its count as $v_j = \sum_{i=1}^n I_{B_j}(x_i)$, where $I_A(x)$ is an indicator function, taking 1 if $x \in A$ and 0 otherwise. A frequency histogram plots

$$\hat{g}_1(x) = v_j \quad x \in B_j, \tag{5.1}$$

while a percentage histogram plots

$$\hat{g}_2(x) = \frac{100 v_j}{n} \quad x \in B_j. \tag{5.2}$$

As long as the widths for all bins equal a constant $h_j = h$, either of these will give a reasonable visual depiction. But as neither integrates to 1, neither is a true density estimate. If bin widths vary, neither frequency nor percentage histograms should be used, as they will give excessive visual weight to the larger bins.

A density histogram, in contrast, plots

$$\hat{f}(x) = \frac{v_j}{nh_j} \quad x \in B_j. \quad (5.3)$$

A simple calculation demonstrates that the density histogram integrates to 1 and is therefore a true density, and it can be shown to provide a reasonable estimate of f . The use of h_j in rescaling means that f will be well estimated, even with varying bin widths.

See Fig. 5.1 for an example of a density histogram. The data consist of average winter (December, January, and February) temperature minimums, in degrees Celsius, for $n = 16,236$ grid points in the US state of Colorado over the period 1995–2004. This 10-year average of temperature minimums, as well as average temperature maximums (degrees Celsius) and total precipitation (millimeters), were constructed from monthly datasets available from the Spatial Climate Analysis Service at Oregon State University (<http://www.ocs.oregonstate.edu/prism/>). These high-resolution, gridded datasets are based on the parameter-elevation regressions on the independent slopes model (PRISM) discussed in Daly et al. (1994) and Gibson et al. (1997) that incorporates station-level meteorological data, a digital elevation model, as well as other spatial datasets, in a type of “expert system” designed to represent how an experienced climatologist would create a climate map.

The histogram in Fig. 5.1 clearly shows a bimodal structure relating to the geography of the state, a sharp mode to the right for the eastern plains, and a shorter, broader mode indicating the lower temperatures of the mountain grid points.

The choice of bin width plays a critical role in the appearance and accuracy of the histogram. Theoretical analysis of the density histogram involves the multinomial distribution and a Taylor’s series on the true distribution f ; see Scott (1992) for details. The variance of $\hat{f}(x)$ is dominated by a term of the form $f(x)/nh_j$, for x in B_j ,

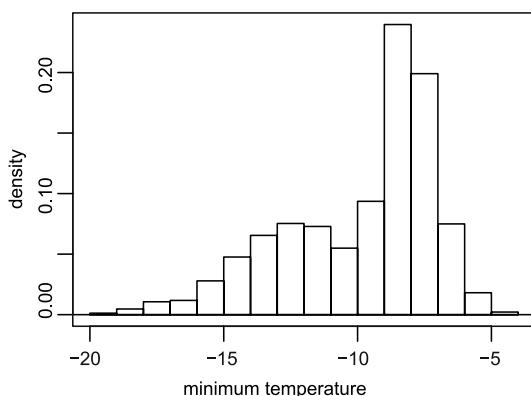


Figure 5.1. Density histogram of average minimum winter temperature (degrees Celcius) over a grid of locations in the US state of Colorado

indicating that the variance will be reduced for large bin width. On the other hand, the bias of $\hat{f}(x)$ for $x \in B_j$ is dominated by

$$\left(\frac{t_{j+1} + t_j}{2} - x \right) f'(x) = (m_j - x) f'(x), \quad (5.4)$$

where m_j is the midpoint of bin B_j . Not surprisingly, the piecewise-constant histogram has greatest bias in bins where the true density has the largest (positive or negative) slope. However, this effect can be reduced by the use of smaller bins, since $|m_j - x| < h_j/2$ for all $x \in B_j$.

Clearly, no single bin width will perform optimally for both variance and bias, but we can balance these competing forces by considering the mean integrated squared error (MISE), found as the sum of the integrated variance and the integrated square of the bias. Optimization over a single bin width h suggests that asymptotic MISE will be minimized for bin width

$$h^* = \left(\frac{6}{R(f')} \right)^{1/3} n^{-1/3}, \quad (5.5)$$

where the “roughness” functional, R , is defined by $R(g) = \int g(x)^2 dx$. Unfortunately, the presence of $R(f')$ limits the applicability of this rule, as it is highly unlikely to be known when f itself must be estimated.

As a more practical alternative, many computer packages follow a recommendation by Sturges (1926) that the number of bins be roughly $1 + \log_2(n)$. Sturges chose the bin counts $v_j = \binom{K-1}{j}$ for $j = 0, 1, \dots, K-1$, so that

$$n = \sum_{j=0}^{K-1} v_j = (1+1)^{K-1} = 2^{K-1}. \quad (5.6)$$

Hence the number of bins $K = 1 + \log_2(n)$. Sturges’ bin counts are proportional to the binomial $B(K-1, \frac{1}{2})$ probabilities, which is approximately normal for moderate K . Hence, Sturges’ rule is a version of a normal reference rule but motivated by binomial approximations rather than by mean squared error considerations.

While Sturges’ rule gives fairly reasonable widths for small samples from smooth densities, the number of bins increases (and therefore h decreases) at a rate far slower than optimal for AMISE purposes. Better rules replace $R(f')$ in the theoretical formula with the value for a normal distribution, resulting in $h^* = 3.5\sigma n^{-1/3}$. Scott (1979) suggests using the sample standard deviation s in $h_S = 3.5 s n^{-1/3}$, while Freedman and Diaconis (1981) suggest using the more robust interquartile range IQR in $h_{FD} = 2 IQR n^{-1/3}$. Usually, $h_{FD} < h_S$, since $\sigma = IQR/1.35$ for the normal density and hence $3.5\sigma = 2.6 IQR$, which is 30 % wider than h_{FD} . Although less oversmoothed than the estimates generated by Sturges’ rule, estimates using the normal-based rules can still be smoother than optimal for more complicated underlying densities. This may not be terrible; oversmoothed histograms often look better to many viewers, as the bias of overly large bins can be easier to mentally smooth out than the multiple small modes that can appear in undersmoothed estimates. Nonetheless, the strong ef-

fect of bin width suggests that for visualization purposes, it is wise to view a collection of histograms with more than one choice of bin width; a particular sequence recommended in practice is $h = h_S/1.05^k$ for $k = 0, 1, \dots$, until h is obviously too narrow.

The exact locations of the bin edges does not enter into the expression for h^* because the bin edge is generally theoretically less important than the bin width. The exception occurs when f has a known point of discontinuity. Placing a bin edge at such a point will provide a simple, automatic adjustment. As 0 is frequently a point of discontinuity, selecting bin edges as $t_0 = 0, t_j = jh$, is often a wise choice. Despite the lower-order theoretical effect, placement of bin edges can have substantial visual effect on histograms of moderate sample size. As with bin width, inspecting multiple histograms with different bin edges is highly recommended. A JAVA applet is available in the Rice Virtual Lab in Statistics to explore the interaction of bin width and bin edge selections; see the histogram applet at <http://www.ruf.rice.edu/~lane/rvls.html>. Features that appear in most or all alternative histogram views should be given much more credence than those that appear only once.

Improved Binned Density Estimates

5.1.2

Visual and theoretical improvement can be obtained in density estimates from histogram-style binned data through the use of interpolation. The oldest such technique is the frequency polygon, generated by linearly interpolating histogram bin centers. By this means, the slope of f may be tracked, and the bias improves from $O(h)$, depending on $f'(x)$, to $O(h^2)$, depending on $f''(x)$; details may be found in Scott (1992).

The edge frequency polygon of Jones et al. (1998), instead interpolates the histogram bin edges, at heights representing the averages of adjacent histogram bin heights. The result reduces variance and optimal MISE, at the cost of a small increase in bias.

Minnotte's (1996) biased-optimized frequency polygon interpolates histogram bin centers, but at heights calculated to ensure that the multinomial probabilities represented by the bin data proportions are maintained. Although the estimates may go negative, and have higher optimal MISE properties than the edge frequency polygon, their minimal bias recommends them in cases where large amounts of data are collected into coarse bins and no finer binning is possible. These can be improved still further by interpolating with cubic or higher-order splines. The resulting higher-order histosplines (Minnotte, 1998) achieve $O(h^4)$ or higher levels of bias, and can strongly outperform other estimates when large samples are prebinned into wide bins.

Figure 5.2 shows examples of the standard frequency polygon, edge frequency polygon, bias-optimized frequency polygon, and cubic higher-order histospline computed from the histogram information of Fig. 5.1.

The effects of the bin origin nuisance parameter can be minimized by computing several histograms, each with the same bin width, but with different bin origins. Scott's (1985) averaged shifted histogram (ASH) is a weighted average of m histograms, $\hat{f}_1, \hat{f}_2, \dots, \hat{f}_m$, all constructed with the same bin width, h , but with shifted

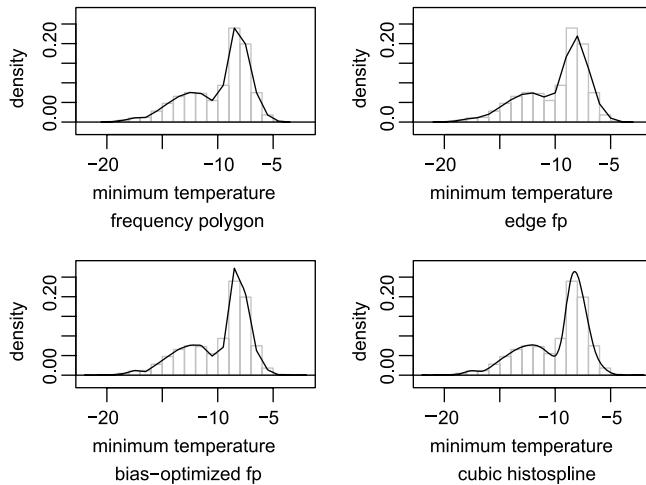


Figure 5.2. Frequency polygon and variants constructed from the histogram information of Fig. 5.1

bin origins, $\{0, h/m, 2h/m, \dots, (m-1)h/m\}$. The data are prebinned into intervals of width $\delta = h/m$. Let the bin count v_k denote the number of points in bin $B_k = ((k-1)\delta, k\delta]$. Then the equally weighted ASH is defined by the equation

$$\hat{f}(x) = \frac{1}{m} \sum_{j=1}^m \hat{f}_j(x) \quad x \in B_k . \quad (5.7)$$

Using weights $w_m(i)$, the weighted ASH becomes

$$\hat{f}(x) = \frac{1}{nh} \sum_{j=-m}^m w_m(j) v_{k+j} \quad x \in B_k . \quad (5.8)$$

Figure 5.3 shows the effect of equally weighted averaging of increasing numbers of histograms for the data and bin width of Fig. 5.1.

5.1.3 Kernel Density Estimates

The bin origin can be eliminated altogether by the use of a kernel density estimate. The result is superior theoretically as well as smoother and thus more appealing visually.

The estimate requires a smoothing parameter, h , that plays a role similar to that of the bin width of a histogram and that is sometimes referred to as the bandwidth of the estimate. It also requires a kernel function, K , which is usually selected to be a probability density function that is symmetric around 0.

From these, the estimator may be written as

$$\hat{f}(x) = \frac{1}{nh} \sum_{i=1}^n K\left(\frac{x-x_i}{h}\right) = \frac{1}{n} \sum_{i=1}^n K_h(x-x_i) , \quad (5.9)$$

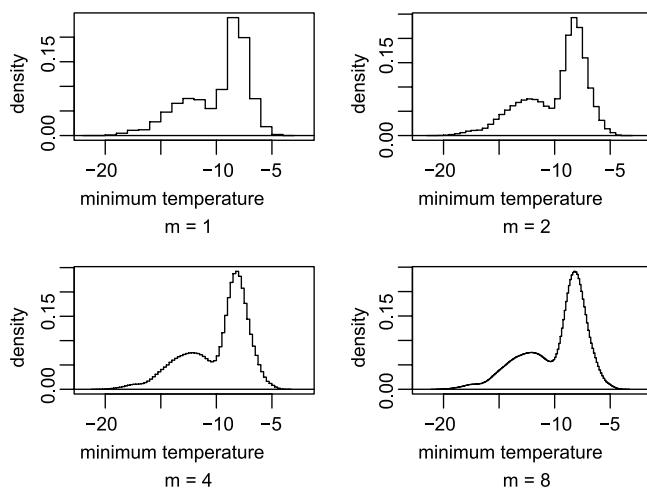


Figure 5.3. Averaged shifted histograms constructed from the data and bin width of Fig. 5.1 with increasing numbers of averaged bin edges

with $K_h(t) = K(t/h)/h$. The estimate is the sum of a collection of n probability masses, each with shape K and size n^{-1} , centered on the observations. Figure 5.4 demonstrates the construction of such an estimate for the percentage of silica in 22 chondrite meteorites (from Ahrens, 1965), with a standard normal kernel and a bandwidth of 1.

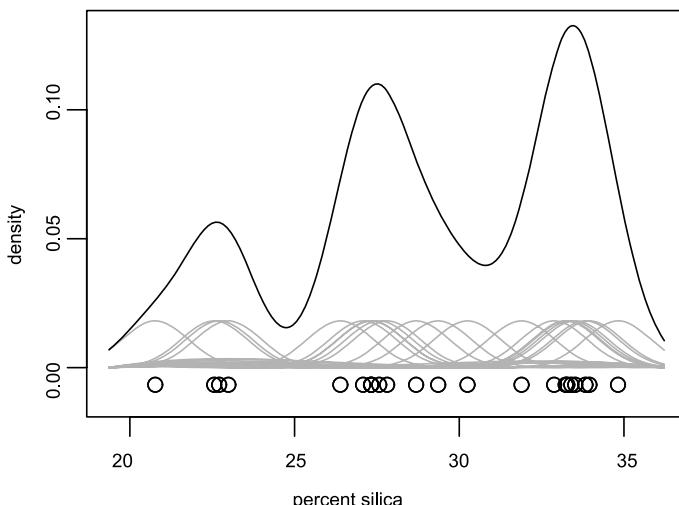


Figure 5.4. Example of kernel density estimate construction for percent silica in chondrite meteorites. The original observations and individual kernel functions are displayed. The final estimate is the vertical sum of all $n=22$ kernel functions

Choice of kernel is relatively unimportant, as any reasonable p.d.f. will provide a similar picture. Optimal mean integrated squared error may be achieved using the Epanechnikov kernel, a truncated parabola with form

$$K_E(t) = \frac{3}{4}(1 - t^2) \quad -1 \leq t \leq 1. \quad (5.10)$$

The discontinuous derivative at ± 1 is inherited by the estimate at numerous points, so for visual purposes this choice of kernel may not be ideal. The biweight kernel,

$$K_B(t) = \frac{15}{16}(1 - t^2)^2 \quad -1 \leq t \leq 1, \quad (5.11)$$

is nearly as efficient and has a continuous first derivative, so it is often preferred for the smoother appearance of its estimates. (The triweight kernel, $K_T(x) = 35(1 - t^2)^3/32$, has a continuous second derivative.) Finally, the standard normal kernel, $K_N(t) = \phi(t)$, is smoother still and often preferred for its infinite number of continuous derivatives and its uniquely well-behaved mode structure (Sect. 5.1.5).

Choice of the bandwidth h is more critical and, as with a histogram, can have a very strong effect on the resulting estimate. Large h leads to oversmoothing and (asymptotic) bias proportional to $h^2 f''(x)$, while small h leads to an undersmoothed, highly multimodal estimate with (again asymptotic) variance proportional to $f(x)/nh$. Optimization shows that the asymptotic mean integrated squared error may be minimized by choosing $h^* = c_K [R(f'')n]^{-1/5}$, with c_K depending on the kernel and equaling 1.719, 2.036, and 0.776 for the Epanechnikov, biweight, and normal kernels, respectively. As with the histogram, the presence of a function of f in the optimal choice of h leads to a requirement of alternative approximations.

One approach is again to assume a normal distribution for f . This leads to the normal reference rule $h = c_{NK} \hat{\sigma} n^{-1/5}$, for $c_{NK} = 2.35, 2.78$, and 1.06 for the Epanechnikov, biweight, and normal kernels. This will be oversmoothed for most nonnormal densities, so an initial estimate with extreme skewness or multiple strong modes may argue for a second estimate with smaller h . Figure 5.5 demonstrates the effect of h with three normal kernel estimates for a subset of 250 points of the minimum temperature data from Fig. 5.1. The normal reference rule suggests that $h = 1.01$ for this data, but the strong bimodality indicates that a smaller choice, such as the middle $h = 0.5$ example, is probably more appropriate. Taking h smaller still, as in the rightmost $h = 0.25$ estimate of Fig. 5.5, leads to a clearly undersmoothed estimate. Note that the individual kernels at the bottom of each plot are scaled correctly relative to one another, but not to their full estimates. For visibility, each has been rescaled vertically to 10 times the height of a single kernel used for this sample.

A variety of more highly computational approaches for data-based bandwidth selection have also been proposed. These include variations on cross-validation [including Scott and Terrell (1987), Hall and Marron (1988), and Sain et al. (1994)] as well as methods using pilot estimates of $R(f'')$ [such as Sheather and Jones (1991) and Hall et al. (1991)]. While these can lead to improved choice of h and accordingly

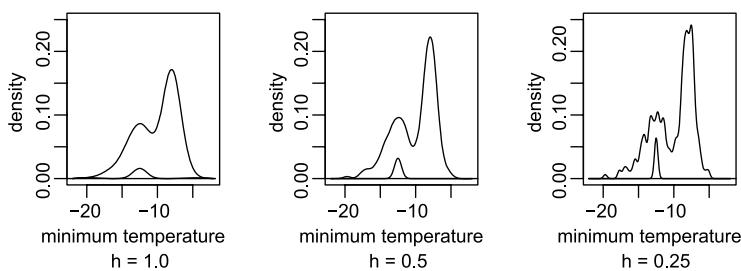


Figure 5.5. Effect of h on kernel density estimates of a subset of the minimum temperature data

better estimates, the results can be unstable and should not replace the examination of multiple estimates by the skilled analyst.

One advantage kernel estimates have over simpler density estimates such as histograms is the ease with which they can be compared across different datasets. Kernel density estimates are continuous and hence are well suited to overplotting with different colors or line types, which makes visual comparisons between different groups simple and effective.

An example is shown in Fig. 5.6. The black curve represents a kernel density estimate of the temperature midpoint from the PRISM data for grid cells in Colorado using a normal kernel and normal reference bandwidth. The grey curve represents a similarly constructed kernel estimate off the temperature midpoint for the grid cells in the neighboring state of Kansas. The bimodality in the kernel estimate for Colorado is attributable to the differences between the eastern plains and the mountain

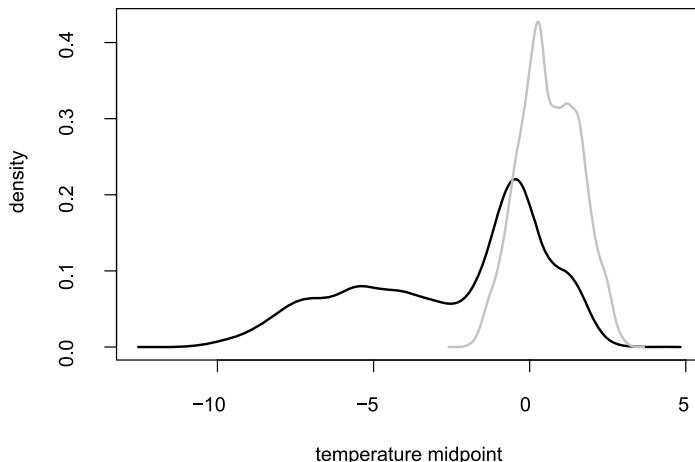


Figure 5.6. Univariate kernel density estimates of average temperature midpoint based on the PRISM data. Black line: grid points in Colorado; grey line: Kansas

regions. The kernel estimate for Kansas has a dominant mode consistent with the eastern plains of Colorado.

5.1.4

Kernel Variants

Minor adjustments allow for estimation of density derivatives as well as the function itself. The derivative of \hat{f} is a reasonable estimate of the derivative of f and can be estimated directly as

$$\hat{f}'(x) = \frac{1}{nh^2} \sum_{i=1}^n K' \left(\frac{x - x_i}{h} \right). \quad (5.12)$$

The variance of this estimate is inherently greater than that of \hat{f} , and the optimal choice of h is correspondingly larger.

Returning to estimates of f , a variety of approaches are available for reducing the bias of a kernel estimate. See Jones and Signorini (1997) for a survey and comparison of some of the most common higher-order methods. As a rule, most of these will have fairly minimal effect on the visual impact of the estimate.

The oldest and best-known higher-order methods replace the p.d.f. kernel K with one involving carefully computed negative regions so as to have second, and possibly higher, moments equal to 0 (Bartlett 1963). For example, starting with standard (second-order) kernel K , let

$$s_m = \int t^m K(t) dt. \quad (5.13)$$

Then the function

$$K_{(4)}(t) = \left(\frac{s_4 - s_2 t^2}{s_4 - s_2^2} \right) K(t) \quad (5.14)$$

will have a 0 second moment and be a fourth-order kernel. Using $K_{(4)}(t)$ in place of $K(t)$ will reduce the bias of the estimate. Unfortunately, such an estimate will frequently include undesirable visual artifacts, including extra modes and negative lobes in the tails of the density estimate, so is not recommended for visualization purposes.

Bias-reduction methods involving “variable location” approaches (Samiuddin and El-Sayyad, 1990; Hall and Minnotte, 2002) provide better solutions. Instead of replacing the kernel K , these methods use one of our good second-order kernels but center them on transformed values, $\gamma(x_i)$, rather than on the raw x_i s themselves. These transformations depend upon pilot estimates of f and its derivatives in a way that will reduce the bias. For example, replacing x_i with

$$\gamma(x_i) = x_i + h^2 \frac{s_2}{2} \frac{\hat{f}'(x_i)}{\hat{f}(x_i)} \quad (5.15)$$

provides fourth-order bias and improved estimation near local extrema, but the use of the standard kernel guarantees positivity and fewer extraneous bumps than found with the fourth-order kernel.

Yet another approach to bias reduction relies on variable bandwidths. By choosing $h_i = h(x_i)$ proportional to $\hat{f}^{-1/2}(x_i)$, Abramson (1982) demonstrated that this is yet another way to improve the order of the bias; see also Terrell and Scott (1992) and Sain and Scott (1996). This approach also has much to recommend it for density visualization. By using small bandwidths near modes we sharpen the estimates where bias is greatest. At the same time, wide kernels for isolated points in the tails of the density lead to smoother estimates with fewer spurious modes there.

A compromise between the single bandwidth of a standard kernel estimate and the n bandwidths of a variable bandwidth estimate may be found in the filtered kernel technique of Marchette et al. (1996). The authors use an initial normal mixture distribution estimate with a small number of components to select bandwidths proportional to the component standard deviations. Kernels with each bandwidth are averaged for each x_i with weights proportional to their component densities at x_i . In this way, the smoothness of each region may be adjusted individually to emphasize large, important features, while deemphasizing unimportant features such as the many minor modes that may often be observed in the tails of densities.

Figure 5.7 shows these variants for the data of Fig. 5.5. For the fourth-order and variable-location estimates, the bandwidth is $h = 0.5$, while the variable-bandwidth and filtered estimates have the same value for the geometric averages of their bandwidths. Each panel also includes the standard kernel estimate in grey for comparison (as in the middle panel of Fig. 5.5). All of the variants strengthen and sharpen the large mode on the right. The fourth-order and variable-location estimates also strengthen the smaller bumps and modes in the left tail, while the variable-bandwidth and filtered estimates deemphasize the same.

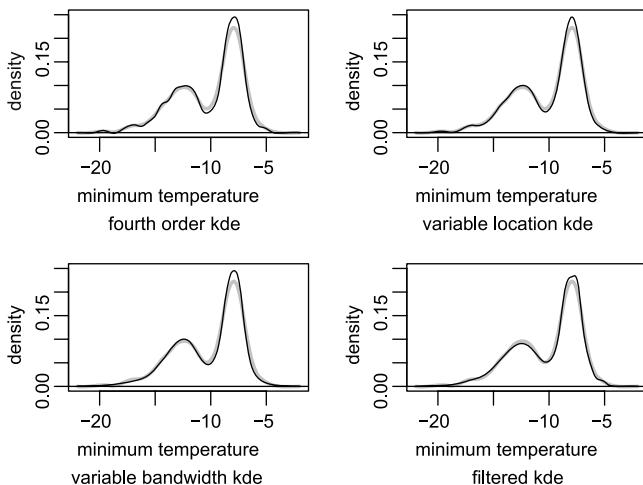


Figure 5.7. Variant kernel density estimates for minimum temperature data

Multiscale Visualization of Density Estimates

The importance of the smoothing parameter in the appearance of density estimates has led to some additional visualization tools that examine a dataset at a wide variety of smoothing levels.

Recognizing that the most visually striking and important aspect of a density estimate is often the number and location of modes, Minnotte and Scott (1993) proposed plotting those against the bandwidth for a set of kernel estimates using a wide range of smoothing parameters. As the number of modes is roughly monotone decreasing in the bandwidth (strictly, if the normal kernel is used), the authors called the resulting plot the “mode tree”. The mode tree is highly effective for examining modal behavior under varying levels of smoothing for a dataset or density estimation method. Unless the modal behavior of the density estimation method is the effect of interest, the normal kernel is recommended. As Silverman (1981) showed, the number of modes is nonincreasing in h for a normal kernel density estimate. Visually, this means that the mode tree is well behaved in this case, with mode traces appearing as you continue down the tree and continuing once they appear. Minnotte and Scott (1993) demonstrate that this is not the case for estimates derived from nonnormal kernels.

Minnotte and Scott (1993) also demonstrate how additional features may be added for an “enhanced mode tree,” including locations of antimodes and inflection points, measures of sizes of modes, and regions of positive and negative second derivatives (the latter are often called “bumps”). In this way, a great deal of information about the behavior of the density estimates may be examined without restriction to a single bandwidth, or even a small set of them.

Use of the filtered kernel technique of Marchette et al. (1996) leads to the “filtered mode tree” of Marchette and Wegman (1997). The filtering reduces the visual importance of minor modes in the tail of the density while inflating the prominence of large central modes.

Overplotting the bumps (regions of negative second derivative) over different bandwidths for multiple resamples, subsamples, or jittered versions of the data leads to the “mode forest” of Minnotte et al. (1998). Again, the effect is a kind of visual inference, emphasizing large central modes, while deemphasizing those minor ones in the tails of the density.

Finally, Chaudhuri and Marron (1999) combined the ideas of the mode tree, “scale space” from computer vision research (which is closely related to smoothing parameter variation), and some simple inference to propose SiZer (for Significant Zero crossings). At each location-by-bandwidth pixel, one of three colors is plotted depending on the estimated density slope – significantly positive, significantly negative, or not significantly different from zero. The resulting patterns may be examined for modality information.

Figure 5.8 shows the mode tree, filtered mode tree, subsampled mode forest, and SiZer plot for the minimum temperature data of Fig. 5.5. For the latter, increasingly dark grey levels indicate significantly positive slope, nonsignificant slope, and significantly negative slope, while white identifies regions of insufficient data density for inference. Note the emphasis on the bimodal structure in each of the four plots, al-

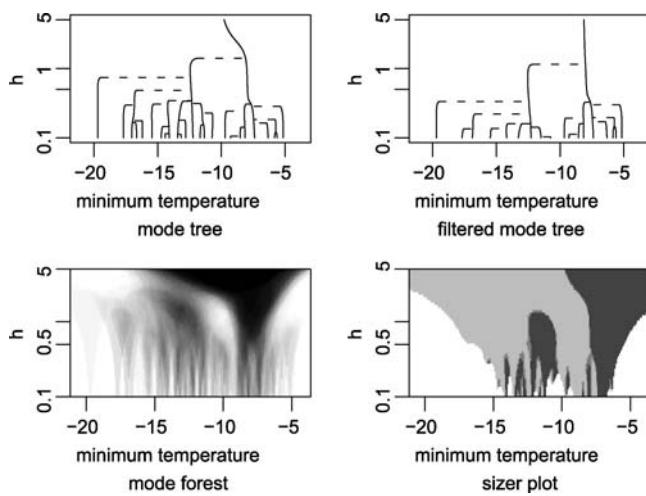


Figure 5.8. Multiscale visualization methods of minimum temperature data. Darker regions of the mode forest represent higher-confidence “bumps” while light, medium, and dark regions on the sizer plot represent, respectively, significantly positive, nonsignificant, and significantly negative slopes in “scale space”

though there is suggestion of a more complicated structure as well. The full complexity of the mode behavior may be seen in the basic mode tree, while the filtered mode tree emphasizes the two key modes as found in its underlying mixture model. The dark regions of the mode forest perform implicit inference as they indicate bumps found in many or most of the subsamples of the full data, while the SiZer plot makes the inference explicit, with alternating light and dark regions strongly suggesting the presence of underlying modes.

Bivariate Density Estimates

5.2

The basic tool for graphically exploring the bivariate distribution between pairs of measurements is the well-known scatterplot. A scatterplot is constructed by simply plotting the pairs of points in the coordinate plane. Generally, some functional relationship between the variable represented by the y -axis and the variable represented by the x -axis is not assumed. However, it is precisely the power of the scatterplot to visualize such relationships that makes it so useful a tool for data analysis.

Friendly and Dennis (2005) present an excellent treatise on the scatterplot and note that early authors recognized the need to augment the scatterplot to account for repeated values in the data, such as glyphs and different plot characters to indicate a multiplicity of points with the same observed value. Cleveland and McGill (1984), Scott (1992), and others have also noted this and many of the modifications and enhancements that have been proposed to convey additional information in scatter-

plots. These authors have also noted the difficulties that arise due to the overplotting of characters designed to represent a single observation.

This phenomenon, referred to as “too much ink” by Tufte (1983), has given rise to the use of more formal density estimators, in particular the histogram and the kernel density estimator, to visualize bivariate distributions and relationships in data. Bivariate density estimates remain straightforward to calculate but require more sophisticated visualization techniques to plot.

5.2.1

Bivariate Histograms

Bivariate histograms share many of the strengths and weaknesses of their univariate cousins. They remain simple to calculate, with the computational effort being primarily focused on determining the counts of the observations in what are now 2-D bins. Bin size and location issues remain important, although there is also the additional issue of the shape of the bivariate bins. Again, one should examine multiple examples when possible to discover what are true features and what are artifacts of the histogram mesh.

For a rectangular mesh, an asymptotic analysis of the MISE, similar to the univariate case, shows that the optimal size of the edges of the bivariate bins is proportional to $n^{-1/4}$. Assuming uncorrelated and normally distributed data gives rise to the normal reference rule $h_k^* = 3.5\sigma_k n^{-1/4}$ for $k = 1, 2$ and where σ_k is the standard deviation for the k th variable. See Scott (1992) for details.

Of particular interest in the construction of bivariate histograms is the notion of an optimal bin shape. Three types of regular bins are possible: rectangular, triangular, and hexagonal. Scott (1988) compared the three shapes and showed that the hexagonal bins offer a slight improvement in asymptotic MISE when compared to rectangular bins; triangular bins were substantially worse than the other two. Carr et al. (1987) also suggest using hexagonal bins, although from a different perspective. In particular, they show that, when using some sort of glyph representing the frequency for each bin, a rectangular mesh resulted in a type of visual artifact from the vertical and horizontal alignment; the hexagonal bin structure has a much improved visual appeal.

Displaying bivariate histograms can be accomplished in a number of ways. A traditional bivariate histogram with a rectangular mesh can be displayed using a type of 3-D bar chart. Often structure in the histograms of this type can be obscured by the viewing angle. Rotating the display and viewing the histogram at different viewing angles can reveal this hidden structure. In a non-interactive setting, such operations are of course unavailable. An alternative for displaying bivariate histograms is the use of the so-called image plot in which color is used to represent the frequency or density of points in each bin.

An example demonstrating the need and benefit of bivariate histograms is shown in Fig. 5.9. In the left frame, a scatterplot is shown using the average winter midpoint temperature and the log-transformed total precipitation for the Colorado PRISM data. The scatterplot in Fig. 5.9 clearly demonstrates the phenomenon of “too much ink” and little of the structure in the data can be discerned from the plot. In the right

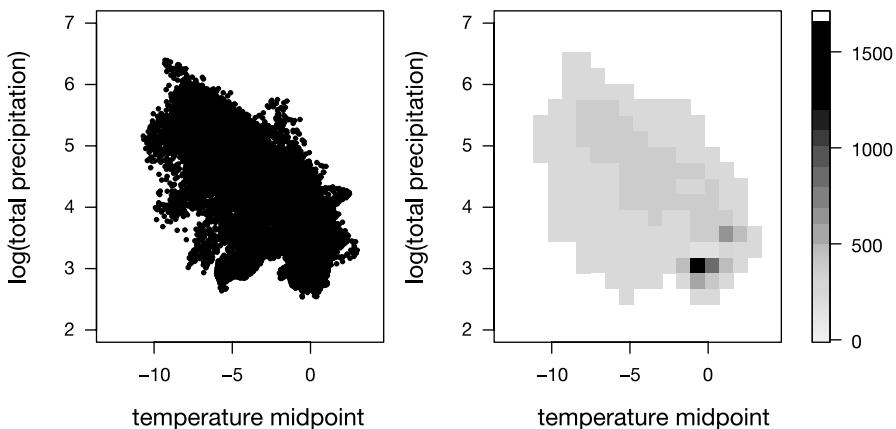


Figure 5.9. The *left frame* shows a scatterplot of the average winter midpoint temperature and the logarithm of the total precipitation for Colorado PRISM data. The *right frame* displays an image plot for a bivariate histogram with the same data

frame, an image plot is used to display a bivariate histogram of the data. The bin-widths for the histogram were determined based loosely on the normal based rule discussed earlier. A greyscale is used to display the density of points in each bin with darker shades indicating a higher density. Far more of the structure in the data appears, as it is clear that there are at least two modes in the data, including one sharp mode represented by the higher temperatures and lower precipitation of the eastern plains and a much more broad mode representing the lower temperatures and higher precipitation in the mountains.

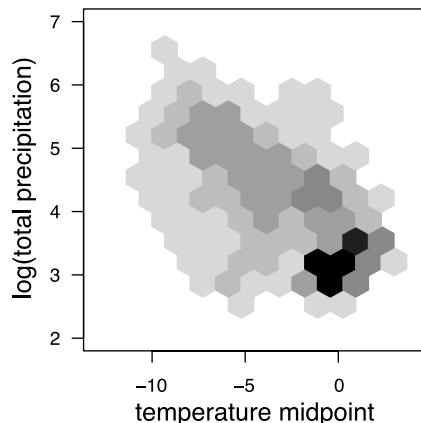


Figure 5.10. A bivariate histogram of the Colorado PRISM data using a hexagonal mesh. Shading is as in the *right panel* of Fig. 5.9

As mentioned earlier, hexagonal binning was introduced as way of improving the visual appeal of bivariate histograms that are displayed using some sort of glyph to represent the frequency in each bin. An alternative is to use greyscale or color coding of the bins to represent frequency. An example using the PRISM data is shown in Fig. 5.10. The number of hexagon bins was chosen to be consistent with the number of bins in the rectangular mesh of the histogram in Fig. 5.9. The hexagons appear to further clarify the multimodal structure in the data.

5.2.2 Bivariate Kernel Density Estimators

Bivariate kernel estimates are more common than bivariate histograms. Basic computation is a simple extension of the univariate method. Choice of the kernel function is a little more complicated but may usually be satisfied with a bivariate kernel generated from one of the standard univariate kernels.

Bandwidth selection also becomes more troublesome, as a 2×2 bandwidth matrix is now required. Product kernels (diagonal bandwidth matrix) that allow for different degrees of smoothing in each dimension are appropriate for most datasets, and often transformations of the original data are used to make the data more amenable to a more simple form of the kernel. Wand and Jones (1993) give an excellent discussion of the issues with parameterizing the bandwidth matrix for bivariate kernel density estimators. The exact form of the bivariate kernel estimator follows from the more general multivariate presentation at the beginning of Sect. 5.3.

Scott (1992) demonstrates that the optimal smoothing parameter for the product kernel estimator is proportional to $n^{-1/6}$. Further, for uncorrelated and normally distributed data, the asymptotic MISE bandwidth is given as $h_k^* = (2/3)^{1/6} \sigma_k n^{-1/6}$ for $k = 1, 2$ and σ_k the standard deviation for the k th variable.

Displaying bivariate density estimates can be accomplished easily using contour plots or via 3-D perspective or wireframe plots. An example is shown in Fig. 5.11, again using the Colorado PRISM data. The left frame shows a contour plot where each contour represents the points of equal height of the density. The sharp mode corresponding to the eastern plains is clearly visible, while there appears to be even further evidence of multimodal structure.

The right plot in Fig. 5.11 shows a perspective or wireframe plot of the estimated density. Note that the density has been rotated in the figure in an attempt to better display the structure in the data. Clearly additional modes are seen, suggesting more structure in the data than simply the rough division of the state into the eastern plains and mountain regions. Of course, densities of this kind (multimodal with differing scales) pose a special challenge to density estimators with fixed meshes or bandwidths. Variable bandwidth methods for kernel estimators have been proposed for bivariate and higher-dimensional data; see Terrell and Scott (1992) and Sain (2002).

Whether to view such plots using 3-D perspective plots or 2-D contour plots is often a matter of personal taste. One view is that the perspective plot is more useful for obtaining an overview of the structure, while a contour plot is more useful for obtaining precise information such as the location of a mode.

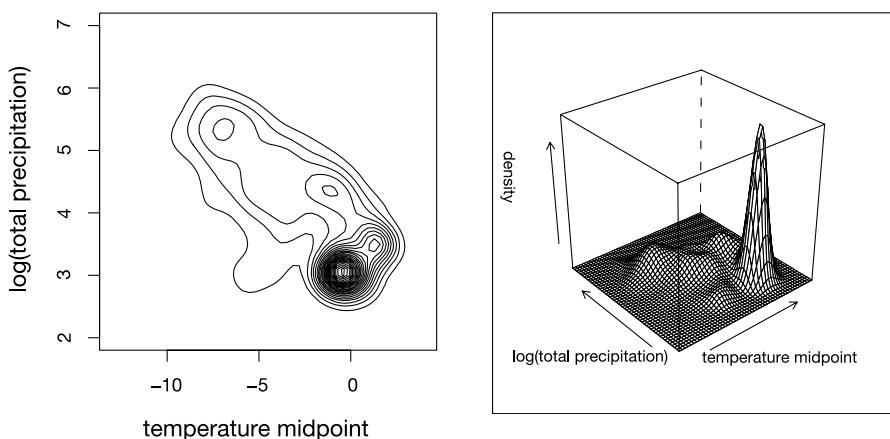


Figure 5.11. Contour (left frame) and perspective (right frame) plots of a bivariate kernel density estimate using the Colorado PRISM data. The perspective plot has been rotated for better visibility of the density structure

An interesting special case of the use of kernel methods and other visualization techniques for bivariate densities arises from the analysis of spatial point patterns. For example, maps of the locations of disease or some other event of interest are produced with the hope of identifying potential clusters in the data. Often, some assumption of uniform spatial randomness is adopted and there are various tests that can be performed to examine this hypothesis. Visualization using kernel density estimates, for example, offers a graphical alternative, in particular, when the number of events is large enough to make a simple map of the locations ineffective (“too much ink”).

An example is shown in Fig. 5.12 in which a bivariate kernel density estimate of the distribution of the locations of sites in Louisiana reporting to the US Environmental Protection Agency’s Toxic Release Inventory (TRI). The TRI is a publicly accessible database that contains information on toxic releases reported by certain manufacturing and industrial sites as well as certain federal facilities.

This density estimate was constructed using the spherically symmetric kernel with a bandwidth that is related to the assumption that the immediate health and other impacts of these TRI sites extend four miles symmetrically around the site. Overlaid on the parish (county) map of Louisiana is a greyscale image plot of the density. Several clusters are clearly seen, in clear violation of any assumption of uniform spatial randomness. In particular, the concentration of sites stands out along the so-called Industrial Corridor, which is loosely defined as the stretch of the Mississippi River between Baton Rouge (located at $30^{\circ} 32'$ N latitude, $91^{\circ} 9'$ W longitude) and New Orleans (at $29^{\circ} 59'$ N, $90^{\circ} 15'$ W).

Comparing bivariate density estimates, even using kernel estimates as in Fig. 5.6, requires some care. Extending the example of Fig. 5.6 comparing Colorado and Kansas PRISM data, Fig. 5.13 shows contour plots of bivariate kernel density estimates of average temperature midpoint and logarithm of the total precipitation. The density

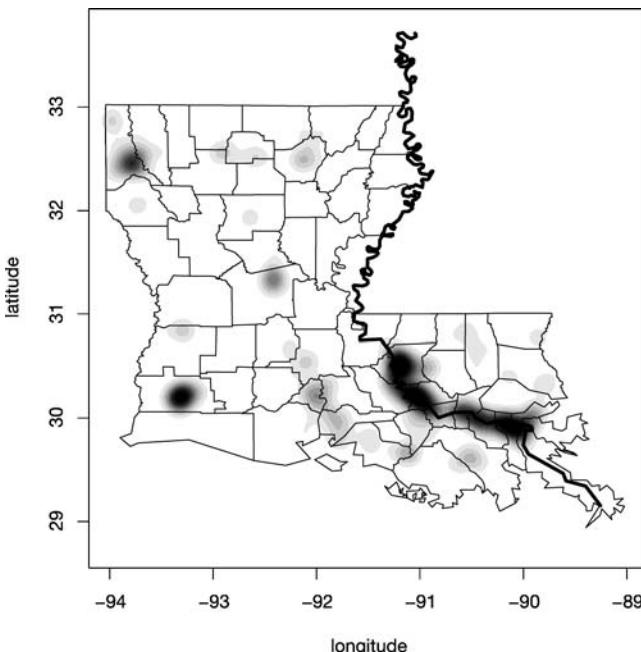


Figure 5.12. Kernel density estimate of the Toxic Release Inventory (TRI) sites in the US state of Louisiana. *Dark regions:* areas of high density. *Thick black line:* path of the Mississippi River

estimate from Colorado is displayed using the black contours while the density estimate for Kansas is displayed using the grey contours. To keep the plot from becoming overly complicated, a small number of specifically chosen contours is used. In particular, three contours are computed for each density and these contours contain roughly 25, 50, and 75 % of the data. Multimodality is clearly present in both estimates, representing different regions of the two states. The much more compact density of the Kansas data is strongly representative of its much greater geographic homogeneity relative to Colorado.

5.3 Higher-dimensional Density Estimates

Three- and 4-D kernel density estimates are quite feasible theoretically. In the multivariate setting, $\mathbf{x} \in \mathfrak{R}^d$, the kernel, $K(\mathbf{t})$, is a multivariate density function often chosen to be a product kernel, defined by

$$K(\mathbf{t}) = \prod_{j=1}^d K_1(t_j). \quad (5.16)$$

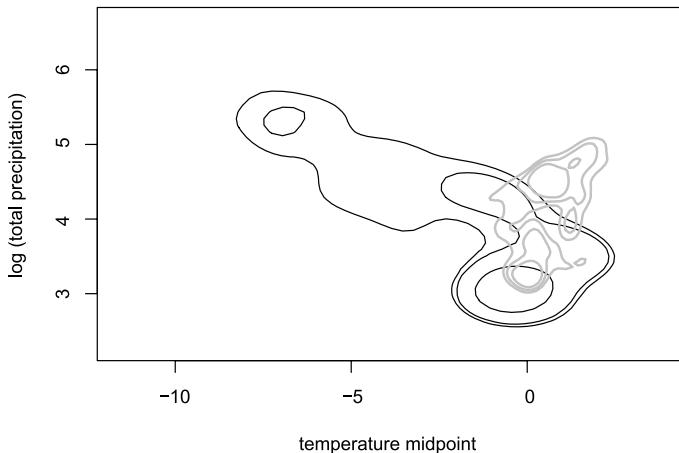


Figure 5.13. Bivariate kernel density estimates of regional climate model output of average temperature and the logarithm of precipitation over a grid of locations in the western USA. *Black contours* represent climate in the state of Colorado while the *grey contours* represent climate in the (more geographically homogenous) state of Kansas

The univariate bandwidth, h , is replaced by an invertible $d \times d$ matrix, H , whose properties are explored in Scott (1992) and Wand and Jones (1993). Then the scaled univariate kernel, $K_h(t)$, is generalized to be

$$K_H(\mathbf{t}) = \frac{1}{|H|} K(H^{-1}\mathbf{t}). \quad (5.17)$$

By a multivariate change of variables, $K_H(\mathbf{t})$ integrates to 1 if $K(\mathbf{t})$ does.

For multivariate data, $\{\mathbf{x}_1, \dots, \mathbf{x}_n\}$, the multivariate kernel estimator is given by

$$\hat{f}(\mathbf{x}) = \frac{1}{n} \sum_{i=1}^n K_H(\mathbf{x} - \mathbf{x}_i). \quad (5.18)$$

The simplest product kernel is the standard multivariate normal

$$K(\mathbf{t}) = (2\pi)^{-d/2} \exp\left(-\frac{1}{2}\mathbf{t}^T \mathbf{t}\right), \quad (5.19)$$

which leads to the kernel estimate

$$\hat{f}(\mathbf{x}) = \frac{1}{n} \sum_{i=1}^n \frac{1}{(2\pi)^{d/2} |H|} \exp\left[-\frac{1}{2} (\mathbf{x} - \mathbf{x}_i)^T H^{-1 T} H^{-1} (\mathbf{x} - \mathbf{x}_i)\right]. \quad (5.20)$$

While the calculations and theory of multivariate density estimation are straightforward, the challenge of identifying and understanding the structure of multivariate

data may be quite difficult. The first step is examining the univariate densities of all the individual variables $\hat{f}(x)$, followed by understanding how pairs of variables covary by the examination of all bivariate density plots, $\hat{f}(x, y)$.

Just as the univariate estimates can only hint at the actual bivariate structure in data, so too bivariate estimates can only hint at the trivariate structure, and so on. Three- and 4-D kernel density estimates are quite feasible theoretically but should be approached with caution for several reasons. The choices of kernel and bandwidth become more complicated, although a product kernel and diagonal bandwidth matrix will often be reasonable. Data becomes thinner at higher dimensions, so larger sample sizes are required to get reasonable results. Most importantly, plotting of the full estimate would require four and five dimensions, respectively, as a perspective plot requires one more dimension than the number of variables.

Clearly, one must set aside any thought of examining the entire function, $\hat{f}(x, y, z)$ or $\hat{f}(x, y, z, t)$. However, note that a bivariate contour plot only requires two dimensions. Likewise, a 3-D density estimate may still be examined as a 3-D contour plot. A single contour slice is a level set of the density, for example,

$$S_\alpha = \{(x, y, z) : \hat{f}(x, y, z) = \alpha \hat{f}_{\max}\}, \quad (5.21)$$

where \hat{f}_{\max} is the largest value of the density and α ranges from 0 to 1. For normal data, the contour is an ellipse (or sphere). When $\alpha = 1$, S_1 is the mode. As α decreases, the size of the ellipse increases.

Of course, a contour plot of a bivariate density is not complete if only one contour level is displayed. Likewise, a trivariate density requires examination of at least three to five levels, depending upon the complexity of the estimated density. This task will require some way to see “through” the individual contours: either transparency, or some method of removing part or all of outer contours so that inner ones may be viewed. Some care should be taken here to carefully distinguish different levels, and possibly upper and lower regions on each side of the contours as well, as otherwise opposite features such as modes and holes may be visually indistinguishable.

When dealing with four or more variables, one’s options are more limited. The best choice appears to be to examine the conditional 3-D estimates as above, for a series of selected “slices” in the fourth variable. For example, a series of values $t_1 < t_2 < \dots < t_k$ are selected and contour shells of the 3-D arrays

$$\hat{f}(x, y, z, t_\ell) \quad \ell = 1, 2, \dots, k \quad (5.22)$$

are displayed. If hardware permits, k may be taken as quite large and the sequence of view may be animated. The animation is effective because the density contours vary smoothly, which the human brain can easily decode. This approach is similar to some of the ideas found in trellis coplots (Cleveland, 1993; Becker, et al., 1996).

To illustrate these ideas, the full PRISM dataset for the continental United States is employed. There are 481 475 grid points, each representing about 6 square miles. The variables used are elevation, precipitation, and maximum temperature for the months December–February averaged over the period 1995–2004. As the variables are quite

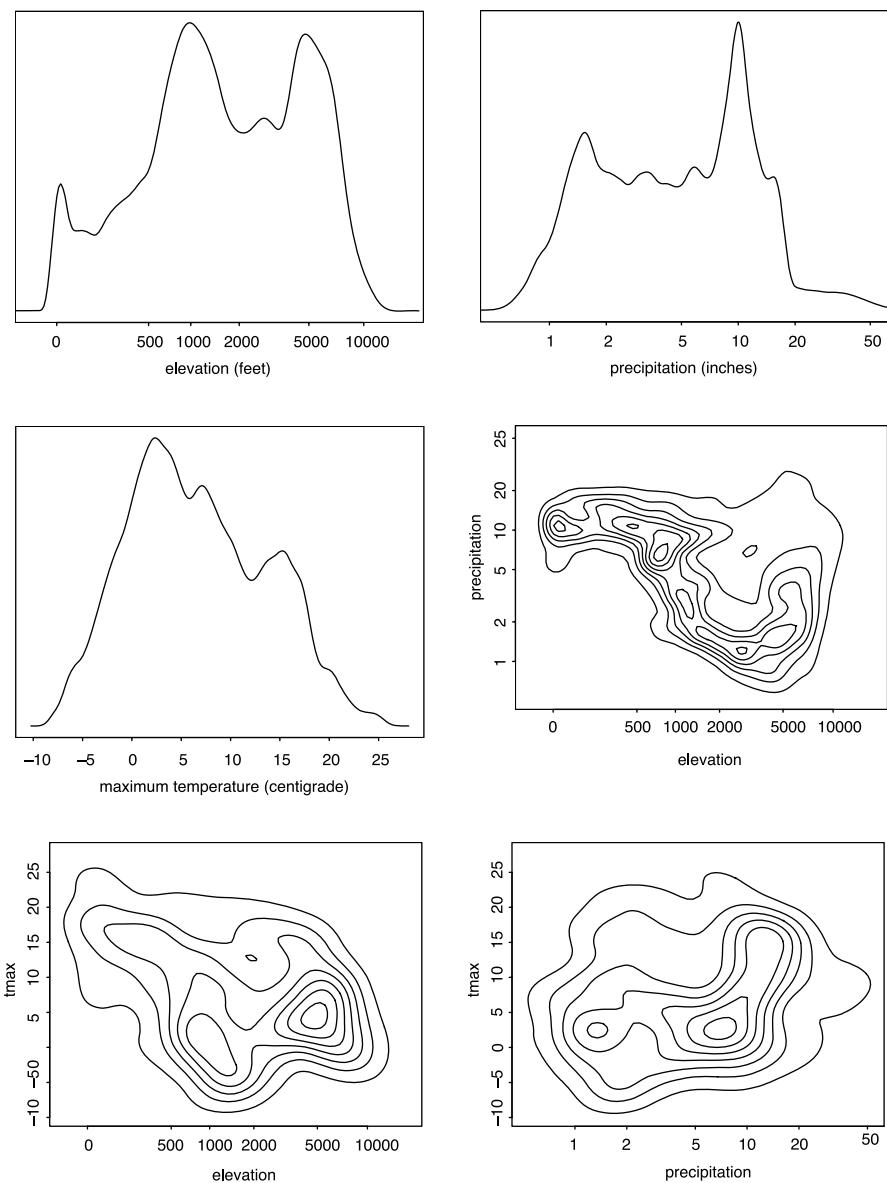


Figure 5.14. Univariate and bivariate density estimates (ASH) of the elevation of the US mainland, precipitation, and maximum temperature; see text for data description and transformations

skewed, the first two variables were reexpressed as $\log_{10}(x + 74)$ and $\log_{10}(y)$, as elevation ranged from -73 to 4005 m, and precipitation from 9 to 3007 mm.

Univariate averaged shifted histograms (as described in Sect. 5.1.2) of these three variables are displayed in the first three panels of Fig. 5.14. For the densities displayed

in Fig. 5.14, the ASH weights were sampled from the quadweight kernel, $K(t) = \frac{315}{256}(1-t^2)_+^4$. Specifically,

$$w_m(j) = m \cdot \frac{K(j/m)}{\sum_{i=-m}^m K(i/m)} \quad j = -m, \dots, 0, \dots, m. \quad (5.23)$$

Each variable was prebinned into 400 intervals. The ASH smoothing parameters were picked by eye to smooth out most of the noisy structure. All of these estimates are multimodal. Regions of high elevation are somewhat unique climatically.

Next, the bivariate ASH density estimates were constructed in the remaining panels of Fig. 5.14. The data were prebinned into 75×75 intervals. A product version of the ASH using the quadweight kernel was selected. Again, the structure is more complex than a single Gaussian and more complex than a mixture of a few Gaussians.

Finally, all three variables were examined simultaneously in one ASH (Fig. 5.15). The data were prebinned into a mesh of size $75 \times 75 \times 75$ and the quadweights again employed. The shell displayed corresponds to $\alpha = 25\%$. The structure is quite fascinating. Of course, given the huge dataset, one has high confidence that the features clearly visible are not due to noise alone.

Without the availability of color in the figure, adding other contour levels is ill advised for such complex contours. Instead, a sequence of slices of the trivariate ASH may be displayed (Fig. 5.16). The climate is less complex near sea level and high altitudes. It would be interesting to link features in these frames with geographical information. Such research has been shown to be fruitful (Whittaker and Scott, 1999).

Without color, attempts to show 4-D and 5-D densities are best postponed. However, it should be clear from these examples the potential power that such displays may bring. These figures bring both an overview and yet great detail. Given the increasing number of massive datasets available for analysis, these tools can prove highly effective in the hands of a trained analyst.

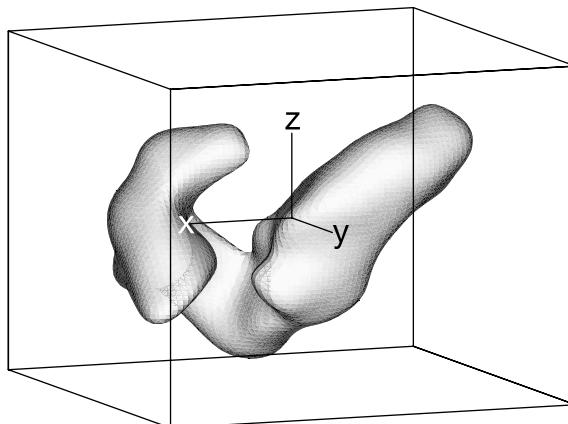


Figure 5.15. Trivariate density estimate (ASH) of elevation (x), precipitation (y), and maximum temperature (z)

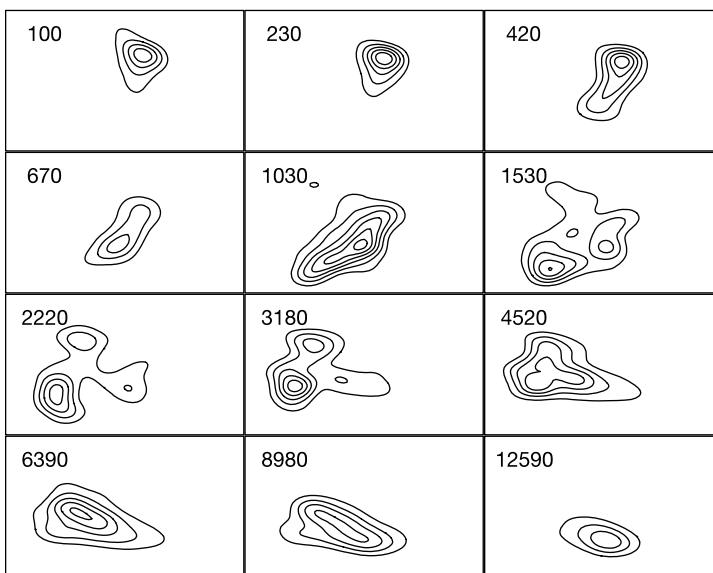


Figure 5.16. Bivariate slices of temperature (y) vs. precipitation (x) of the trivariate density estimate (ASH) at a sequence of levels of the elevation (z)

References

- Abramson, I.S. (1982). On bandwidth variation in kernel estimates – a square root law. *Ann Stat* 10:1217–1223
- Ahrens, L.H. (1965). Observations on the Fe-Si-Mg relationship in chondrites. *Geochimica et Cosmochimica Acta* 29:801–806
- Bartlett, M.S. (1963). Statistical estimation of density functions. *Sankhyā Ser A* 25:245–254
- Becker, R.A., Cleveland, W.S. and Shyu, M.J. (1996). The design and control of Trellis display. *J Comput Graph Stat* 5:123–155
- Carr, D.B., Littlefield, R.J., Nicholson, W.L. and Littlefield, J.S. (1987). Scatterplot matrix techniques for large N . *J Am Stat Assoc* 82:424–436
- Chaudhuri, P. and Marron, J.S. (1999). SiZer for exploration of structures in curves. *J Am Stat Assoc* 94:807–823
- Cleveland, W.S. (1993). *Visualizing Data*. Hobart, Summit NJ
- Cleveland, W.S. and McGill, R. (1984). The many faces of a scatterplot. *J Am Stat Assoc* 79:807–822
- Daly, C., Neilson, R.P. and Phillips, D.L. (1994). A statistical-topographic model for mapping climatological precipitation over mountainous terrain. *J Appl Meteorol* 33:140–158
- Freedman, D. and Diaconis, P. (1981). On the histogram as a density estimator: L_2 theory. *Zeitschrift für Wahrscheinlichkeitstheorie und verwandte Gebiete*. 57:453–476

- Friendly, M. and Dennis, D. (2005). The early origins and development of the scatterplot. *J Hist Behav Sci* 41:103–130
- Gibson, W., Daly, C. and Taylor, G. (1997). Derivation of facet grids for use with the PRISM model. In: *Proceedings of the 10th American Meteorological Society conference on applied climatology*. pp 208–209
- Hall, P. and Marron, J.S. (1988). Variable window width kernel estimates of probability densities. *Probabil Theory Related Fields* 80:37–49
- Hall, P. and Minnotte, M.C. (2002). High-order data sharpening for density estimation. *J R Stat Soc Ser B* 64:141–157
- Hall, P., Sheather, S.J., Jones, M.C. and Marron, J.S. (1991). On optimal data-based bandwidth selection in kernel density estimation. *Biometrika* 78:263–269
- Jones, M.C., Samiuddin, M., Al-Harbe, A.H. and Maatouk, T.A.H. (1998). The edge frequency polygon. *Biometrika* 85:235–239
- Jones, M.C. and Signorini, D.F. (1997). A comparison of higher-order bias kernel density estimators. *J Am Stat Assoc* 92:1063–1073
- Marchette, D.J., Priebe, C.E., Rogers, G.W. and Solka, J.L. (1996). Filtered kernel density estimation. *Comput Stat* 11:95–112
- Marchette, D.J. and Wegman, E.J. (1997). The filtered mode tree. *J Comput Graph Stat* 6:143–159
- Minnotte, M.C. (1996). The bias-optimized frequency polygon. *Comput Stat* 11:35–48
- Minnotte, M.C. (1998). Achieving higher-order convergence rates for density estimation with binned data. *J Am Stat Assoc* 93:663–672
- Minnotte, M.C. and Scott, D.W. (1993). The mode tree: a tool for visualization of nonparametric density features. *J Comput Graph Stat* 2:51–68
- Minnotte, M.C., Marchette, D.J. and Wegman, E.J. (1998). The bumpy road to the mode forest. *J Comput Graph Stat* 7:239–251
- Sain, S.R. (2002). Multivariate locally adaptive density estimation. *Comput Stat Data Anal* 39:165–186
- Sain, S.R. and Scott, D.W. (1996). On locally adaptive density estimation. *J Am Stat Assoc* 91:1525–1534
- Sain, S.R., Baggerly, K.A. and Scott, D.W. (1994). Cross-validation of multivariate densities. *J Am Stat Assoc* 89:807–817
- Samiuddin, M. and El-Sayyad, G.M. (1990). On nonparametric kernel density estimates. *Biometrika* 77:865–874
- Scott, D.W. (1979). On optimal and data-based histograms. *Biometrika* 66:605–610
- Scott, D.W. (1985). Averaged shifted histograms: effective nonparametric density estimators in several dimensions. *Ann Stat* 13:1024–1040
- Scott, D.W. (1988). A note on choice of bivariate histogram bin shape. *J Offic Stat* 4:47–51
- Scott, D.W. (1992). *Multivariate Density Estimation: Theory, Practice, and Visualization*. Wiley, New York
- Scott, D.W. and Terrell, G.R. (1987). Biased and unbiased cross-validation in density estimation. *J Am Stat Assoc* 82:1131–1146
- Sheather, S.J. and Jones, M.C. (1991). A reliable data-based bandwidth selection method for kernel density estimation. *J R Stat Soc Ser B* 53:683–690

- Silverman, B.W. (1981). Using kernel density estimates to investigate multimodality. *J R Stat Soc Ser B* 43:97–99
- Sturges, H.A. (1926). The choice of a class interval. *J Am Stat Assoc* 21:65–66
- Terrell, G.R. and Scott, D.W. (1992). Variable kernel density estimation. *Ann Stat* 24:1236–1265
- Tufte, E.R. (1983). *The Visual Display of Quantitative Information*. Graphics Press, Cheshire, CT
- Wand, M.P. and Jones, M.C. (1993). Comparison of smoothing parameterizations in bivariate kernel density estimation. *J Am Stat Assoc* 88: 520–528
- Whittaker, G. and Scott, D.W. (1999). Nonparametric regression for analysis of complex surveys and geographic visualization. *Sankhyā Ser B* 61:202–227

Structured Sets of Graphs

III.6

Richard M. Heiberger, Burt Holland

6.1	<i>Introduction</i>	417
6.2	<i>Cartesian Products and the Trellis Paradigm</i>	417
	Trellis Paradigm	418
	Implementation of Trellis Graphics	418
6.3	<i>Scatterplot Matrices: splom and xysplom</i>	419
	Example – Life Expectancy	419
	Display of Scatterplot Matrix	420
	Example – A Scatterplot Matrix with Conditioning.....	422
	Coordinating Sets of Related Graphs.....	422
	Summary Plot with Legend	425
	Example – an xysplom with Labeled Correlation Coefficients.....	426
	Ladder of Powers Plot – Wool Data	427
6.4	<i>Regression Diagnostic Plots</i>	429
	Case Statistics	429
	Example – Kidney Data	429
6.5	<i>Analysis of Covariance Plots</i>	431
	Example – Hot Dog Data	432
	Cartesian Product of Model Parameters	433
6.6	<i>Interaction Plots</i>	434
	Two-factor Rhizobium Example	434
	Extended Two-way Interaction Plot	434
	Three-factor Vulcanized Rubber Example.....	435
	Design Issues for the Two-way Interaction Plot	437
	Two-way Interaction Plots with Simple Effects	437

6.7	<i>Boxplots</i>	439
	Assessing Three-way Interaction.....	439
	Sequences of Boxplots	440
	Microplots.....	441
	Example – Catalyst Data	441
	Example – Muscle Data, continued	442
6.8	<i>Graphical Display of Incidence and Relative Risk</i>	442
6.9	<i>Summary</i>	444
6.10	<i>File Name Conventions</i>	444

We present many examples of structured sets of graphs that convey and support statistical analyses. Structured sets of graphs can be drawn with any modern statistical software system with graphics capabilities. We use S-PLUS and R, two dialects of the S language that offer substantial capabilities for producing graphs customized to the particular needs and visions of the analyst. We emphasize two basic paradigms for constructing structured graphs: Cartesian products and the Trellis paradigm. Our software for all examples in this article is available from Heiberger and Holland (2004).

Introduction

6.1

S-PLUS and R offer users substantial capabilities to customize graphs to their particular needs and visions when they are accessed using command language rather than their graphical user interfaces (GUIs). Production software, that is, software already developed by someone else, needs to be standardized, packaged, and restrictive, allowing the user less control. Analysts occasionally require a graph unlike any readily available elsewhere. We recommend that serious data analysts invest time in becoming proficient in writing code rather than using GUIs. Users of a GUI are limited to the current capabilities of the GUI. While the design of GUIs will continually improve, their capabilities will always remain far behind what skilled programmers can produce. Even less-skilled analysts can take advantage of cutting-edge graphics by accessing libraries of graphing functions such as those accompanying our text or available at Statlib and elsewhere on the Internet.

Our graphical displays are designed for elementary to intermediate statistical analyses, but the graphs themselves are relatively sophisticated constructions. Our examples extend the concept of a structured presentation of plots of different sets of variables, or of different parametric transformations of the same set of variables. Several of the examples extend the interpretation of the model formula, that is, the semantics of the formula, to allow easier exposition of standard statistical techniques.

Our examples are taken from several sources. These include classical and recent examples from the statistical literature, standard texts in statistics, the authors' textbook Heiberger and Holland (2004) (referred to in the sequel as HH), and projects on which the authors have worked as consulting statisticians.

Cartesian Products and the Trellis Paradigm

6.2

A feature common to many of the displays is the Cartesian product principle behind their construction.

The Cartesian product of two sets A and B is the set consisting of all possible ordered pairs (a, b) , where a is a member of set A and b is a member of set B . Many of our innovative graphs are formed as a rectangular set of panels, or subgraphs, where each panel is based on one pair from a Cartesian product. The sets defining the Cartesian product differ for each graph type. For example, a set can be a collection of variables, functions of a single variable, levels of a single factor, functions of a fitted model, different models, etc.

When constructing a graph that can be envisioned as a Cartesian product, it is necessary that the code writer be aware of the Cartesian product relationship. The code for such a graph includes a command that explicitly states the Cartesian product.

6.2.1

Trellis Paradigm

Many of the graphs in this article are constructed using the trellis paradigm pioneered by S-PLUS. The trellis system of graphics is based on the paradigm of repeating the same graphical specifications for each element in a Cartesian product of levels of one or more factors.

The majority of the methods supplied in the S-PLUS `trellis` library are based on a typical formula having the structure

$$y \sim x \mid a * b \quad (6.1)$$

where

y is either continuous or a factor

x is continuous

a is a factor

b is a factor

and each panel is a plot of $y \sim x$ for the subset of the data defined by the Cartesian product of the levels of a and b .

6.2.2

Implementation of Trellis Graphics

The concept of trellis plots can be implemented in any graphics system. In the S family of languages (S-PLUS and R), selection of the set of panels, assignment of individual observations to one panel in the set, and coordinated scaling across all panels is automated in response to a formula specification at the user level. In other languages that are capable of displaying multiple panels on the same physical page, the user (at this writing) is responsible for those tasks.

The term “trellis” comes from gardening, where it describes an open structure used as a support for vines. In graphics, a trellis provides a framework in which related graphs can be placed.

Scatterplot Matrices: `splom` and `xysplom`

6.3

A scatterplot matrix (`splom`) is a trellis display in which the panels are defined by a Cartesian product of variables. In the standard scatterplot matrix constructed by `splom`, the same set of variables defines both the rows and columns of the matrix. More generally, what we term an `xysplom` uses different sets of variables defining the rows and columns of the matrix. We strongly recommend the use of `sploms` and `xysploms`, sometimes conditioned on values of relevant categorical variables, as initial steps in analyzing a set of data. Examples are given in Sects. 6.3.1–6.3.6.

An `xysplom`, produced with our function `xysplom` (Heiberger and Holland, 2004), is used to produce a rectangular subset, often an off-diagonal block, of a scatterplot matrix. It involves a Cartesian product of the form $[\text{variables}] \times [\text{variables}]$, where the two sets of variables contain no common elements. A large `splom` may be legibly presented as a succession of smaller `sploms` (diagonal blocks of the large `splom`) and `xysploms` (off-diagonal blocks of the large `splom`).

An example where `xysploms` are useful in their own right is when examining a set of potential response variables against members of a set of potential explanatory variables.

We use an extension

$$u + v \sim w + x + y + z \mid a * b \quad (6.2)$$

of the syntax of the standard model formula to define the variables of the `xysplom` function. The rows of the `xysplom` are defined by the crossing of the set of variables on the left-hand side of the formula with the set of variables on the right-hand side of the formula. The expanded `xysplom` generated with Eq. (6.2) will contain, for each combination of the elements in `a` and `b`, an `xysplom` having two rows and four columns.

Example – Life Expectancy

6.3.1

For each of the 40 largest countries in the world (according to 1990 population figures), data are given for a country's life expectancy at birth categorized by gender, number of people per television set, and number of people per physician. This is a subset of the full data set contained in a study cited by Rossman (1994) that sought a short list of variables that could accurately predict life expectancy:

```
life.exp: Life expectancy at birth
ppl.per.tv: Number of people per television set
ppl.per.phys: Number of people per physician
```

Figure 6.1 is a scatterplot matrix for a final linear model for these data. The variables `ppl.per.tv` and `ppl.per.phys` were log-transformed to correct for positive skewness in the original data. This figure demonstrates that the response `life.exp` is moderately negatively correlated with both explanatory variables, and the two explanatory variables are positively associated.

6.3.2

Display of Scatterplot Matrix

Figure 6.1 has several noteworthy characteristics. The panels are symmetric about the main diagonal running from SW to NE, which, as explained below, is a more appealing choice than NW to SE. The panels are square, which makes sense because both dimensions contain the same variables. We display both $y \sim x$ and $x \sim y$ since we don't know in advance which is more helpful or appropriate. The panels on the main diagonal are used for labeling and tick marks.

Figure 6.2 contains an alternate splom orientation that we do not recommend. In the alternate orientation, with the downhill diagonal and rectangular panels, each variable is depicted on two different scales, making comparisons of each panel with its transpose more difficult than with an uphill diagonal and square panels.

Figure 6.3 compares the axes of symmetry of figures resembling Figs. 6.1 and 6.2. Figure 6.3a has six axes of symmetry. We focus on panel 6, which appears in positions reflected about the main NW–SE axis. The individual points within panels 6 and 6' are reflected about the dashed SW–NE line, as indicated by the position of the arrow. The other four axes, which reflect respectively panel 5, panels 3 and 4, panel 2, and panel 1, are indicated with dotted lines. Figure 6.3b has only one axis of symmetry. The arrow for panel 6 is reflected by the same SW–NE axis that reflects panels 6 and 6'.

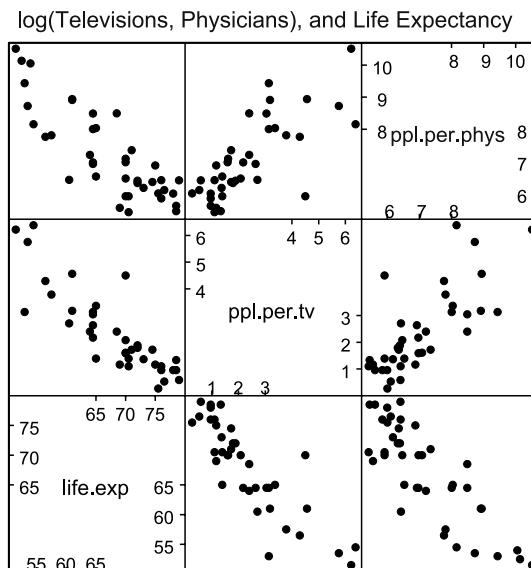


Figure 6.1. $\log(\text{televisions})$, $\log(\text{physicians})$, and life expectancy (File: hh/grap/code/grap.f6.s)

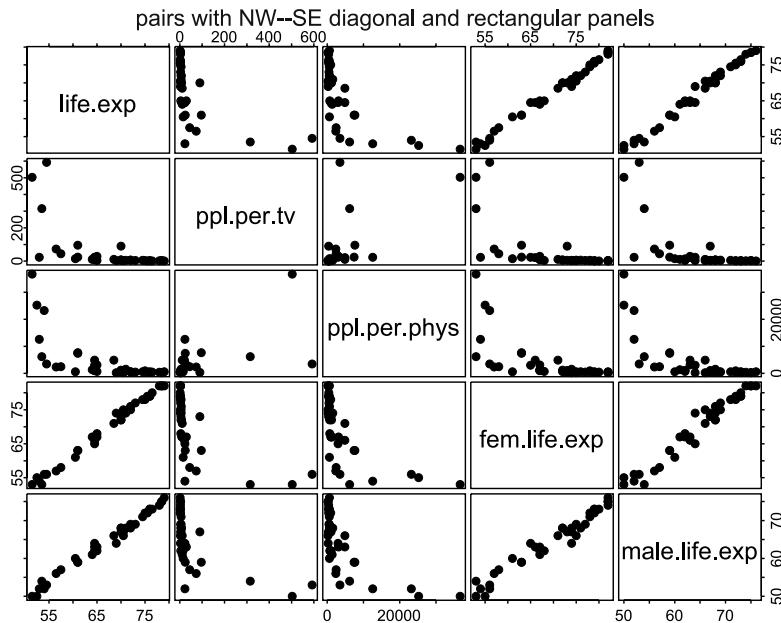


Figure 6.2. Alternate orientation with rectangular panels for splom. **We do not recommend this orientation.** The downhill diagonal is harder to read (Fig. 6.3). The rectangular panels make it hard to compare each panel with its transpose. The location of comparable panels (`life.exp ~ ppl.per.tv` and `ppl.per.tv ~ life.exp`, for example) reflect on a common NW–SE axis; the content of the comparable panels reflects on a unique SW–NE axis (File: `hh/grap/code/grap.f11.s`)

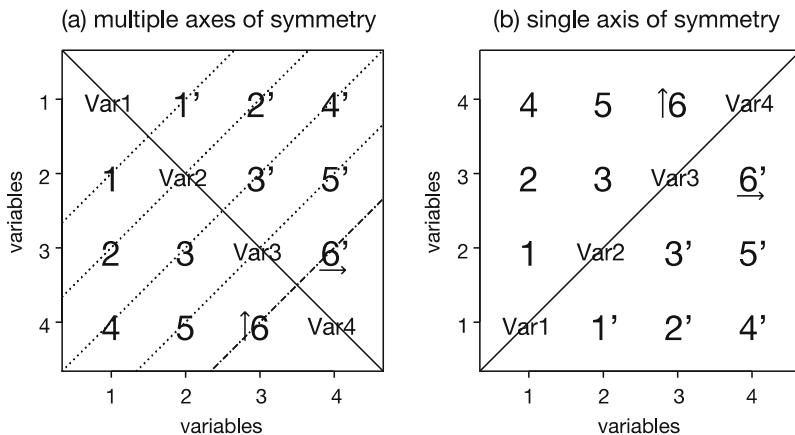


Figure 6.3. Axes of symmetry for splom. The *left panel*, with multiple axes of symmetry, is very difficult to read because the sets of panels with the same variable are reflected on the common NW–SE axis and the variables within each pair of panels are reflected on their individual SW–NE axes (File: `hh/grap/code/grap.f12.s`)

6.3.3

Example – A Scatterplot Matrix with Conditioning

The goal of a chiropractic research project (Harrison et al., 2002) was to model pain indices constructed from patient pain questionnaires as functions of skeletal measurements summarizing patient posture:

SBA: sacral base angle

API: angle of pelvic incidence

PTPIA: posterior tangent pelvic incidence angle

Sex: female, male

Group: pain category: normal, chronic, acute

Associations between these two classes of variables could suggest chiropractic skeletal adjustments to address pain. We illustrate with a subset of the data set that includes three continuous skeletal measurements for both sexes and three pain categories from a sample of 150 subjects. Figure 6.4 was prepared as an initial look at these data, not as a presentation of an interim or final analysis.

Figure 6.4 exemplifies what is meant by a *structured set of graphs*. It systematically unifies 36 interrelated graphs defined by the Cartesian product of several sets. The figure consists of a 2×3 arrangement of scatterplot matrices (sploms). The two columns are defined by sex of the patient and the three rows by pain category. Within each of the sploms, the nine panels are defined by the Cartesian product of the set of three continuous variables (SBA, API, PTPIA) crossed with itself. The upper triangle of each splom contains the mirror image of the set of plots in the lower triangle.

Evidently these skeletal measurements do not differ by sex but do differ according to pain category. The measurements are more tightly clustered for subjects classified as pain-free (normal) than for those having acute or chronic pain. In addition, we see that measurements API and SBA are more highly correlated for pain subjects than those without pain. To ease the reader's task in seeing both the tightness and the correlation, we collect in Fig. 6.5 all the SBA ~ API panels from Fig. 6.4 and also show the marginal distributions for the Sex, Group, and Total.

6.3.4

Coordinating Sets of Related Graphs

The graphical issues that needed attention in Fig. 6.5 are

Positioning: the panels containing marginal displays need to be clearly delineated as distinct from the panels containing data from just a single set of levels of the factors. We do this by placing extra space between the set of panels for the individual factor values and the panels containing marginal displays.

Scaling: all panels need to be on exactly the same scale to enhance the reader's ability to compare the panels visually. We use the automatic scaling feature of trellis plots to scale simultaneously both the individual panels and the marginal panels.

Labeling: we indicate the marginal panels by use of the strip labels. Each panel label is constructed from the levels of the two factors defining it. The marginal panels contain a level name from just one factor. The Total panel is named without reference to the factor level names.

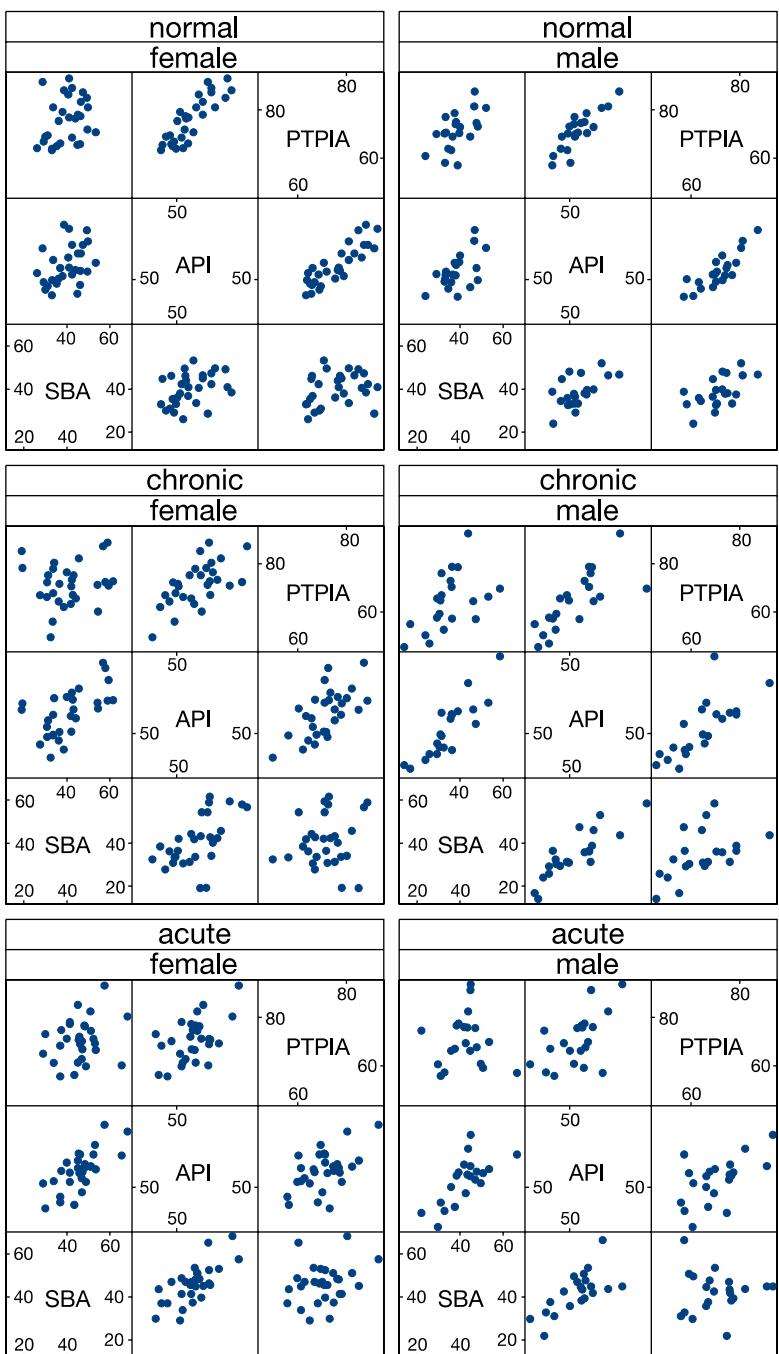


Figure 6.4. Skeletal measurements by sex and pain category (File: hh/csc/code/lumb.s)

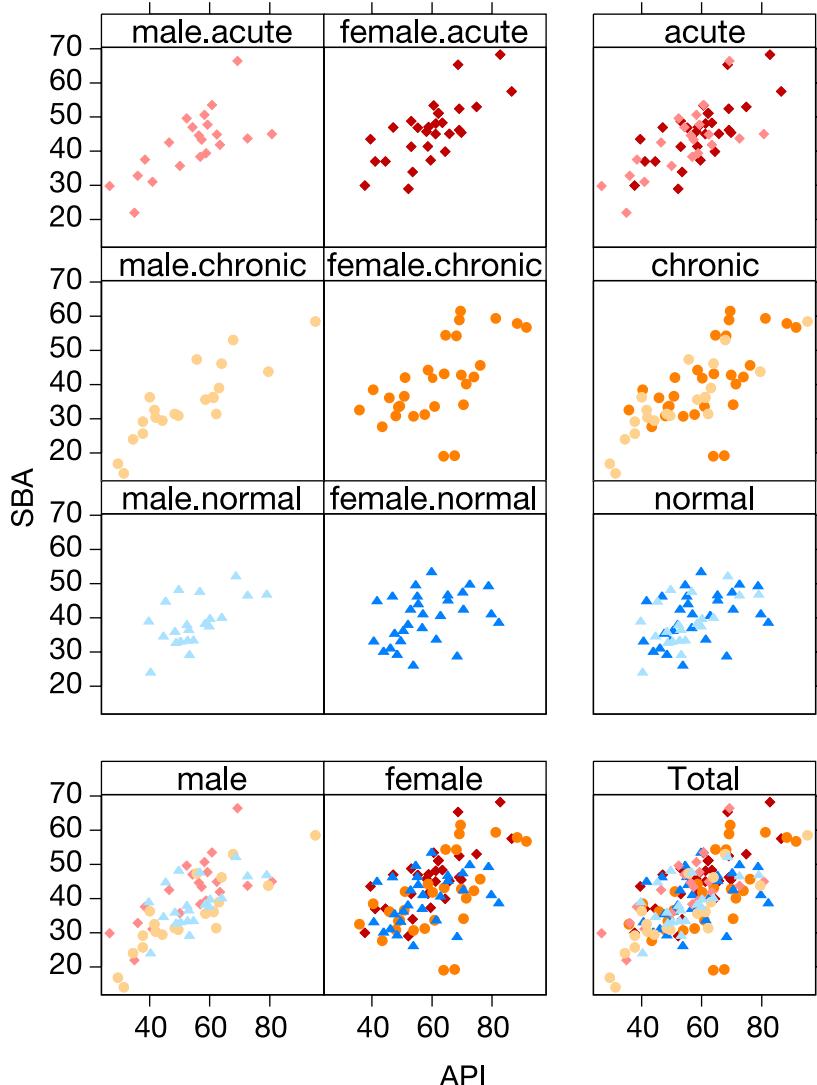


Figure 6.5. The SBA ~ API panel from Fig. 6.4 is expanded and shown with marginal plots for each of the two grouping variables: Sex and Group. The Sex margin at the *bottom* of the figure shows not much difference between male and female. The Group margin on the *right-hand* side of the figure shows that both variables, API and SBA, have a much narrower range in the pain-free normal group than in either of the two pain groups. There is a suggestion of an interaction in the female.chronic panel, where we see a larger range for SBA for values of API near 60 and a correspondingly lower correlation than in the male.chronic panel. In this figure, the colors and symbols are implicitly defined by the labels of the panels. The colors and symbols are explicitly defined in the key in Fig. 6.6 in which we expand the *lower right* Total panel of Fig. 6.5 (File: hh/csc/code/lumb-i.s)

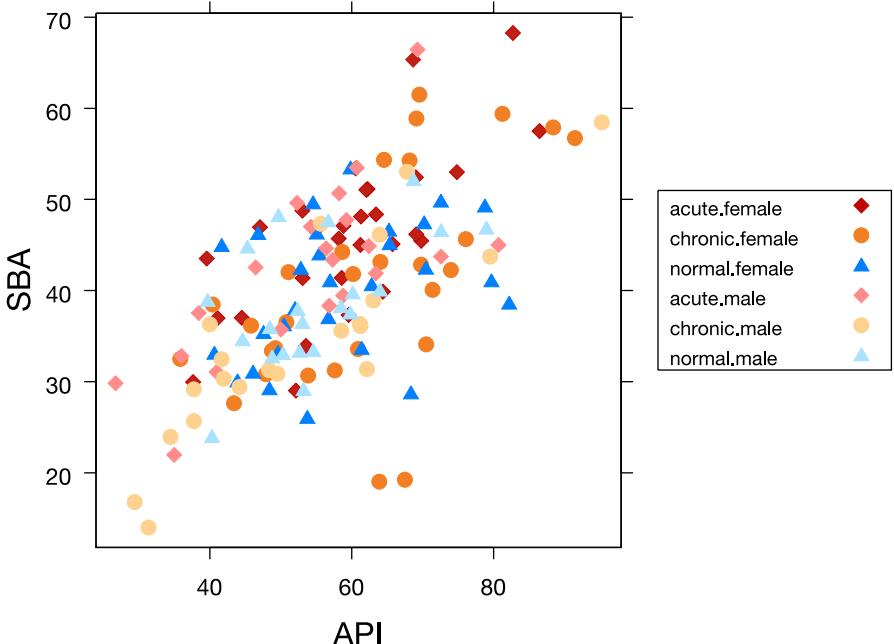


Figure 6.6. The Total panel (*lower right*) of SBA ~ API panel from Fig. 6.5 is expanded and the key defining the symbols and colors is displayed. Now that we have seen the detail of Fig. 6.5, and we are able to see most of the features in this single panel (File: hh/csc/code/lumb.s)

Color and shape of plotting characters: we used three contrasting colors for the three-level factor. We also used three distinct plotting characters for the three-level factor. This is both redundant – reemphasizing the difference between levels – and defensive – protecting the interpretability of the graph from black-and-white copying by a reader. We used lighter and darker shades of the same color to distinguish the two-level factor Sex. The different darknesses will usually survive photocopying into black and white.

The ColorBrewer Web site Brewer (2002) gives a discussion on the principles of color choice and gives a series of palettes for distinguishing nominal sets of items or sequences of items.

Summary Plot with Legend

6.3.5

Figure 6.6 is an expansion of the grand total panel of Fig. 6.5 with a legend contrasting the combinations of pain group and sex. The legend is needed in Fig. 6.6 because it does not have the implicit definitions of the individual panels. We chose to label the legend with level names constructed as the Cartesian product of the names of the levels of the two defining factors.

6.3.6

Example – an **xysplom** with Labeled Correlation Coefficients

Figure 6.7 is taken from a study Harrison et al. (2006) of scoliosis, abnormal lateral curvature of the spine. An initial stage of the study required associating each of two response variables (`DispL` and `DispR`, measures of translation displacement) with its own set of four potential explanatory variables. All explanatory variables are angles between lines drawn through projected centers of mass on AP (front–back) radiographs of thoracic and lumbar vertebrae. Variable names including the letter “L” are measurements on the left side of the subject’s body and the names including the letter “R” are measurements on the right side of the body. The figure shows moderate correlations between most of the pairs of variables.

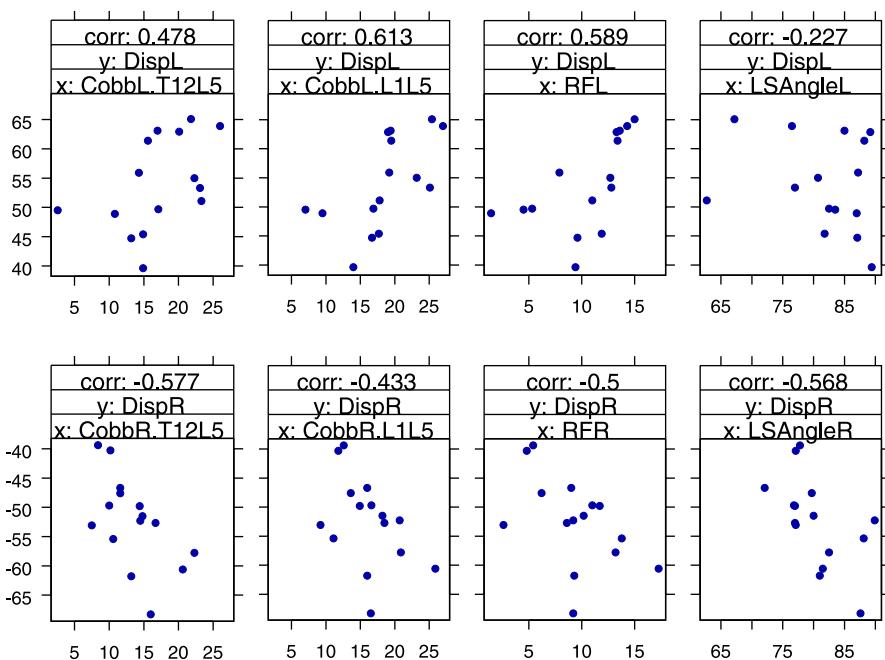


Figure 6.7. An xysplom of two response variables, each with four associated explanatory variables. Pairwise correlations are shown in the *strip labels*. The separate *y*-scales for each response variable were forced to show a common number of data units per inch of graph. Each pair of *x*-scales, for Left and Right measurements of similar quantities, has a common scale. All four panels on the *left*, those containing an *x*-variable name that includes the string “Cobb,” have the same scale (File: h2/splm/code/scolio2.s)

The mechanics of scale control in Fig. 6.7 differ for S-Plus and R. In S-Plus, we needed to draw eight graphs and give each one its own `xlim` and `ylim` arguments. In R, the `xlim` and `ylim` arguments can take a list and therefore all eight panels on the top can be drawn with a single call to the `HH` function `xysplom`.

The `xysplom` has an option to display pairwise correlations in the strip labels. We use that capability in Fig. 6.7.

Ladder of Powers Plot – Wool Data

6.3.7

A power transformation often succeeds in changing from a skewed distribution to one more closely resembling a symmetric unimodal distribution. Such transformations are basic items in analysts' toolkits because many standard statistical analyses require variables to be approximately normally distributed.

The family of power transformations $T_p(x)$, often called *Box–Cox transformations* Box and Cox (1964), is given by

$$T_p(x) = \begin{cases} x^p & (p > 0), \\ \ln(x) & (p = 0), \\ -x^p & (p < 0). \end{cases} \quad (6.3)$$

The *ladder of powers* is the sequential set of power transformations with the particular choice of powers $p = -1, -\frac{1}{2}, 0, \frac{1}{2}, 1, 2$.

In a study of the “plasticity of wool,” Ripa and Speakman (1951) reprinted in Tukey (1977), interest lies in relating `diam`, the coefficient of variation of the diameter of a wool fiber, to the amount of `time` in minutes the fiber is stretched under a prescribed load. Figure 6.8 displays an `xysplom` of the set of ladder of powers transformations of `diam` against the ladder of powers transformations of `time`. Suppose the goal is to model some transformation of a variable y as a linear function of some transformation of x . This is best accomplished if we can find a transformation of each variable such that the two transformed variables are closely linearly related. We examine this plot for the pair of transformations that best renders a linear relationship. Clearly the best transformation for `time` is the square root. Careful examination of the figure suggests that leaving `diam` untransformed is slightly better than using a square root transformation for `diam`. At this stage of the analysis, we would ordinarily ask the client which of these two transformations is more readily interpretable.

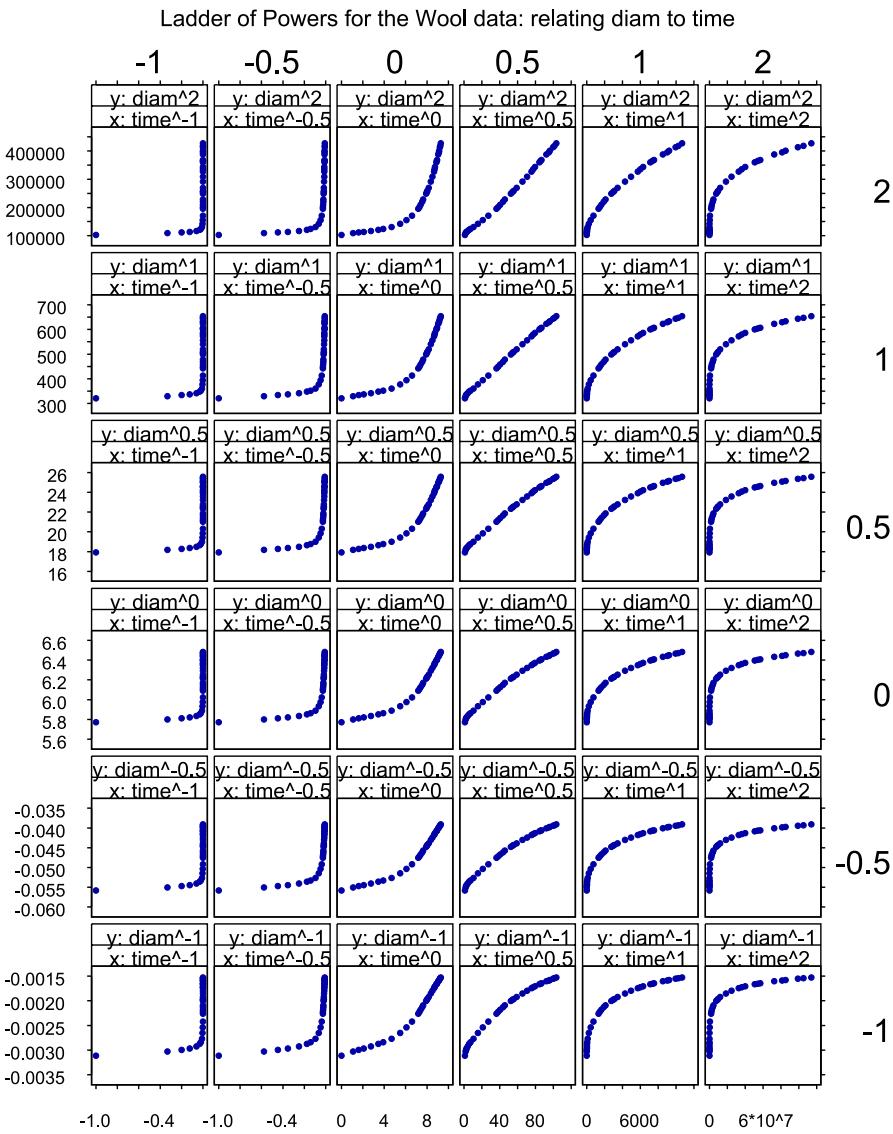


Figure 6.8. Ladder of powers plot for wool data: relating diam to time. Data source: Ripa and Speakman (1951), reprinted in Tukey (1977) (File: h2/splm/code/wools.s)

Regression Diagnostic Plots

6.4

After developing an initial reasonably fitting linear regression model, an analyst may wish to assess whether any of the observations unusually impact on the quality of the fit.

Case Statistics

6.4.1

A strategy for doing so is to examine various *case statistics* that have a value for each of the n cases in the data set. Belsley et al. (1980) presents detailed discussions of case statistics including definitions, formulas, interpretation, and suggested thresholds for flagging a case as unusual. If a case statistic has a value that is unusual, based on thresholds developed in the literature, the analyst should *scrutinize* the case. One action the analyst might take is to delete the case. This is justified if the analyst determines the case is not a member of the same population as the other cases in the data set. But deletion is just one possibility. Another is to determine that the flagged case is unusual in ways apart from those available in its information in the present data set, and this may suggest a need for additional predictors in the model.

We focus on five distinct case statistics, each having a different function and interpretation. (One of these, DFBETAS, is a vector with a distinct value for each regression coefficient including the intercept coefficient.) For small data sets the analyst may choose to display each statistic for all cases. For larger data sets we suggest that the analyst display only those values of the case statistics that exceed a threshold, or flag, indicating that the case is unusual in some way.

A *regression diagnostic plot* displays all commonly used case statistics on a single page. Included are thresholds for flagging cases as unusual along with identification of such cases. Heretofore, presentations of regression diagnostics have been presented on multiple pages of tabular output with one row per observation. It is very difficult to read such tables and examine them for cases that exceed accepted thresholds.

Example – Kidney Data

6.4.2

Creatine clearance is an important but difficult to measure indicator of kidney function. Shih and Weisberg (1986), also presented in Neter et al. (1996), discuss the modeling of clearance as a function of the more readily measured variables serum clearance concentration, age, and weight. The datafile is (hh/datasets/kidney.dat).

At an intermediate stage of analysis of these data, the researchers posed the linear regression model

```
clearance ~ concent + age + weight + concent * age
```

Figure 6.9 is a regression diagnostic plot for this model. It flags five cases as being unusual. Case 16 has a high leverage value implying an unusual set of predictors. The

lm(formula = clearance ~ concent + age + weight + concent * age, data = kidney)

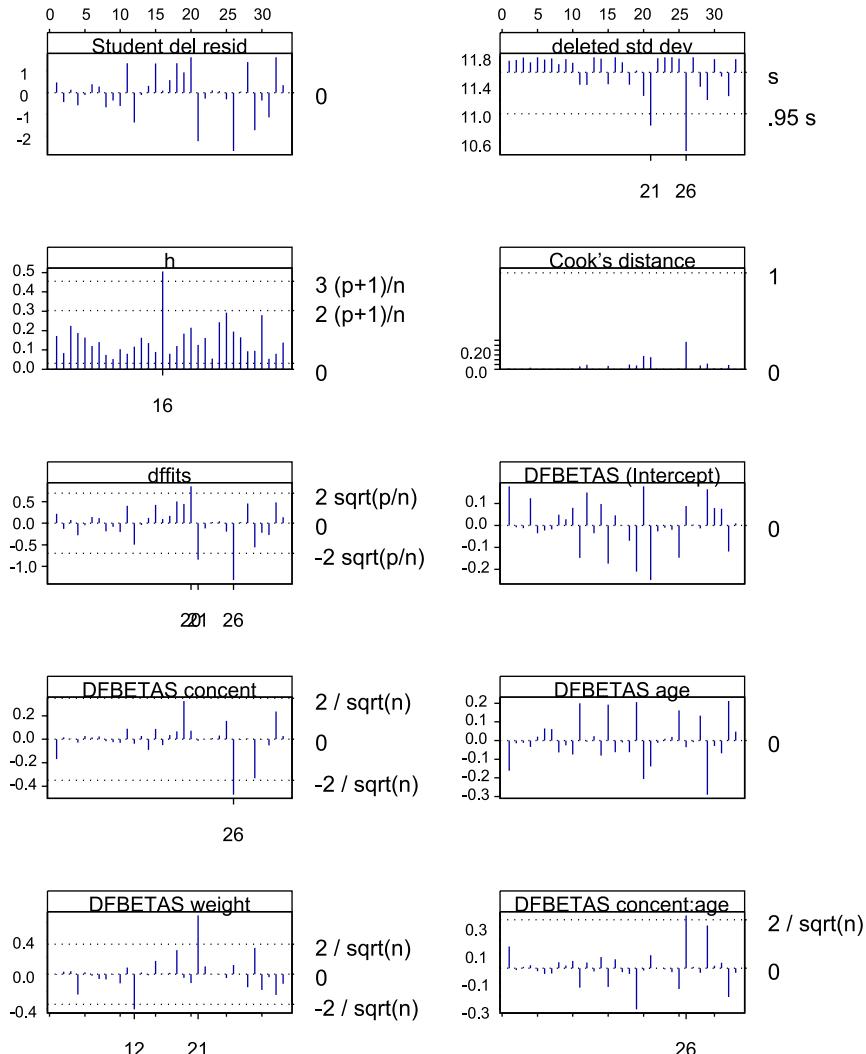


Figure 6.9. Regression diagnostics plot for Kidney data (File: hh/csc/code/kidney2.s)

DFBETAS flags indicate that cases 12, 21, and 26 appreciably impact one or more regression coefficient estimates. The `dffits` flags suggest that cases 20, 21, and 26 have substantial effects on model predictions. In addition, `deleted std dev` hints that cases 21 and 26 impact the overall goodness of fit. We draw the splom in Fig. 6.10 to show these points identified. We see that four of the five flagged points are extreme on at least one variable. Point 16 has low `clearance`, high `concent`, and high `age`. Point 21 has low `weight`. Points 20 and 26 have low `age`. Point 12 is harder to interpret from just this graph as it has high `weight`, but not the highest `weight`.

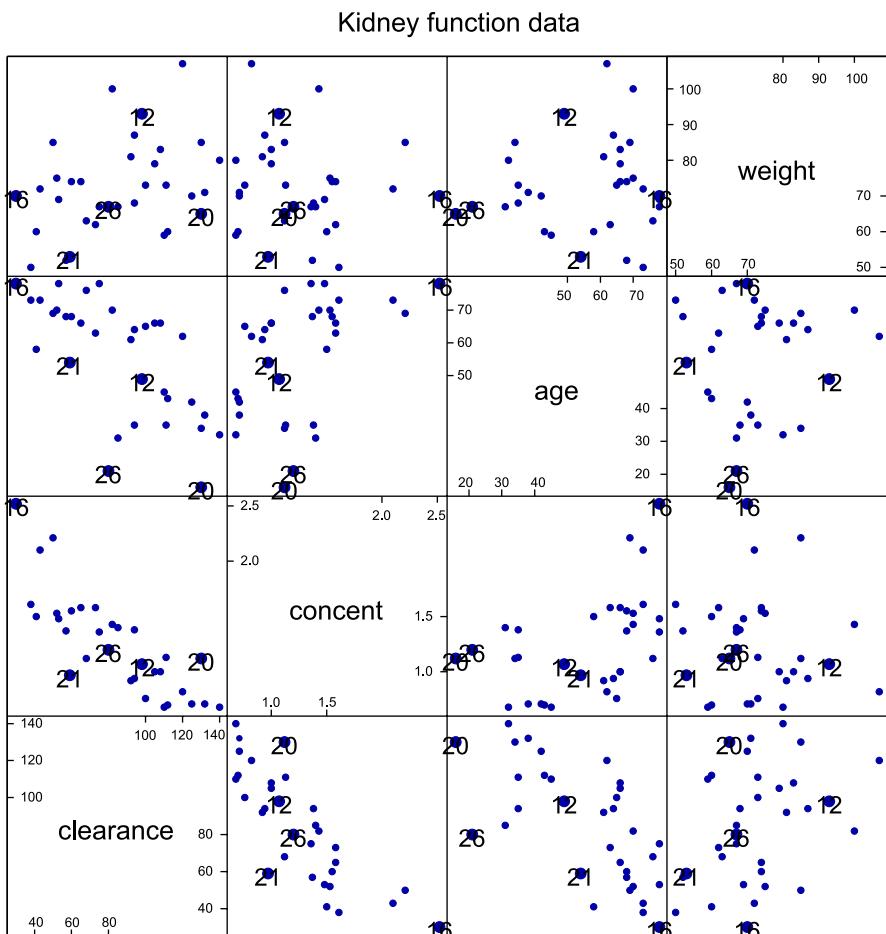


Figure 6.10. Points identified by regression diagnostics in Fig. 6.9 (File: hh/csc/code/kidney2.s)

Analysis of Covariance Plots

6.5

Analysis of covariance (ANCOVA) plots are defined by the Cartesian product of display format. In Fig. 6.11, an example with one factor and one covariate, we show separate panels for each level of the factor on the left side and superposed panels on the right side. The display extends naturally to ANCOVA with two factors and one covariate.

1. The `ancova` function constructs both the ANOVA table and the ANCOVA plot from a single specification of a model formula.
2. Depending on the level of overlap of the x - and y -ranges and the collinearity of the groups, it may be more advantageous to look at the set of separate panels or at the single superposed panel. We display both.

3. We have options in the `ancova` function to show any one of
- Horizontal slopes (ANOVA ignoring the covariate),
 - Common slope (ordinary ANCOVA),
 - Common intercept (regression ignoring the factor),
 - Distinct lines (interaction of factor and covariate).

6.5.1

Example – Hot Dog Data

Hot dogs based on poultry are said to be healthier than ones made from either meat (beef and pork) or all beef. A basis for this claim may be the lower-calorie (fat) content of poultry hot dogs. Is this advantage of poultry hot dogs offset by a higher sodium content than meat hot dogs?

Researchers for *Consumer Reports* analyzed three types of hot dog: beef, poultry, and meat (mostly pork and beef, but up to 15 % poultry meat). The data available in file (`hh/datasets/hotdog.dat`) come from Consumers Union (1986) and were later used by Moore and McCabe (1989):

Type: type of hot dog (beef, meat, or poultry)

Calories: calories per hot dog

Sodium: milligrams of sodium per hot dog

We used these data to model Sodium as a function of the categorical variable Type and the continuous variable Calories.

The model displayed in Fig. 6.11 has common slope and possibly differing intercepts. Displays comparable to Fig. 6.11 can be constructed for the other three models. The common slope model is the correct final model, a conclusion that required support from the ANOVA table for the interaction model. In context, for given fat (calorie) content, poultry hot dogs contain significantly more sodium than beef or meat hot dogs, but the three types have insignificantly different increases in sodium as calories increase.

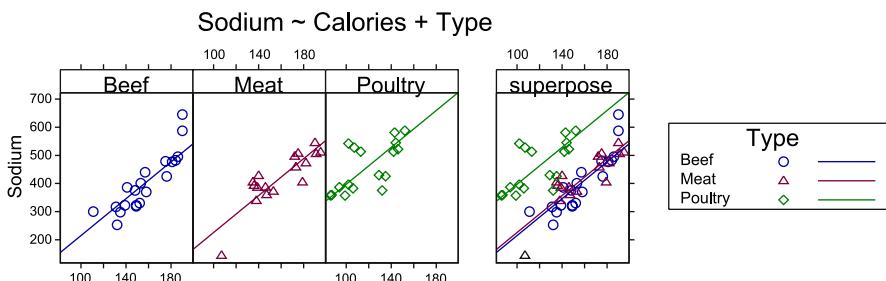


Figure 6.11. The left set of panels show one level each. The right panel, labeled “superpose,” shows all three groups. Sometimes it is easier to interpret the individual panels, other times the superposed panel. We therefore normally print both. The HH Heiberger and Holland (2004) `ancova` function automatically constructs all panels from a single call (Files: `hh/regbb/code/hotdog.s`, `hh/csc/code/hotdog.csc.s`)

Cartesian Product of Model Parameters

6.5.2

Figure 6.12 displays all four models as a Cartesian product of model parameters. The models in the columns of Fig. 6.12 are distinguished by the absence or presence of a parameter for Type – forcing a common intercept in the left column and allowing different intercepts by Type in the right column. The three rows are distinguished by how the covariate Calories is handled: separate slopes by Type in the top row, constant slope for all Types in the middle row, or identically zero slope (horizontal line) in the bottom row.

Figure 6.12 is structured as a set of *small multiples*, a term introduced by Tufte (2001) to indicate repetition of the same graphical design structure. “Small multiples are economical: once viewers understand the design of one slice, they have immediate access to the data in all other slices. Thus, as the eye moves from one slice to the next, the constancy of the design allows the viewer to focus on changes in the data rather than on changes in graphical design (Tufte, 2001, p. 48).” Figure 6.12 may be interpreted as a four-way Cartesian product: slope (α vs. α_i), intercept ($\beta = 0$, β , β_j), individual panels vs. superpose, hot dog type (beef, meat, poultry) with an ordinary two-way scatterplot with a fitted line inside each element of the four-way product.

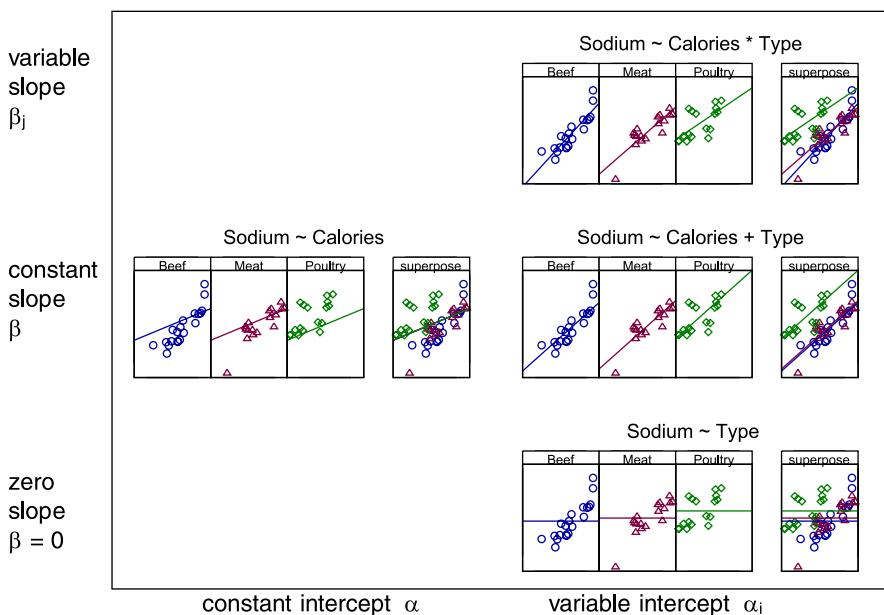


Figure 6.12. Sets of ANCOVA plots as a Cartesian product of models with the intercept having two levels (α and α_i) and slope having three levels (0, β , and β_j). The *middle plot on the right* is identical to the plot in Fig. 6.11 (Files: hh/regbb/code/hotdog.s, hh/csc/code/hotdog.csc.s)

Interaction Plots

Two factors A and B are said to interact if the changes in a response variable Y as factor A goes from one level to another differ depending on the level of the second factor B . The notation in the ANOVA setting is: “ $(\mu_{ij} - \mu_{i'j})$ differs from $(\mu_{ij'} - \mu_{i'j'})$.”

The standard display of an interaction uses separate lines for each level of one factor, the *trace* factor, and by convention connects the points for each level of the second factor, the *x-factor*. Connecting the levels of the trace factor is an interesting convention because the levels are usually – as in this example – on a nominal scale and the implied legitimacy of interpolation is not meaningful. Parallel trace lines indicate lack of interaction. Nonparallel trace lines indicate interaction. The p -value in the ANOVA table is used to determine how far from parallel the lines must be to reject the null hypothesis of no interaction. If interaction is determined to be present, then the main effects are usually not interpretable and we must use simple effects (Sect. 6.6.5) instead.

6.6.1

Two-factor Rhizobium Example

Erdman (1946) discusses experiments to determine if antibiosis occurs between *Rhizobium meliloti* and *Rhizobium trifoli*. Rhizobium is a bacteria, growing on the roots of clover and alfalfa, that fixes nitrogen from the atmosphere into a chemical form plants can use. The research question for Erdman was whether there was an interaction between the two types of bacteria, one specialized for alfalfa plants and the other for clover plants. If there were an interaction, it would indicate that clover bacteria mixed with alfalfa bacteria changed the nitrogen-fixing response of alfalfa to alfalfa bacteria or of clover to clover bacteria. The biology of the experiment says that interaction indicates antibiosis or antagonism of the two types of rhizobium. That is, the goal was to test whether the two types of rhizobium killed each other off. If they did, then there would be less functioning bacteria in the root nodules and consequently nitrogen fixation would be slower.

A portion of Erdman's study involves a two-way factorial layout with factors `strain` at six levels of rhizobium cultures and `comb`, a factor with two distinct bacteria as the two levels. The univariate response is a function of the nitrogen content per milligram of plants grown under these 12 conditions. Erdman was specifically interested in whether two factors interact, as this would have implications for best choice of `strain`. The standard interaction plot for these data is in Fig. 6.13.

6.6.2

Extended Two-way Interaction Plot

Figure 6.14 is a trellis plot that illustrates all the main effects and two-way interactions for a multifactor model. Boxplots for the main effects are shown along the main (SW–NE) diagonal of the matrix. Two standard interaction plots, with interchanged roles for the trace and *x*-factors, are shown along the off-diagonals. In our experience it is not redundant to show the interchanged roles because usually one of the interaction

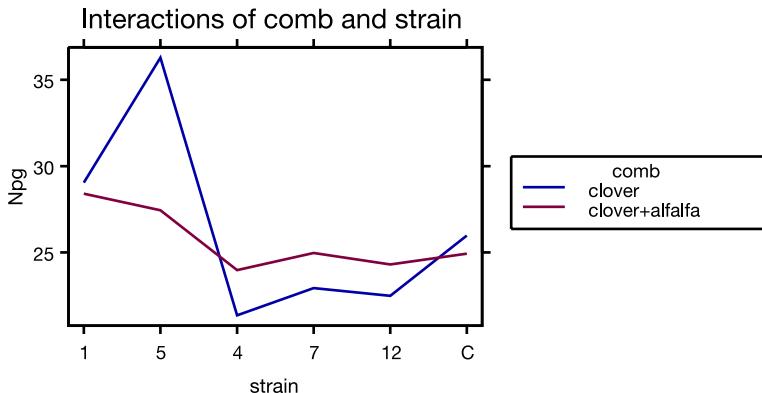


Figure 6.13. `intxplot (Npg ~ strain, groups=comb)`

Standard interaction plot of rhizobium data. At most levels of strain, the clover line is similar to or slightly lower than the combination (clover + alfalfa) line. At strain = 5, the clover line is much higher than the combination line (Files: h2/intx/code/rhizobium-clover.s, hh/csc/code/rhizobium-clover-CSC.s)

Npg: main effects and 2-way interactions

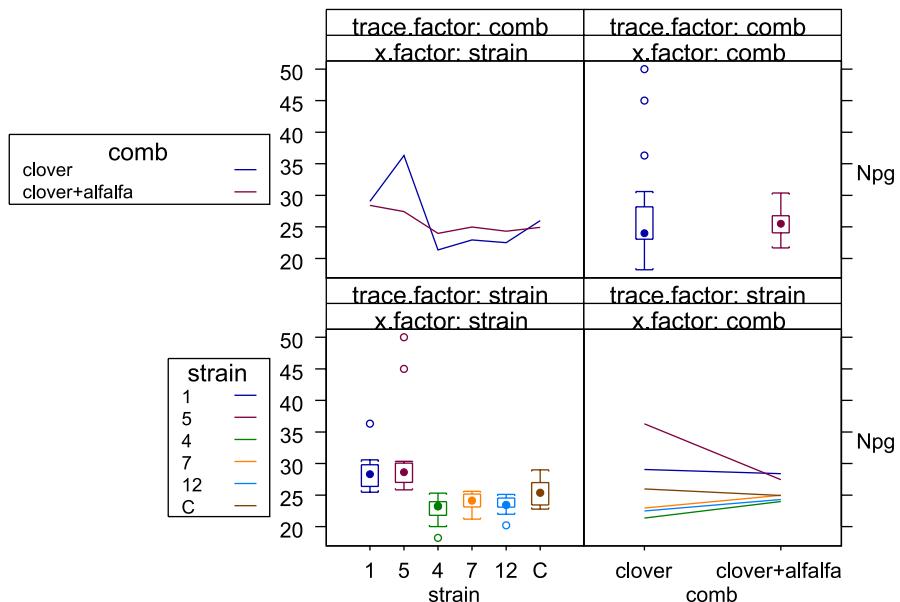


Figure 6.14. Extended interaction plot for *rhizobium* data. The upper-left panel is identical to Fig. 6.13 (File: h2/intx/code/rhizobium-clover.s)

plots is more readily interpretable than the other. Constructing this plot involves the Cartesian product [factors] \times [factors]. Rows are labeled by the trace factor. The color scheme is constant in each row. Columns are labeled by the *x*-factor. The *x*-axis tick marks are constant in each column.

A conclusion from Fig. 6.14 is: as factor comb goes from clover to clover+alfalfa, the response npg (nitrogen per gram) decreases significantly only for level 5 of strain.

6.6.3

Three-factor Vulcanized Rubber Example

An example of a two-way interaction plot involving three factors appears in Fig. 6.15. This is a $5 \times 3 \times 4$ factorial experiment designed to compare the wear resistance of vulcanized rubber. It was desired to maximize wear resistance, along with minimizing the costs of three factors: **filler** at five quality levels, **pretreatment** (three methods), and **raw** (four qualities). The data come from Davies (1954). As there is only one observation from each treatment combination, it is assumed that the three-factor interaction is negligible and therefore available as the model residual.

Figure 6.15 corresponds to the model with all three main effects and all three two-factor interactions. The most striking conclusion from Fig. 6.15 is that for **filler** level 2, the change in response to changing levels of factor **raw** is different from what occurs at the other four levels of **filler**.

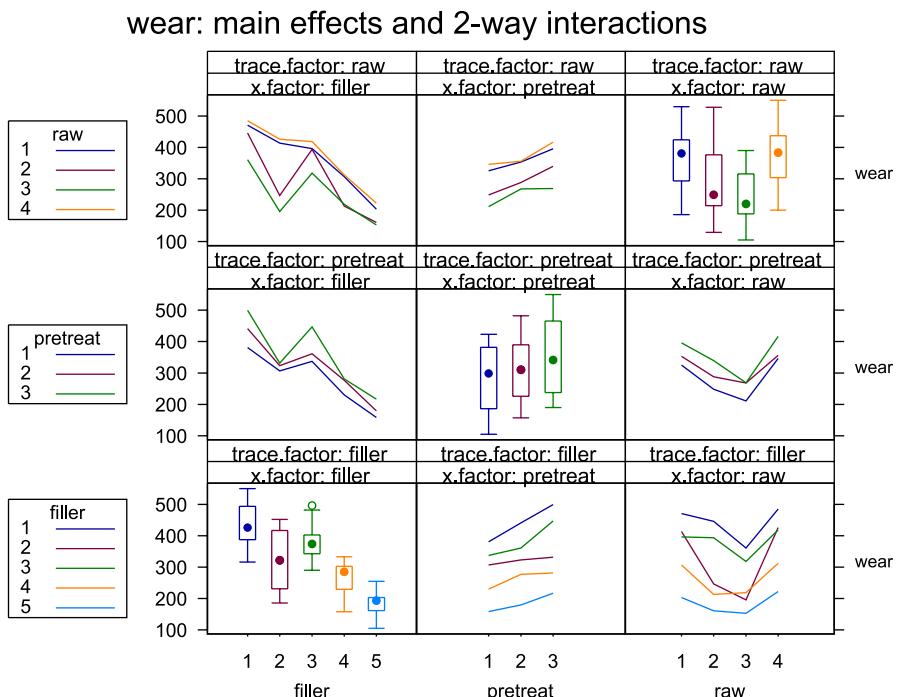


Figure 6.15. HH Figure 13.11, p. 423. Main effects and two-way interactions for wear resistance of vulcanized rubber (File: h2/intx/code/vulcan.intx.s)

Design Issues for the Two-way Interaction Plot

6.6.4

The two-way interaction plot, displayed in Fig. 6.15, shows all the main effects and two-way interactions for designs with two or more factors. The model in S notation is `wear ~ (filler + pretreat + raw)^2`. We construct the figure by analogy with the splom (scatterplot matrix). The rows and columns of the two-way interaction plot are defined by the Cartesian product of the factors.

1. Each main diagonal (SW–NE) panel shows a boxplot for the marginal effect of a factor.
2. Each off-diagonal panel is a standard interaction plot of the factors defining its position in the array. Each point in the panel is the mean of the response variable conditional on the values of the two factors. Each line in the panel connects the cell means for a constant level of the *trace* factor. Each vertically aligned set of points in the panel shows the cell means for a constant value of the *x*-factor.
3. Panels in mirror-image positions interchange the trace- and *x*-factors. This duplication is helpful rather than redundant because one of the orientations is frequently much easier to interpret than the other.
4. The rows are labeled with a key that shows the line type and color for the trace factor by which the row is defined.
5. Each box in the boxplot panels has the same color, and optionally the same line type, as the corresponding traces in its row.
6. The columns are labeled by the *x*-factor.

Two-way Interaction Plots with Simple Effects

6.6.5

Figure 6.14 showed interaction between the `strain` and `comb` factors. Consequently the marginal main effects, showing the average effect of one factor over all levels of the other, were not relevant. We redraw the two-way interaction plot, this time with simple effects in the main diagonal, in Fig. 6.16. Simple effects of one factor, `strain` for example, are conditional on a level of the other factor, in this case `comb`. Simple effects are usually the correct set of means to look at in the presence of interaction. In the left panels of Fig. 6.16 we split each box in the main diagonal, and offset the matching points in the traces in the off-diagonal, to show the nesting of the combinations within each `strain`. The extreme outliers in the `strain=5` are seen to occur entirely in `comb = "clover"`. In the right panels we split each box to show the nesting of the strains within the combinations. The outliers in `comb = "clover"` are seen to be due primarily to `strain=5`.

The 12 boxes in the lower-left and upper-right panels of Fig. 6.16 are identical, each showing the distribution for one of the 12 strain–comb combinations. The order in which the boxes appear, and the colors assigned to them, differs to reflect the different nestings. The individual boxes are not labeled. Instead, the outer factor is labeled and “rug fringes” for the inner factor are displayed. The reader can use the legend to identify specific boxes.

Npg: Simple effects and 2-way interactions

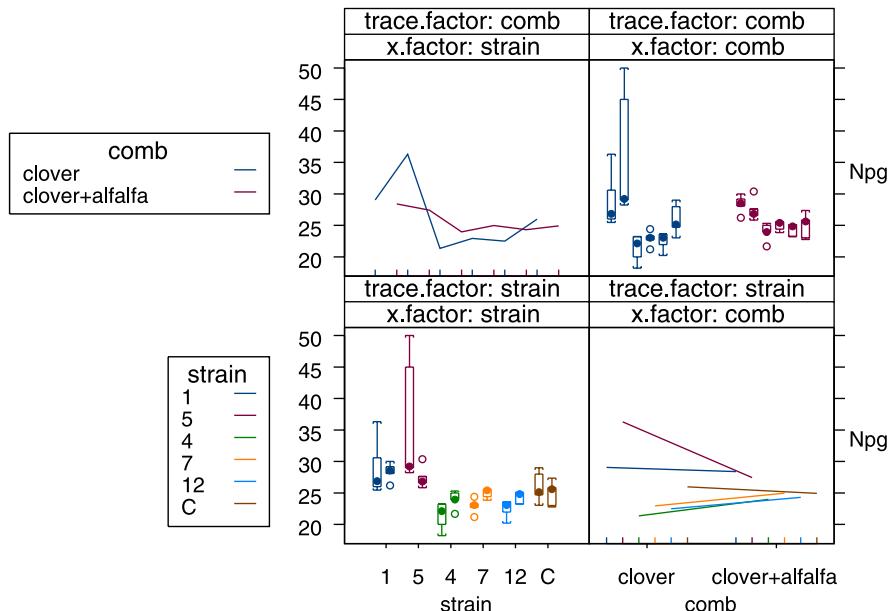


Figure 6.16. Revision of Fig. 6.14 to display simple effects (Files: hh/csc/code/rhizobium-clover3b.s, hh/csc/code/rhizobium-clover5.s)

Figure 6.16 has several noteworthy features.

1. Color coding by rows. The top row of Fig. 6.16 continues the color coding of the top row of Fig. 6.14. Each level of `comb` has its own color. The split of the upper right-hand panel into individual boxes emphasizes the pattern within each level of `comb`. The bottom row of Fig. 6.16 continues the color coding of the bottom row of Fig. 6.14. The visible change in the `Npg` value for similarly colored boxes in the left panel (`comb` within each `strain`) agrees with the direction of the slope of the similarly colored trace lines in the right panel.
2. *x*-axis labels. The left and right panels both have an *x*-axis that itemizes the same levels of the interaction. They differ in the sequence in which those levels are presented.
3. *x*-axis spacing. This version of the graph emphasizes the nesting of the combinations within strains (left-hand panels) by spacing the within-combination boxes slightly closer than the between-combination boxes. The left-hand panels also use color coding by strain. The nesting of strains within combinations (right-hand panels) is shown purely by color coding by combination.
4. Reading by columns. Each boxplot pair in the left-hand panel expands on the difference in traces of means for the corresponding strain. Each trace in the right-hand panel illustrates the difference in mean for the corresponding pair of boxplots for the corresponding strain.

Boxplots

6.7

Assessing Three-way Interaction

6.7.1

Cochran and Cox (1957) report on an experiment to assess the effect of electrical stimulation to prevent the atrophy of muscle tissue in rats. This experiment contained a response variable `wt.d` (weight of treated muscle), a covariate `wt.n` (weight of untreated muscle), and three factors, `current`, `n.treat`, and `minutes`. There are two replications of the entire experiment. It is desired to model `wt.d` as a function of the other variables. The datafile is (`hh/datasets/muscle.dat`).

Figure 6.17 is designed to assess whether there is a significant three-way interaction. The three-way interaction is not significant in this example. If there were a significant three-way interaction, the patterns in boxplots in adjacent rows and columns would not be the same. For example, we note a hint of a difference in the `y.adj ~ minutes` behavior across panels. It has a negative slope in the `galvanic ~ 3` panel and a positive slope in the `faradic ~ 3` panel, but a positive slope in the `galvanic ~ 6` panel and a negative slope in the `faradic ~ 6` panel. However, an ANOVA table for a full model containing the three-factor interaction indicates

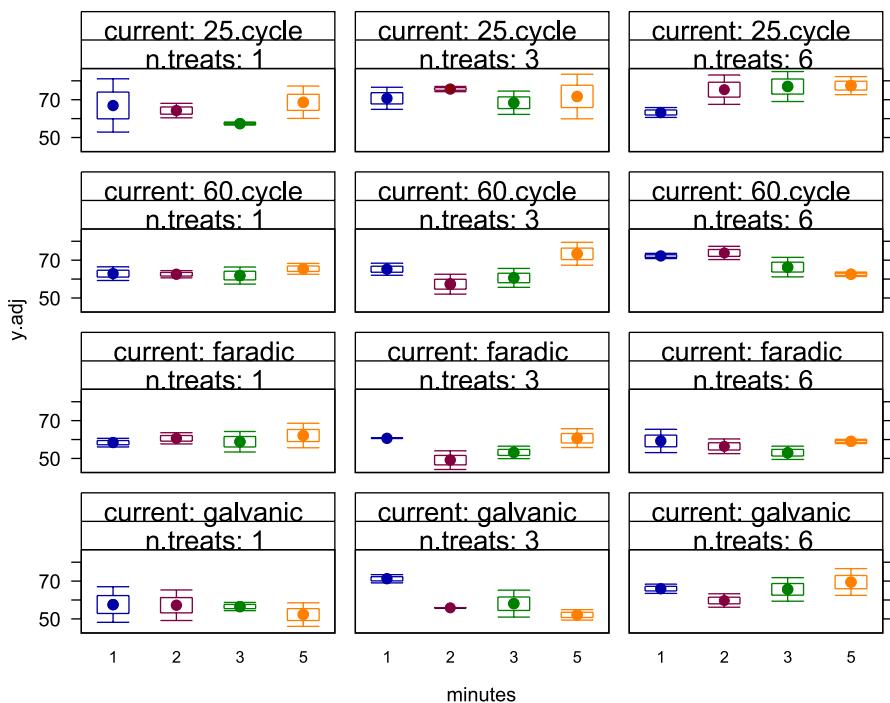


Figure 6.17. Three-way interactions of all effects. One of the $(3!) = 6$ possible orderings (Files: `hh/dsgn/code/ccl76.s`, `hh/csc/code/ccl76-csc.s`)

that these differences in slope are not significant. That is, the three-factor interaction is not statistically significant. The boxplots in Fig. 6.17 are all based on samples of size 2. Such boxplots are a well-defined but uncustomary way to display such a sample.

6.7.2

Sequences of Boxplots

A recent comparative clinical trial at GlaxoSmithKline (Amit et al., 2007) examined the result of monitoring ASAT in two treatment groups A and B after 0, 1, 2, 4, 8, 12, and 24 weeks. ASAT and other clinical biochemistry laboratory blood assays were monitored to give an early signal of potential adverse events in liver function in response to the treatments. The sample size for treatment A was about 200 patients and for treatment B about 400. ASAT (aspartate aminotransferase), an enzyme associated with liver parenchymal cells, is raised in acute liver damage. The display in Fig. 6.18

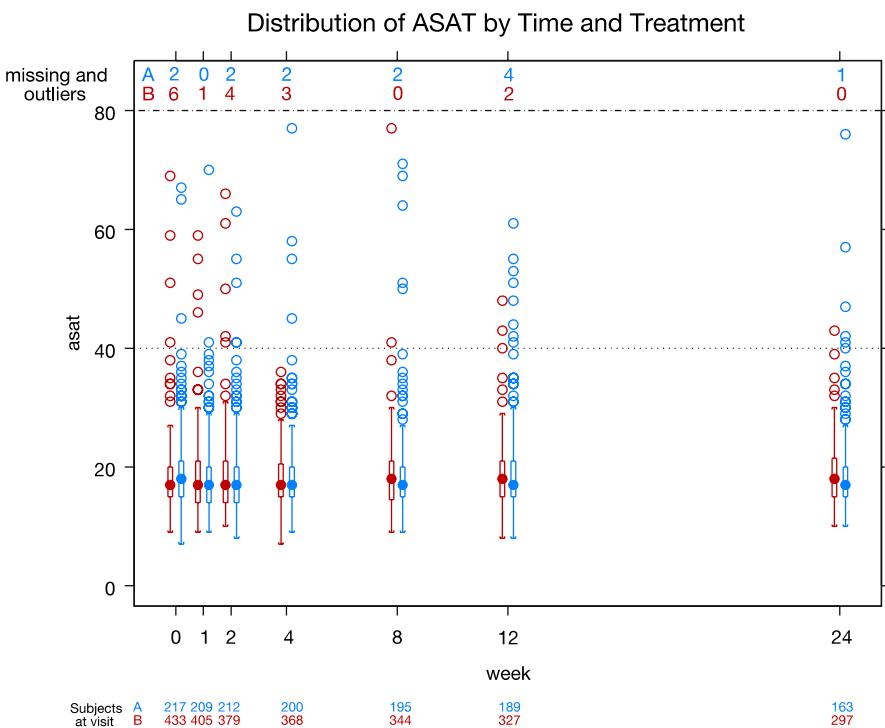


Figure 6.18. Comparative distributions of responses of two treatment groups at unequally spaced points in time. Boxplots are color coded to distinguish between the two treatment groups. The covariate time is correctly scaled on the horizontal axis. Since the data are positively skewed with extreme outliers and with missing values, we choose to truncate observations at $\text{asat}=80$ in order to display full details on the bulk of the data. The sample sizes and numbers of missing and outlying observations are noted on the graph (File: h2/intx/code/lft.asat.s)

was designed specifically to monitor this data situation. The horizontal axis is proportional to the time scale, the treatments are distinguished by color, the sample sizes are identified at each time point, a reference line locates the center of the normal range, outliers are noted, and extreme outliers are identified.

MicropLOTS

6.7.3

Boxplots capture comparative information better than numbers. They don't have to take much space, therefore they can fit into tables of numbers and satisfy both the regulations for displaying numbers and the legibility of displaying graphs. We call the plots *micropLOTS* when they are deliberately sized to fit into a regulation-mandated table of numbers without interfering with the overall layout. When small plots are placed into a table of numbers, they can carry the same or more information per cm² as the numbers themselves. The two examples here have different objectives and therefore are arranged in different patterns.

Example – Catalyst Data

6.7.4

With the catalyst data from Montgomery (1997) we are interested in comparing the concentrations of one component of a liquid mixture in the presence of each of four catalysts. We investigate whether the catalysts provide for equal mean concentrations and then, since this does not appear to be true, we study the extent of differences among the mean concentrations.

The micropLOT of small parallel boxplots in Fig. 6.19 takes up little more page space than the table of summary statistics. We placed y =concentration on the vertical axis, the axis that we have been trained to think of as the appropriate location for a response variable. Under each box we placed the table of relevant statistics: mean, number of replications, and standard deviation. The F -test shows a significant difference between the catalyst means. We can see it visually from the micropLOT or tabularly from the numbers.

	Df	Sum of Sq	Mean Sq	F Value	Pr(F)
catalyst	3	85.68	28.56	9.916	0.0014
Residuals	12	34.56	2.88		

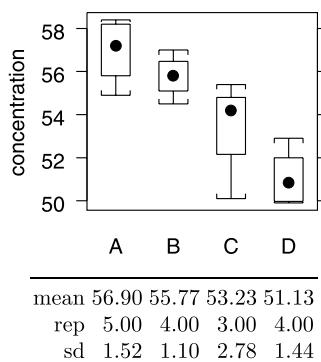


Figure 6.19. ANOVA table, boxplots, and means for each catalyst (File: hh/oway/code/catalystsm.s)

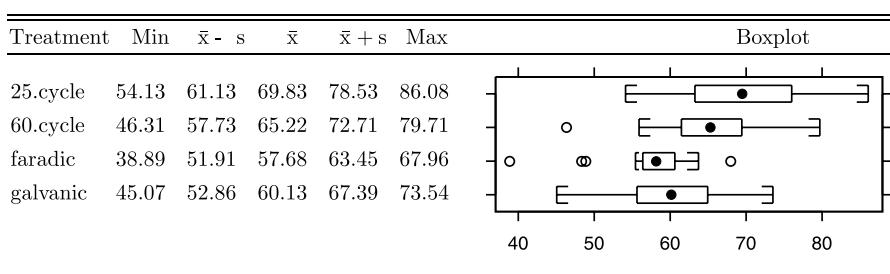


Figure 6.20. Muscle data: distribution statistics and boxplots for adjusted weights (Files: hh/dsgn/code/jsm.ccl76.s, hh/dsgn/code/jsm.ccl76.s)

6.7.5

Example – Muscle Data, continued

The Muscle data was introduced in Sect. 6.7.1. In Fig. 6.20 we display microplots that compare the distributions of responses for each level of the factor current. The microplots in Fig. 6.20 are horizontally oriented. These consume even less vertical page space than the vertically oriented boxplots in Fig. 6.19. The advantage of reduced page space with a horizontal orientation must be weighed against a preference for the vertical response scale for vertically oriented boxplots.

In this setting, we are able to align the numbers with a similar interpretation. Thus, for example, all the means are in the same column. Vertical alignment of comparable numbers often makes it easier for the reader to understand them.

Graphical Display of Incidence and Relative Risk

6.8

In a clinical trial of any new pharmaceutical product, any adverse events (negative side effects of the treatment) must be reported by the company sponsoring the trial to the US Food and Drug Administration. These data in file (h2/datasets/aedotplot.dat) from Amit et al. (2007) is based on a clinical trial at GlaxoSmithKline.

Figure 6.21 compares the incidence rate of multiple adverse events in a clinical trial. It is intended as a graphical summary of adverse events in the trial, highlighting the key differences between two treatments. It is a panel plot with two associated dotplots. The left-hand dotplot gives the comparative incidence of selected adverse events under each of the two treatments. The right-hand dotplot gives estimates and isolated 95 % confidence intervals for the associated relative risks.

We place a vertical reference line at relative risk=1 to facilitate the interpretation that an event having a confidence interval including 1 does not distinguish the effects of treatments A and B. Events having confidence intervals that do not cross 1 suggest a significant difference between treatments A and B.

The adverse events are ordered by relative risk, so that those with the largest increases in risk for the experimental treatment are prominent at the top of the display.

Most Frequent On-Therapy Adverse Events Sorted by Relative Risk

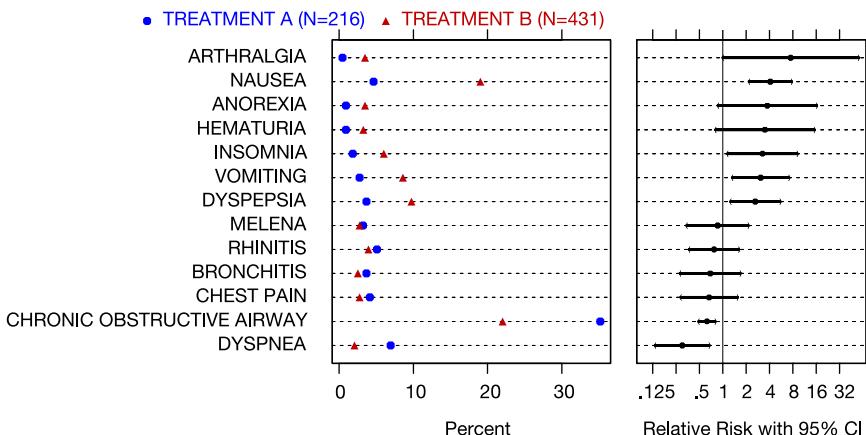


Figure 6.21. The left-hand panel shows the percentage of subjects in each treatment arm who experienced an adverse event. The right-hand panel shows the relative risk and asymptotic 95 % confidence intervals (Files: h2/stdt/code/aerelrisk.s, hh/jsm2005/code/aerelrsk.jsm2005.s)

This could be reversed, to put the largest increases at the bottom if preferred, or the order could be defined by the actual risk in one of the treatment arms rather than the relative risk.

Together, the two panels display four variables for each type of adverse event: the incidences of A and B in the left-hand panel, and the relative risk and its confidence interval in the right-hand panel. The two panels are linked, sharing the same row names on the *y*-axis identifying the types of adverse events experienced.

The two treatments are differentiated by color, and by symbol in case the display is viewed in black and white.

The relative risks are displayed on a log scale, as the asymptotic confidence intervals are symmetrical on this scale. A vertical reference line is drawn to indicate equality of risks, to facilitate appreciation of the size of each relative risk with reference to its confidence interval. The *x*-axis labeling may need adjustment to ensure legibility: in this example, the 0.25 tick label was suppressed to avoid overwriting the 0.125 tick label.

Although this example displays the relative risk in the right-hand panel, any relevant measure of treatment effect for proportions can be used. For example, in dose-ranging studies, the odds ratio from a logistic regression model with dose as a covariate may be substituted for the relative risk.

The size of the displayed confidence interval could be varied. There is clearly a multiplicity issue here, if there is interest in assessing the statistical significance of differences of the relative risk for so many types of events. However, the main aim of this display is to highlight potential signals by providing an estimate of treatment effect and the precision about that estimate. The criterion for inclusion of specific adverse

events in the display needs to be selected with care. In this example, the criterion used was to show only those events that had at least 2 % incidence in treatment B.

This type of panel display is useful for randomized trials with two or more treatment arms. When a study has more than two treatment arms, consider separate sets of graphs for each pairwise comparison of interest.

6.9

Summary

We have illustrated many ways in which sets of graphs are coordinated in a structured manner to increase the amount of information that they carry jointly. The principle of construction used in all the examples is the use of multiple panels placed in a rectangular structure defined by the crossing of sets of labels. While a set is often the set of levels of a factor, it may also be a set of different variables, of different functions of the same variable, of different functions of the data, or of different models applied to the data. We showed examples with several applications of scatterplot matrices, with a series of displays of case statistics used as diagnostics in regression analysis, with interactions of several factors in analysis of variance setting, with boxplots used in conjunction with tabular displays of the same information, and with a graphical display of incidence and relative risk for adverse effects. In all cases, the use of structured sets of graphs enhances the ease of presenting and receiving the information revealed by the analysis.

6.10

File Name Conventions

All examples in this article were constructed with code from the referenced files in the HH online files Heiberger and Holland (2004).

References

- Amit, O., Heiberger, R.M. and Lane, P.W. (2007). Graphical approaches to the analysis of safety data from clinical trials. *Pharmaceutical Statistics*, <http://www3.interscience.wiley.com/cgi-bin/abstract/114129388>.
- Belsley, D.A., Kuh, E. and Welsch, R.E. (1980) *Regression Diagnostics*. Wiley.
- Box, G.E.P. and Cox, D.R. (1964) An analysis of transformations. *J Royal Statist Soc B*, 26:211–252.
- Brewer, C. (2002) *ColorBrewer*. <http://colorbrewer.org>.
- Cochran, W.G. and Cox, G.M. (1957) *Experimental Designs*. Wiley.
- Consumers Union (1986). Hot dogs. *Consumer Reports*, pages 366–367. <http://lib.stat.cmu.edu/DASL/Stories/Hotdogs.html>.
- Davies, O.L. (1954) *Design and Analysis of Industrial Experiments*. Oliver and Boyd.

-
- Erdman, L.W. (1946) Studies to determine if antibiosis occurs among Rhizobia: 1. Between *Rhizobium meliloti* and *Rhizobium trifolii*. *Journal of the American Society of Agronomy*, 38:251–258.
- Harrison, D.E., Betz, J.W., Cailliet, R., Harrison, D.D., Haas, J.W., Colloca, C.J. and Janik, T.J. (2006) Radiographic pseudoscoliosis in healthy male subjects following voluntary lateral translation (side glide) of the thoracic spine. *Archives of Physical Medicine and Rehabilitation*, 87(1):117–22.
- Harrison, D.E., Holland, B., Harrison, D.D. and Janik, T.J. (2002) Further reliability analysis of the Harrison radiographic line drawing methods: Crossed ICCs for lateral posterior tangents and AP modified Risser–Ferguson. *Journal of Manipulative and Physiological Therapeutics*, 25:93–98.
- Heiberger, R.M. and Holland, B. (2004) *Statistical Analysis and Data Display: An Intermediate Course with Examples in S-PLUS, R, and SAS*. Springer-Verlag, New York, first edition. Online files are available from <http://springeronline.com/0-387-40270-5>.
- Montgomery, D.C. (1997) *Design and Analysis of Experiments*. Wiley, 4th edition.
- Moore, D.S. and McCabe, G.P. (1989) *Introduction to the Practice of Statistics*. Freeman.
- Neter, J., Kutner, M.H., Nachtsheim, C.J. and Wasserman, W. (1996) *Applied Linear Statistical Models*. Irwin, fourth edition.
- Ripa, O. and Speakman, J.B. (1951) The plasticity of wool. *Textile Research Journal*, 21:215–221.
- Rossman, A.J. (1994) Televisions, physicians, and life expectancy. *Journal of Statistics Education*. <http://www.amstat.org/publications/jse/archive.htm>.
- Shih, W.J. and Weisberg, S. (1986) Assessing influence in multiple linear regression with incomplete data. *Technometrics*, 28:231–240.
- Tufte, E.R. (2001) *The Visual Display of Quantitative Information*. Graphics Press, second edition.
- Tukey, J.W. (1977) *Exploratory Data Analysis*. Addison-Wesley.

Regression by Parts: Fitting Visually Interpretable Models with GUIDE

III.7

Wei-Yin Loh

7.1	<i>Introduction</i>	448
7.2	<i>Boston Housing Data – Effects of Collinearity</i>	449
7.3	<i>Extension to GUIDE</i>	453
7.4	<i>Mussels – Categorical Predictors and SIR</i>	455
7.5	<i>Crash Tests – Outlier Detection Under Confounding</i>	459
7.6	<i>Car Insurance Rates – Poisson Regression</i>	465
7.7	<i>Conclusion</i>	468

Introduction

Regression modeling often requires many subjective decisions, such as choice of transformation for each variable and the type and number of terms to include in the model. The transformations may be as simple as powers and cross-products or as sophisticated as indicator functions and splines. Sometimes, the transformations are chosen to satisfy certain subjective criteria such as approximate normality of the marginal distributions of the predictor variables. Further, model building is almost always an iterative process, with the fit of the model evaluated each time terms are added or deleted.

In statistical applications, a regression model is generally considered acceptable if it satisfies two criteria. The first is that the distribution of the residuals agrees with that specified by the model. In the case of least-squares regression, this usually means normality and variance homogeneity of the residuals. The whole subject of regression diagnostics is concerned with this problem (Belsley et al., 1980). This criterion can be hard to achieve, however, in complex datasets without the fitted model becoming unwieldy. The second criterion, which is preferred almost exclusively in the machine learning literature, is that the model has low mean prediction squared error or, more generally, deviance.

If model selection is completely software based, the prediction deviance of an algorithm can be estimated by V -fold cross-validation as follows:

1. Randomly divide the dataset into V roughly equal parts.
2. Leaving out one part in turn, apply the algorithm to the observations in the remaining $V - 1$ parts to obtain a model.
3. Estimate the mean prediction deviance of each model by applying the left-out data to it.
4. Average the V estimates to get a cross-validation estimate for the model constructed from all the data.

The value of V may be as small as 2 for very large datasets and as large as the sample size for small datasets. But cross-validation is impractical if the model is selected not by a computer algorithm but by a person making subjective decisions at each stage. In this case, penalty-based methods such as AIC (Akaike, 1973) are often employed. These methods select the model that minimizes a sum of the residual deviance plus a penalty term times a measure of model complexity. Although the rationale makes sense, there is no, and probably never will be, consensus on the right value of the penalty term for all datasets.

A separate, but no less important, problem is how to build a regression model that can be interpreted correctly and unambiguously. In practice, the majority of consumers of regression models often are more interested in model interpretation than in optimal prediction accuracy. They want to know which predictor variables affect the response and how they do it. Sometimes, they also want a rank ordering of the predictors according to the strength of their effects, although this question is meaningless without a more precise formulation. Nonetheless, it is a sad fact that the mod-

els produced by most regression techniques, including the most basic ones, are often difficult or impossible to interpret. Besides, even when a model is mathematically interpretable, the conclusions can be far from unambiguous.

In the rest of this article, we use four examples to highlight some common difficulties: (i) effects of collinearity on modeling Boston housing prices (Sect. 7.2), (ii) inclusion of a categorical predictor variable in modeling New Zealand horse mussels (Sect. 7.4), (iii) outlier detection amid widespread confounding in US automobile crash tests (Sect. 7.5), and (iv) Poisson regression modeling of Swedish car insurance rates (Sect. 7.6). We propose a divide-and-conquer strategy to solve these problems. It is based on partitioning the dataset into naturally interpretable subsets such that a relatively simple and visualizable regression model can be fitted to each subset. A critical requirement is that the partitions be free of selection bias. Otherwise, inferences drawn from the partitions may be incorrect. Another requirement is that the solution be capable of determining the number and type of partitions by itself. In Sect. 7.3 we present an implementation derived from the GUIDE regression tree algorithm (Loh, 2002). At the time of this writing, GUIDE is the only algorithm that has the above properties as well as other desirable features.

Boston Housing Data – Effects of Collinearity

7.2

The well-known Boston housing dataset was collected by Harrison and Rubinfeld (1978) to study the effect of air pollution on real estate price in the greater Boston area in the 1970s. Belsley et al. (1980) drew attention to the data when they used it to illustrate regression diagnostic techniques. The data consist of 506 observations on 16 variables, with each observation pertaining to one census tract. Table 7.1 gives the names and definitions of the variables. We use the version of the data that incorporates the minor corrections found by Gilley and Pace (1996).

Harrison and Rubinfeld (1978) fitted the linear model

$$\begin{aligned} \log(\text{MEDV}) = & \beta_0 + \beta_1 \text{CRIM} + \beta_2 \text{ZN} + \beta_3 \text{INDUS} + \beta_4 \text{CHAS} + \beta_5 \text{NOX}^2 + \beta_6 \text{RM}^2 \\ & + \beta_7 \text{AGE} + \beta_8 \log(\text{DIS}) + \beta_9 \log(\text{RAD}) + \beta_{10} \text{TAX} + \beta_{11} \text{PT} + \beta_{12} \text{B} \\ & + \beta_{13} \log(\text{STAT}) \end{aligned}$$

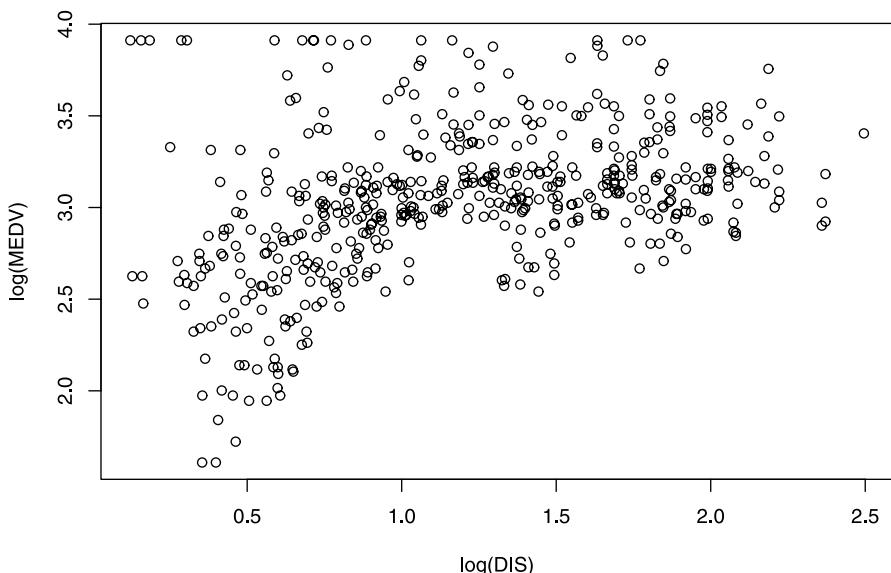
whose least-squares estimates, *t*-statistics, and marginal correlation between each regressor and $\log(\text{MEDV})$ are given in Table 7.2. Note the liberal use of the square and log transformations. Although many of the signs of the coefficient estimates are reasonable and expected, those of $\log(\text{DIS})$ and $\log(\text{RAD})$ are somewhat surprising because their signs contradict those of their respective marginal correlations with the response variable. For example, the regression coefficient of $\log(\text{DIS})$ is negative but the plot in Fig. 7.1 shows a positive slope.

Table 7.1. Variables in Boston housing data

Variable	definition	Variable	definition
ID	Census tract number	TOWN	Township (92 values)
MEDV	Median value in \$1000	AGE	% built before 1940
CRIM	Per capita crime rate	DIS	Distance to employ. centers
ZN	% zoned for lots >25 000 sq. ft.	RAD	Accessibility to highways
INDUS	% nonretail business	TAX	Property tax rate/\$10K
CHAS	1 on Charles River, 0 else	PT	Pupil/teacher ratio
NOX	Nitrogen oxide conc. (p.p.10 ⁹)	B	(% black -63) ² /10
RM	Average number of rooms	LSTAT	% lower-status population

Table 7.2. Least-squares estimates of coefficients and *t*-statistics for regression model for log(MEDV). The marginal correlation between the response variable and each predictor is denoted by ρ

Regressor	β	<i>t</i>	ρ	Regressor	β	<i>t</i>	ρ
Constant	4.6	30.0		AGE	7.1×10^{-5}	0.1	-0.5
CRIM	-1.2×10^{-2}	-9.6	-0.5	log(DIS)	-2.0×10^{-1}	-6.0	0.4
ZN	9.2×10^{-5}	0.2	0.4	log(RAD)	9.0×10^{-2}	4.7	-0.4
INDUS	1.8×10^{-4}	0.1	-0.5	TAX	-4.2×10^{-4}	-3.5	-0.6
CHAS	9.2×10^{-2}	2.8	0.2	PT	-3.0×10^{-2}	-6.0	-0.5
NOX ²	-6.4×10^{-1}	-5.7	-0.5	B	3.6×10^{-4}	3.6	0.4
RM ²	6.3×10^{-3}	4.8	0.6	log(LSTAT)	-3.7×10^{-1}	-15.2	-0.8

**Figure 7.1.** Plot of log(MEDV) vs. log(DIS) for Boston data

To resolve the contradiction, recall that the regression coefficient of $\log(\text{DIS})$ quantifies the linear effect of the variable after the linear effects of the other variables are accounted for. On the other hand, the correlation of $\log(\text{DIS})$ with the response variable ignores the effects of the other variables. Since it is important to take the other variables into consideration, the regression coefficient may be a better measure of the effect of $\log(\text{DIS})$. But this conclusion requires that the linear model assumption be correct. Nonetheless, it is hard to explain the negative linear effect of $\log(\text{DIS})$ when we are faced with Fig. 7.1.

The problem of contradictory signs vanishes when there is only one regressor variable. Although it can occur with two regressor variables, the difficulty is diminished because the fitted model can be visualized through a contour plot. For datasets that contain more than two predictor variables, we propose a divide-and-conquer strategy. Just as a prospective buyer inspects a house one room at a time, we propose to partition the dataset into pieces such that a visualizable model involving one or two predictors suffices for each piece. One difficulty is that, unlike a house, there are no predefined “rooms” or “walls” in a dataset. Arbitrarily partitioning a dataset makes as much sense as arbitrarily slicing a house into several pieces. We need a method that gives interpretable partitions of the dataset. Further, the number and kind of partitions should be dictated by the complexity of the dataset as well as the type of models to be fitted. For example, if a dataset is adequately described by a nonconstant simple linear regression involving one predictor variable and we fit a piecewise linear model to it, then no partitioning is necessary. On the other hand, if we fit a piecewise constant model to the same dataset, the number of partitions should increase with the sample size.

The GUIDE regression tree algorithm (Loh, 2002) provides a ready solution to these problems. GUIDE can recursively partition a dataset and fit a constant, best polynomial, or multiple linear model to the observations in each partition. Like the earlier CART algorithm (Breiman et al., 1984), which fits piecewise constant models only, GUIDE first constructs a nested sequence of tree-structured models and then uses cross-validation to select the smallest one whose estimated mean prediction deviance lies within a short range of the minimum estimate. But unlike CART, GUIDE employs lack-of-fit tests of the residuals to choose a variable to partition at each stage. As a result, it does not have the selection bias of CART and other algorithms that rely solely on greedy optimization.

To demonstrate a novel application of GUIDE, we use it to study the linear effect of $\log(\text{DIS})$ after controlling for the effects of the other variables, *without* making the linear model assumption. We do this by constructing a GUIDE regression tree in which $\log(\text{DIS})$ is the sole linear predictor in each partition or node of the tree. The effects of the other predictor variables, which need not be transformed, can be observed through the splits at the intermediate nodes. Figure 7.2 shows the tree, which splits the data into 12 nodes. The regression coefficients are between -0.2 and 0.2 in all but four leaf nodes. These nodes are colored red (for slope less than -0.2) and blue (for slope greater than 0.2). We choose the cutoff values of ± 0.2 because the coefficient of $\log(\text{DIS})$ in Table 7.2 is 0.2 . The tree shows that the linear effect of $\log(\text{DIS})$ is neither always positive nor always negative – it depends on the values of

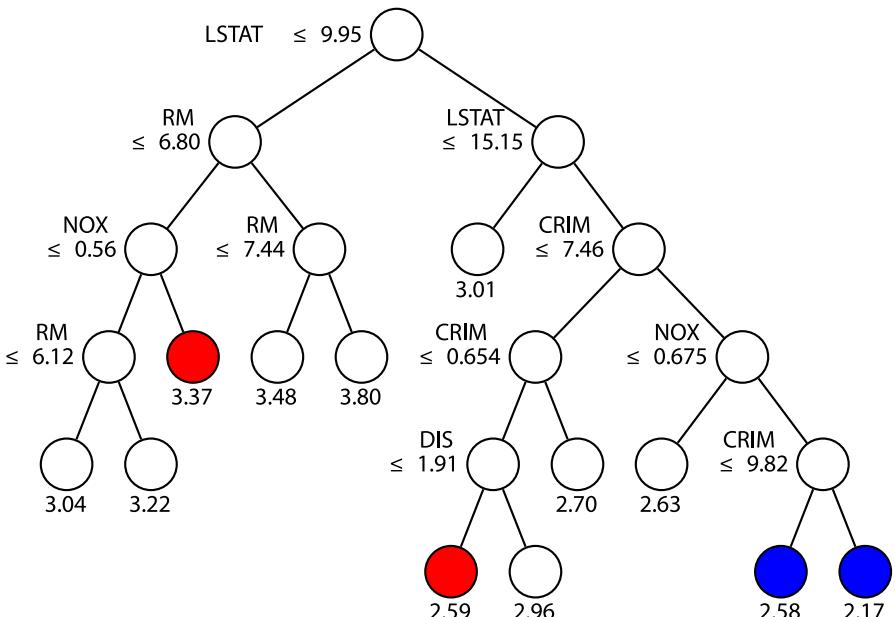


Figure 7.2. GUIDE model for $\log(\text{MEDV})$ using $\log(\text{DIS})$ as linear predictor in each node. At each branch, a case goes to the *left child node* if and only if the given condition is satisfied. The sample mean of $\log(\text{MEDV})$ is printed beneath each leaf node. A *blue leaf node* indicates a slope coefficient greater than 0.2. Correspondingly, a *red leaf node* is associated with a slope coefficient less than -0.2 .

the other variables. This explains the contradiction between the sign of the multiple linear regression coefficient of $\log(\text{DIS})$ and that of its marginal correlation. Clearly, a multiple linear regression coefficient is, at best, an average of several conditional simple linear regression coefficients.

Figure 7.3 explains the situation graphically by showing the data and the 12 regression lines and their associated data points using blue triangles and red circles for observations associated with slopes greater than 0.2 and less than -0.2 , respectively, and green crosses for the others. The plot shows that, after we allow for the effects of the other variables, $\log(\text{DIS})$ generally has little effect on median house price, except in four groups of census tracts (triangles and circles) that are located relatively close to employment centers ($\log(\text{DIS}) < 1$). According to Fig. 7.2, the groups denoted by blue triangles are quite similar. They contain a large majority of the lower-priced tracts and have high values of LSTAT and CRIM . The two groups composed of red circles, on the other hand, are quite different from each other. One group contains tracts in Beacon Hill and Back Bay, two high-priced Boston neighborhoods. The other group contains tracts with DIS lying within a narrow range and with mostly below-average MEDV values. Clearly, the regression coefficient of $\log(\text{DIS})$ in Table 7.2 cannot possibly reveal such details. Unfortunately, this problem is by no means rare. Friedman and Wall (2005), for example, found a similar problem that involves different variables in a subset of these data.

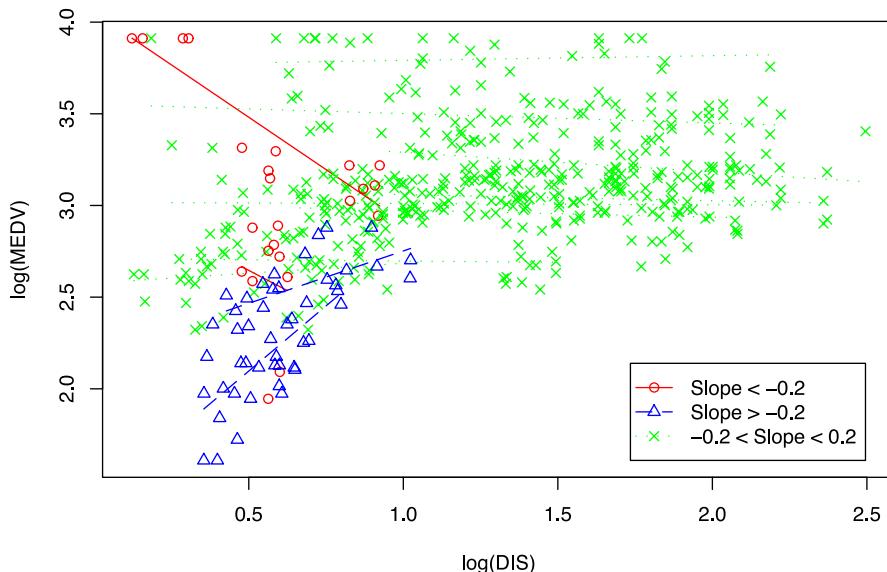


Figure 7.3. Data points and regression lines in the 12 leaf nodes of the Boston data tree. The *blue* and *red* colors correspond to those in Fig. 7.2

Extension to GUIDE

7.3

The basic GUIDE procedure for fitting piecewise constant and piecewise multiple linear models is described in Loh (2002). We present here an extension to fit piecewise simple linear models. The same ideas apply to Poisson regression and to piecewise linear two-predictor models, where the two predictors are chosen at each node via stepwise regression, subject to the standard F-to-enter and F-to-remove threshold values of 4.0 (Miller, 2002). Our extension comprises four algorithms, starting with Algorithm 1.

Algorithm 1: *Tree construction* These steps are applied recursively to each node of the tree, starting with the root node that holds the whole dataset.

1. Let t denote the current node. Fit a simple linear regression to each predictor variable in the data in t . Choose the predictor yielding the smallest residual mean squared error and record its model R^2 .
2. Stop if $R^2 > 0.99$ or if the number of observations is less than $2n_0$, where n_0 is a small user-specified constant. Otherwise, go to the next step.
3. For each observation associated with a positive residual, define the class variable $Z = 1$; else define $Z = 0$.
4. Use Algorithm 2 to find a variable X' to split t into left and right subnodes t_L and t_R .
 - a) If X' is ordered, search for a split of the form $X' \leq x$. For every x such that t_L and t_R contain at least n_0 observations each, find S , the smallest total sum

- of squared residuals obtainable by fitting a simple linear model to the data in t_L and t_R separately. Select the smallest value of x that minimizes S .
- b) If X' is categorical, search for a split of the form $X' \in C$, where C is a subset of the values taken by X' . For every C such that t_L and t_R have at least n_0 observations each, calculate the sample variances of Z in t_L and t_R . Select the set C for which the weighted sum of the variances is minimum, with weights proportional to sample sizes in t_L and t_R .
 5. Apply step 1 to t_L and t_R separately.

Algorithm 2: *Split variable selection*

1. Use Algorithms 3 and 4 to find the smallest curvature and interaction p -values $p^{(c)}$ and $p^{(i)}$ and their associated variables $X^{(c)}$ and $\{X_1^{(i)}, X_2^{(i)}\}$.
2. If $p^{(c)} \leq p^{(i)}$, define $X' = X^{(c)}$ to be the variable to split t .
3. Otherwise, if $p^{(c)} > p^{(i)}$, then:
 - a) If either $X_1^{(i)}$ or $X_2^{(i)}$ is categorical, define $X' = X_1^{(i)}$ if it has the smaller curvature p -value; otherwise, define $X' = X_2^{(i)}$.
 - b) Otherwise, if $X_1^{(i)}$ and $X_2^{(i)}$ are both ordered variables, search over all splits of t along $X_1^{(i)}$. For each split into subnodes t_L and t_R , fit a simple linear model on $X_1^{(i)}$ to the data in t_L and t_R separately and record the total sum of squared residuals. Let S_1 denote the smallest total sum of squared residuals over all possible splits of t on $X_1^{(i)}$. Repeat the process with $X_2^{(i)}$ and obtain the corresponding smallest total sum of squared residuals S_2 . If $S_1 \leq S_2$, define $X' = X_1^{(i)}$; otherwise, define $X' = X_2^{(i)}$.

Algorithm 3: *Curvature tests*

1. For each predictor variable X :
 - a) Construct a $2 \times m$ cross-classification table. The rows of the table are formed by the values of Z . If X is a categorical variable, its values define the columns, i.e., m is the number of distinct values of X . If X is quantitative, its values are grouped into four intervals at the sample quartiles and the groups constitute the columns, i.e., $m = 4$.
 - b) Compute the significance probability of the chi-squared test of association between the rows and columns of the table.
2. Let $p^{(c)}$ denote the smallest significance probability and let $X^{(c)}$ denote the associated X variable.

Algorithm 4: *Interaction tests*

1. For each pair of variables X_i and X_j , carry out the following interaction test:
 - a) If X_i and X_j are both ordered variables, divide the (X_i, X_j) -space into four quadrants by splitting the range of each variable into two halves at the sample median; construct a 2×4 contingency table using the Z values as rows and the quadrants as columns. After dropping any columns with zero column totals, compute the chi-squared statistic and its p -value.

- b) If X_i and X_j are both categorical variables, use their value-pairs to divide the sample space. For example, if X_i and X_j take c_i and c_j values, respectively, the chi-squared statistic and p -value are computed from a table with two rows and number of columns equal to $c_i c_j$ less the number of columns with zero totals.
 - c) If X_i is ordered and X_j is categorical, divide the X_i -space into two at the sample median and the X_j -space into as many sets as the number of categories in its range – if X_j has c categories, this splits the (X_i, X_j) -space into $2c$ subsets. Construct a $2 \times 2c$ contingency table with the signs of the residuals as rows and the $2c$ subsets as columns. Compute the chi-squared statistic and its p -value, after dropping any columns with zero totals.
2. Let $p^{(i)}$ denote the smallest p -value and let $X_1^{(i)}$ and $X_2^{(i)}$ denote the pair of variables associated with $p^{(i)}$.

After Algorithm 1 terminates, we prune the tree with the method described in Breiman et al. (1984, Sect. 8.5) using V -fold cross-validation. Let E_0 be the smallest cross-validation estimate of prediction mean squared error and let α be a positive number. We select the smallest subtree whose cross-validation estimate of mean squared error is within α times the standard error of E_0 . To prevent large prediction errors caused by extrapolation, we also truncate all predicted values so that they lie within the range of the data values in their respective nodes. The examples here employ the default values of $V = 10$ and $\alpha = 0.5$; we call this the *half-SE rule*.

Our split-selection approach is different from that of CART, which constructs piecewise constant models only and which searches for the best variable to split and the best split point simultaneously at each node. This requires the evaluation of all possible splits on every predictor variable. Thus, if there are K ordered predictor variables each taking M distinct values at a node, $K(M - 1)$ splits have to be evaluated. To extend the CART approach to piecewise linear regression, two linear models must be fitted for each candidate split. This means that $2K(M - 1)$ regression models must be computed before a split is found. The corresponding number of regression models for K categorical predictors each having M distinct values is $2K(2^{M-1} - 1)$. GUIDE, in contrast, only fits regression models to variables associated with the most significant curvature or interaction test. Thus the computational savings can be substantial. More important than computation, however, is that CART's variable selection is inherently biased toward choosing variables that permit more splits. For example, if two ordered variables are both independent of the response variable, the one with more unique values has a higher chance of being selected by CART. GUIDE does not have such bias because it uses p -values for variable selection.

Mussels – Categorical Predictors and SIR

In this section, we use GUIDE to reanalyze a dataset, previously studied by Cook (1998), to show that GUIDE can deal with categorical predictor variables as natu-

rally and easily as continuous variables. The data are from the Division of Water Science, DSIR, New Zealand (Camden, 1989). They contain measurements on 201 horse mussels taken from five sites in the Marlborough Sounds, New Zealand, in December 1984. Besides site, each mussel's length, width, depth (all in millimeters), gender (male, female, or indeterminate), viscera mass, muscle mass, and shell mass (all in grams) were recorded, as well as the type of peacock (five categories) found living in its shell.

Cook (1998, p. 214) used Muscle as the response variable and Length, Depth, and Shell as predictors to illustrate his approach to graphical regression. (Note: Cook used the symbols L , W , and S to denote length, depth and shell, respectively.) With the aid of sliced inverse regression (Li, 1991) and power transformations, he found that the mean of Muscle could be modeled by the 1-D subspace defined by the variable

$$\text{SIR1} = 0.001 \text{Length} + 0.073 \text{Depth}^{0.36} + 0.997 \text{Shell}^{0.11}. \quad (7.1)$$

Figure 7.4 shows the banana-shaped plot of Muscle versus SIR1.

The variable Site is not used in Eq. (7.1) because, unlike GUIDE, sliced inverse regression does not easily handle categorical predictor variables. Figure 7.5 shows the result of fitting a GUIDE piecewise best simple linear model to the data. The tree splits first on Site. If Site is neither 2 nor 3, the tree splits further on Depth. The best simple linear predictor is Shell at two of the leaf nodes and Width at the third. Figure 7.6 shows the data and the fitted lines in the leaf nodes of the tree. The plots look quite linear.

On the right side of Fig. 7.5 is the piecewise best two-variable GUIDE model. It splits the data into two pieces, using the same top-level split as the piecewise best

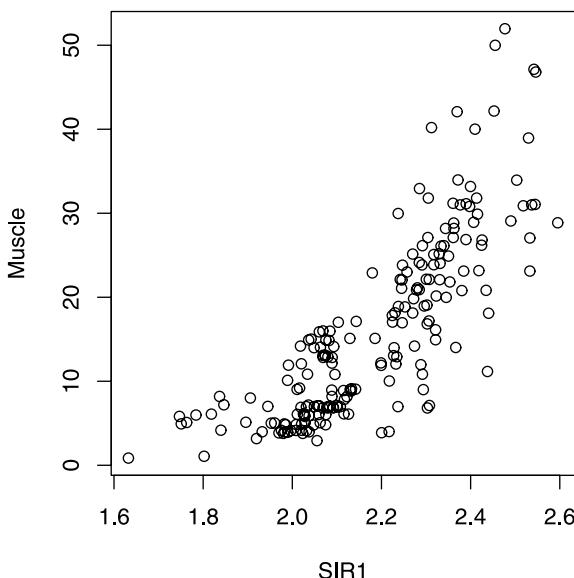


Figure 7.4. Plot of Muscle vs. SIR1 (slightly jittered to reduce overplotting)

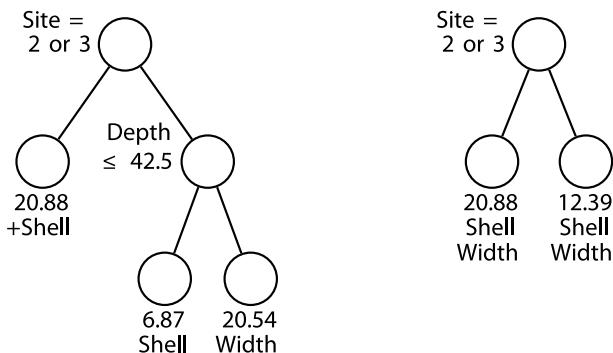


Figure 7.5. Piecewise best simple linear (*left*) and best two-variable linear (*right*) least-squares GUIDE models for mussels data. At each intermediate node, a case goes to the left child node if and only if the condition is satisfied. Beneath each leaf node are the sample mean of Muscle and the selected linear predictors

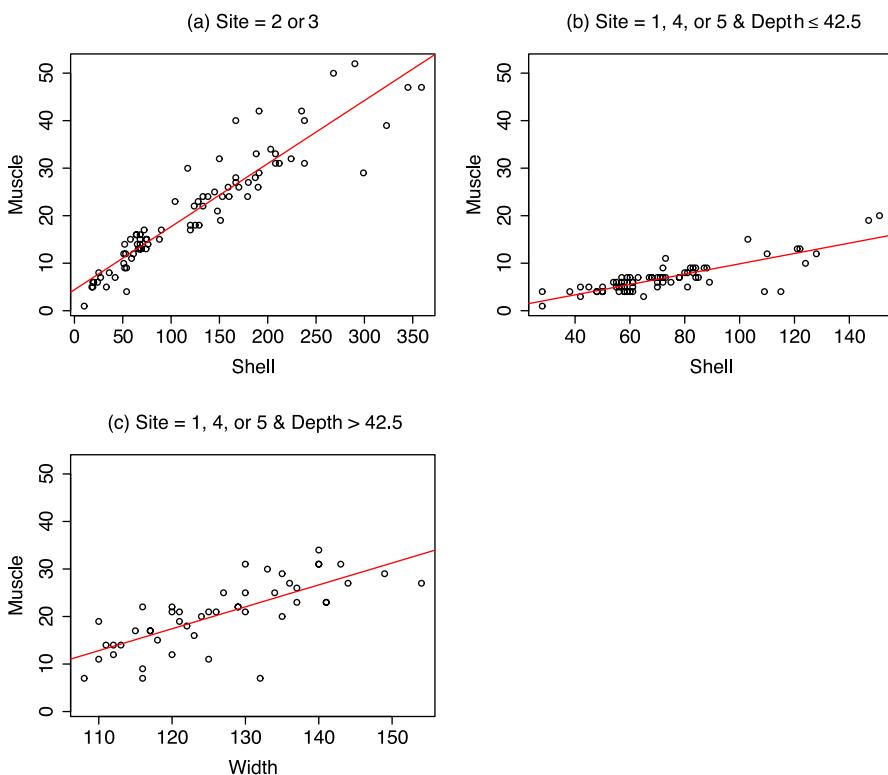


Figure 7.6. Data and fitted lines for the piecewise best simple linear GUIDE model on the left side of Fig. 7.5

simple linear model. Shell and Width are selected as the best pair of linear predictors in both leaf nodes. Figure 7.7 shows shaded contour plots of the fitted functions and data points. Clearly, the mussels from Sites 2 and 3 tend to have greater muscle mass than those from Sites 1, 4, and 5.

Since Site is an important predictor in the GUIDE models, we redraw the SIR plot using different symbols to indicate site information in panel a of Fig. 7.8. The banana-shaped plot is seen to be an artifact caused by combining the sites; the data

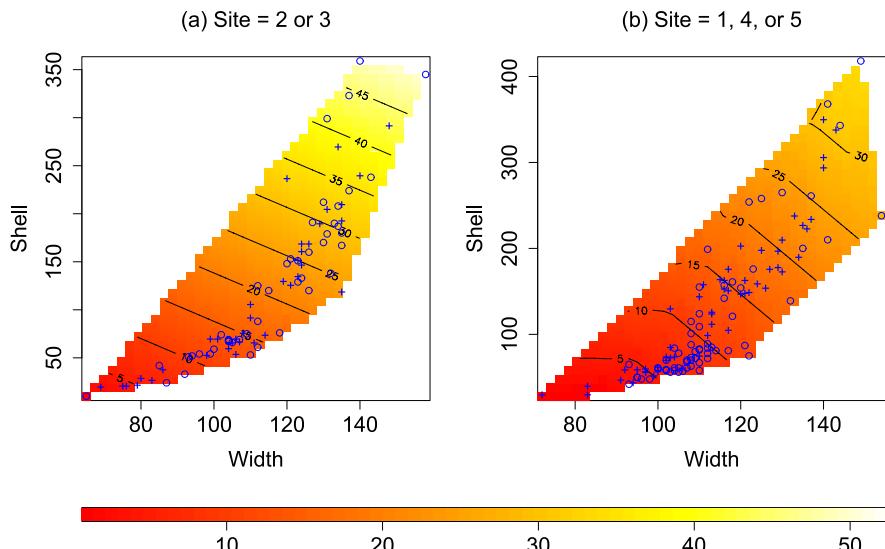


Figure 7.7. Shaded contour plots of fitted functions and data points for the piecewise best two-variable linear model on the right side of Fig. 7.5

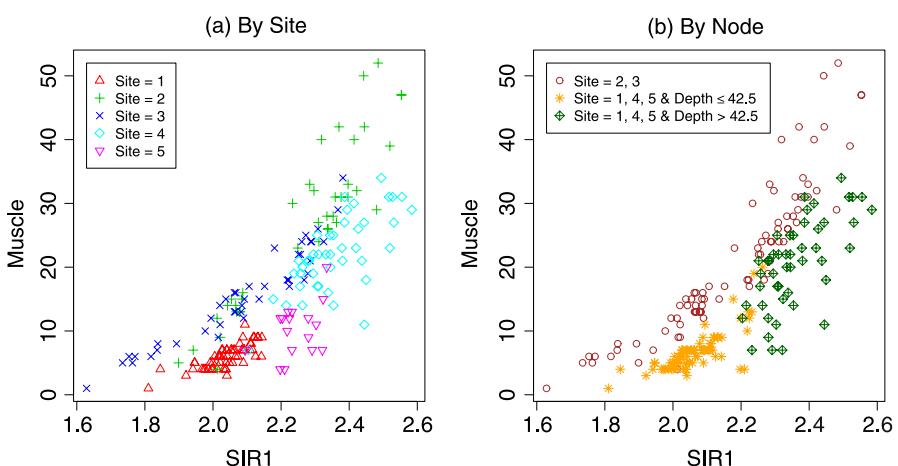


Figure 7.8. Muscle vs. SIR1 by site and by the nodes of the tree in Fig. 7.5a

points within each site are quite linear. Panel b again employs different symbols to indicate leaf node membership according to the piecewise best simple linear model in Fig. 7.6. We see that node membership divides the data into three clusters, with the first cluster belonging to Sites 2 and 3 and the second and third clusters to Sites 1, 4, and 5, depending on whether or not $\text{Depth} \leq 42.5$. The first cluster (indicated by circles) clearly exhibits the most pronounced curvature. This suggests that the nonlinear relationship between Muscle and SIR1 is mainly due to the observations from Sites 2 and 3. On the other hand, we saw in Fig. 7.6a that at these two sites, Muscle varies roughly linearly with Shell. Thus it is likely that the curvature in Fig. 7.4 is at least partly due to the power transformation of Shell in the definition of SIR1 in Eq. (7.1).

Crash Tests – Outlier Detection Under Confounding

7.5

The data in this example are obtained from 1789 vehicle crash tests performed by the National Highway Transportation Safety Administration (NHTSA) between 1972 and 2004 (<http://www-nrd.nhtsa.dot.gov>). The response variable is the square root of the head injury criterion ($\sqrt{\text{hic}}$) measured on a crash dummy. Values of $\sqrt{\text{hic}}$ range from 0 to 100, with 30 being the approximate level beyond which a person is expected to suffer severe head injury. Twenty-five predictor variables, defined in Table 7.3, provide information on the vehicles, dummies, and crash tests. Angular variables are measured clockwise, with -90, 0, and 90 degrees corresponding to the driver's left, front, and right sides, respectively. About one quarter of the vehicle models are tested more than once, with the most often tested being the 1982 Chevy Citation, which was tested 15 times.

Table 7.3. Variables for NHTSA data

Name	Description	Name	Description
hic	Head injury criterion	make	Car manufacturer (62 values)
year	Car model year	mkmodel	Car model (464 values)
body	Car body type (18 values)	transm	Transmission type (7 values)
engine	Engine type (15 values)	engdsp	Engine displacement (liters)
vehtwt	Vehicle total weight (kg)	colmec	Collapse mechanism (11 values)
vehwid	Vehicle width (mm)	modind	Car modification indicator (5 values)
vehspd	Vehicle speed (km/h)	crbang	Crabbed angle (degrees)
tksurf	Track surface (5 values)	pdoft	Principal direction of force (degrees)
tkcond	Track condition (6 values)	impang	Impact angle (degrees)
occtyp	Occupant type (10 values)	dumsiz	Dummy size (6 values)
seposn	Seat position (5 values)	barrig	Barrier rigidity (rigid/deformable)
barshp	Barrier shape (14 values)	belts	Seat belt type (none/2pt/3pt)
airbag	Airbag present (yes/no)	knee	Knee restraint present (yes/no)

Our goal is to identify the vehicle models for which the hic values are unusually high, after allowing for the effects of the predictor variables. Since almost all the tests involve two or more crash dummies, we will give two separate analyses, one for the driver and another for the front passenger dummies. After removing tests with incomplete values, we obtain 1633 and 1468 complete tests for driver and front passenger, respectively. The tests for driver dummies involve 1136 different vehicle models. Figure 7.9 shows a histogram of the $\sqrt{\text{hic}}$ values for the driver data (the histogram for front passenger is similar). There are 22 vehicle models with $\sqrt{\text{hic}}$ values greater than 50. They are listed in Table 7.4, arranged by model year, with the total number of times tested and (in parentheses) the $\sqrt{\text{hic}}$ values that exceed 50. For example, the 2000 Nissan Maxima was tested eight times, of which five gave $\sqrt{\text{hic}}$ values greater than 50.

To identify the outliers after removing the effects of the predictor variables, we need to regress the response values on the predictors. The regression model must be sufficiently flexible to accommodate the large number and mix of predictor variables and to allow for nonlinearity and interactions among them. It must also be suitable for graphical display, as the outliers will be visually identified. These requirements are well satisfied by a piecewise simple linear GUIDE model, which is shown in Fig. 7.10. The tree has three leaf nodes, partitioning the data according to vehspd. Beneath each leaf node is printed the sample mean response for the node and the selected signed linear predictor. We see that model year is the most important linear predictor

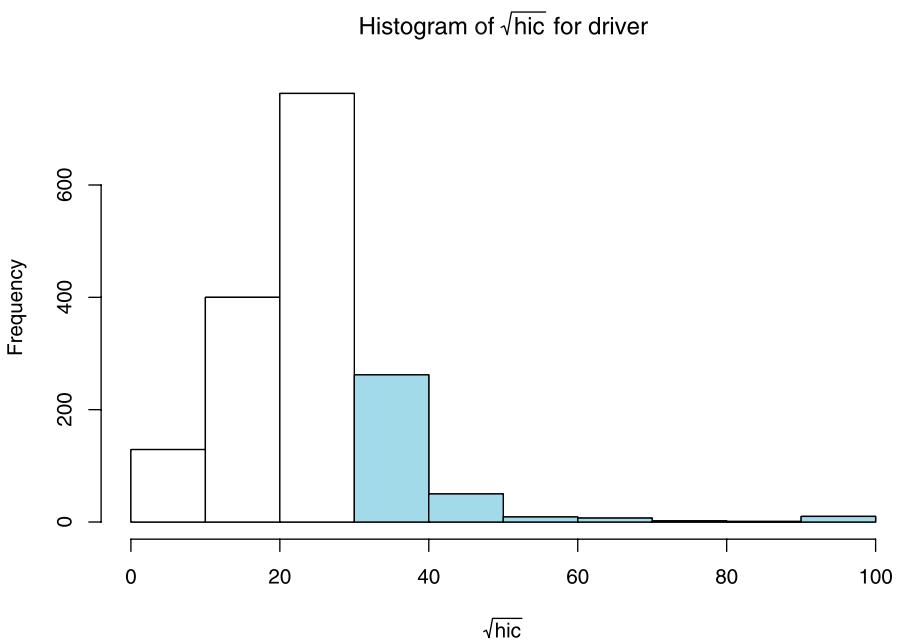


Figure 7.9. Histogram of $\sqrt{\text{hic}}$ for driver dummy data. Shaded areas correspond to $\sqrt{\text{hic}} > 30$

Table 7.4. Vehicles with $\sqrt{\text{hic}}$ (in parentheses) greater than 50 registered on driver dummies. The column labeled # gives the total number of each model tested. For example, five out of eight 2000 Nissan Maxima's tested had $\sqrt{\text{hic}} > 50$

#	Model	#	Model
1	1979 Dodge Colt (96)	2	1983 Renault Fuego (57)
12	1979 Honda Civic (53)	1	1984 Ford Tempo (54)
1	1979 Mazda B2000 Pickup (55)	1	1988 Chevy Sportvan (61)
1	1979 Peugeot 504 (68)	1	1990 Ford Clubwagon MPV (51)
2	1979 Volkswagen Rabbit (62)	4	1995 Honda Accord (94)
3	1980 Chevy Citation (65)	5	2000 Nissan Altima (100)
1	1980 Honda Civic (52)	8	2000 Nissan Maxima (69, 72, 100, 100, 100)
1	1980 Honda Prelude (55)	4	2000 Saab 38235 (72)
2	1981 Mazda GLC (51)	4	2000 Subaru Legacy (100, 100)
2	1982 Chrysler Lebaron (51)	11	2001 Saturn L200 (68, 87, 100)
2	1982 Renault Fuego (61)	9	2002 Ford Explorer (100)

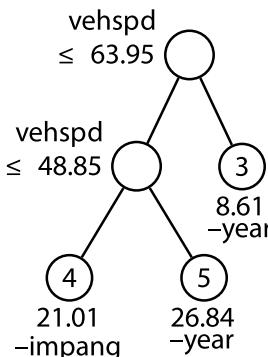


Figure 7.10. Piecewise-simple linear GUIDE model for driver data. At each intermediate node, a case goes to the *left child node* if and only if the condition is satisfied. Beneath each *leaf node* are the sample mean of $\sqrt{\text{hic}}$ and the selected signed linear predictor

in two of the three leaf nodes, and `impang` in the third. In the latter (Node 4), injury tends to be more severe if the impact occurs on the driver side (`impang = -90`).

A very interesting feature of the tree is that the sample mean response is lowest in Node 3, which has the highest values of `vehspd` (>63.95). At first glance, this does not make sense because injury severity should be positively correlated with vehicle speed. It turns out that the design of the experiment causes some variables to be confounded. This is obvious from the upper row of plots in Fig. 7.11, which show the data and regression lines in the three leaf nodes of the tree, using different symbols to indicate whether a vehicle is crashed into a rigid or a deformable barrier. We see that the proportion of tests involving deformable barriers is much greater at high speeds (Node 3) than at low speeds. This would explain the lower injury values among the high-speed tests.

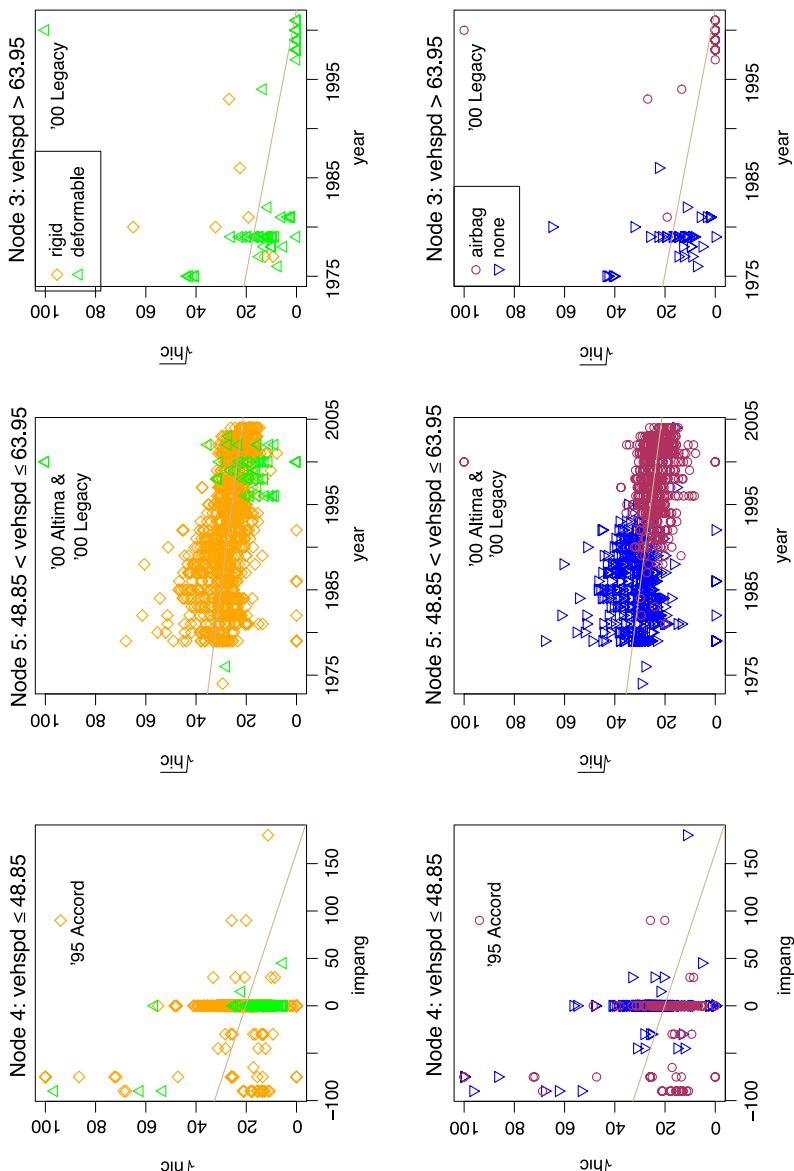


Figure 7.11. Data and fitted regression functions in the leaf nodes of the tree model in Fig. 7.10, using different symbols for *barrig* (top) and *airbag* (bottom) values

Another variable confounded with *vehspd* is *airbag*. This can be seen in the second row of plots in the same figure, where different symbols are used to indicate whether a vehicle is equipped with an airbag or not. We see that almost all vehicles manufactured from 1990 onwards have airbags and that their presence is associated

with lower hic values. Since there is a fair number of such vehicles in Node 3, this could also account for the low sample mean response.

Finally, a third confounding variable is evident in Fig. 7.12, which shows barplots of the proportions of barrier shape type (`barshp`) within each leaf node of the tree. Node 3, whose bars are colored green, stands out in that barrier shapes EOB, GRL, IAT, MBR, and SGN practically never appear in the other two nodes. For some reason, the testers seem to prefer these barrier shapes for high-speed crashes. Thus barrier shape is yet another possible explanation for the low mean response value in the node.

Despite these difficulties, it is clear from the plots that three vehicle models stand out as outliers: 1995 Honda Accord, 2000 Nissan Altima, and 2000 Subaru Legacy. All are foreign imports. The 2000 Subaru Legacy appears as an outlier in two separate tests, one at moderate speed and one at high speed.

Figure 7.13 shows the corresponding tree model for the front passenger data. Now airbag and barrier rigidity appear as split variables after the top-level split on `vehspd`. The plots of the data in the leaf nodes are presented in Fig. 7.14. Everything seems to make sense: injury is less severe when a vehicle is equipped with airbags and when it is crashed into a deformable barrier, and also if impact occurs on the driver side (Node 6). It is interesting to note that in Node 5, where $vehspd \leq 48.25$ and the vehicles are equipped with airbags, rigid barriers are used for the higher speeds and deformable barriers for the lower speeds. This may exaggerate the effect of `vehspd` in this node. The outliers for these data turn out to be all domestic models: 1978

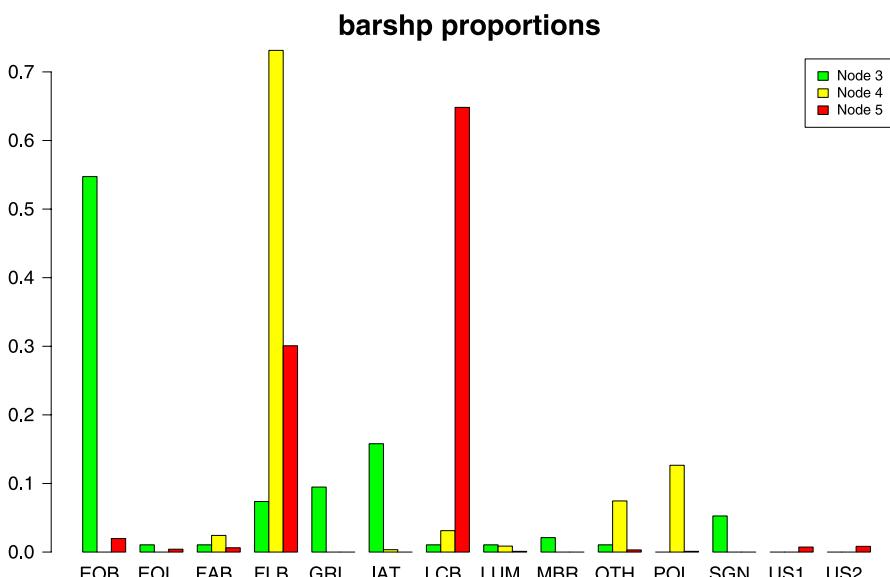


Figure 7.12. Proportions of different barrier shapes within the three leaf nodes of the tree model in Fig. 7.10. The lengths of the bars sum to one for each color

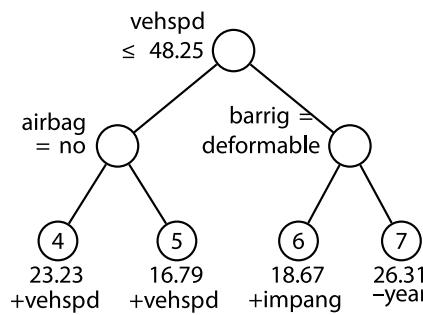


Figure 7.13. Piecewise-simple linear GUIDE model for front passenger data. At each intermediate node, a case goes to the *left child node* if and only if the condition is satisfied. Beneath each *leaf node* are the sample mean of $\sqrt{\text{hic}}$ and the selected signed linear predictor

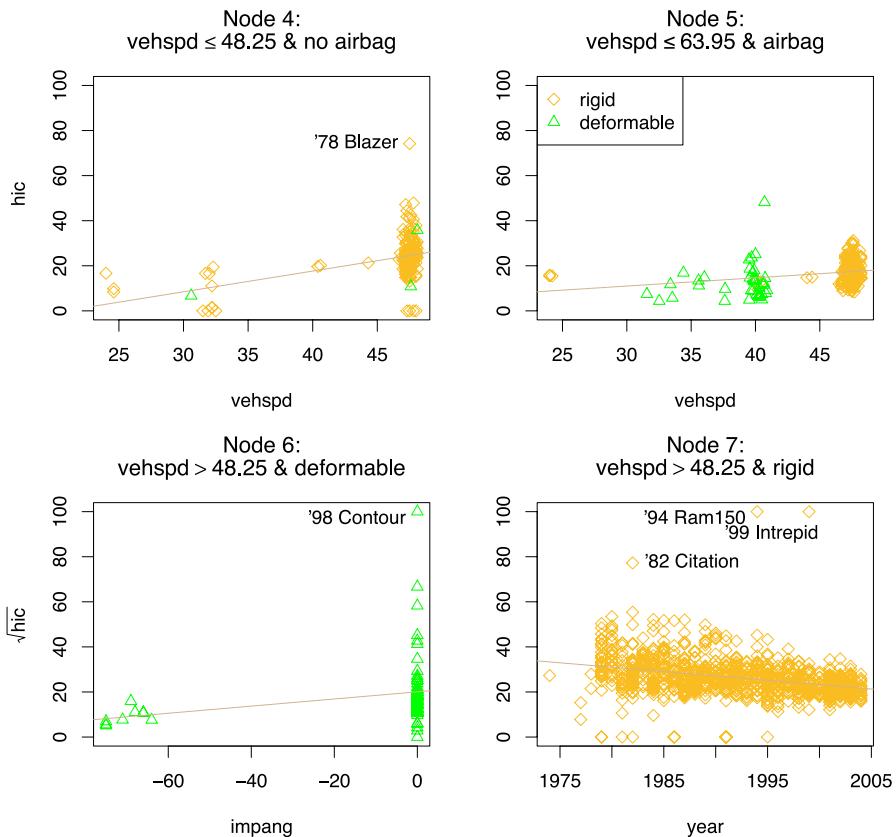


Figure 7.14. Data and fitted regression functions in the leaf nodes of the tree model in Fig. 7.13, with different symbols for *barrig* type

Chevy Blazer, 1982 Chevy Citation, 1994 Ford Ram 150, 1998 Ford Contour, and 1999 Dodge Intrepid.

The good news from both analyses is that no obvious outliers are found among vehicles newer than the 2000 model year.

Car Insurance Rates – Poisson Regression

7.6

The data are from Statlib. A subset of them is given in Andrews and Herzberg (1985, pp. 415–421). The original data consist of information on more than two million third-party automobile insurance policies in Sweden for the 1977 year. For each policy the annual mileage, bonus class (on a seven-point scale), geographical zone (seven categories), and make of car (nine categories) were recorded. Annual mileage is discretized into five categories: (1) less than 10,000 km/year, (2) 10 000–15 000 km/year, (3) 15 000–20 000 km/year, (4) 20 000–25 000 km/year, and (5) more than 25 000 km/year (Hallin and Ingenbleek, 1983). These four explanatory variables yield a $5 \times 7 \times 7 \times 9$ table with 2205 possible cells. For each cell, the following quantities were obtained:

1. Total insured time in years,
2. Total number of claims,
3. Total monetary value of the claims.

Twenty-three cells are empty.

We will model claim rate here. According to Andrews and Herzberg (1985, p. 414), a Swedish Analysis of Risk group decided that a multiplicative model (i.e., an additive Poisson loglinear model) for claim rate is fairly good, and that any better model is too complicated to administer. To challenge this conclusion, we will use GUIDE to fit a piecewise-additive Poisson loglinear model for number of claims, using the log of number of claims as offset variable. Bonus class and mileage class are treated as continuous, and zone and make as categorical variables.

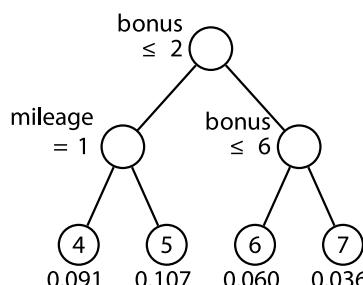


Figure 7.15. GUIDE multiple linear Poisson regression tree for car insurance data. At each intermediate node, a case goes to the *left child node* if and only if the condition is satisfied. The *number in italics* beneath each *leaf node* is the sample claim rate

Figure 7.15 shows the GUIDE tree, which has four leaf nodes and an estimated prediction deviance (based on tenfold cross-validation) about 25% lower than that of a single additive loglinear model. From the sample average claim rates printed beneath the leaf nodes, we see that bonus classes 1 and 2 tend to yield rates two to three times as large as the other bonus classes.

The estimated regression coefficients in the leaf nodes are given in Table 7.5, where the coefficients of the dummy variables corresponding to the first levels of each categorical variable are set to zero. As may be expected, Mileage has a positive slope coefficient in all three nodes where it is not constant. The slope for Bonus is, however, negative wherever it is not constant. Thus the higher the Bonus class, the lower the Claim rate tends to be.

For Make, the coefficient for level 4 has a larger negative value than the coefficients for the other Make levels, uniformly across all the nodes. Hence this level of Make is likely to reduce claim rate the most. In contrast, the coefficient for level 5 of Make is positive in all nodes and is larger than the coefficients for all other levels in three nodes – it is second largest in the remaining node. This level of Make is thus most likely to increase claim rate. The situation is quite similar for Zone: since all its coefficients are negative except for level 1, which is set to zero, that level is most likely to increase claim rate, across all four nodes. The Zone level most likely to decrease claim rate is 7, which has the largest negative coefficient in three of the nodes and the second largest negative coefficient in the fourth node. Figure 7.16 presents the results more vividly by showing barplots of the coefficients for Make and Zone by node. The relative sizes of the coefficients are fairly consistent between nodes.

Table 7.5. Regression estimates for GUIDE model using set-to-zero constraints for the first levels of Make and Zone

	Node 4	Node 5	Node 6	Node 7
Constant	-0.8367	-1.0639	-2.3268	-3.3725
Mileage	Aliased	0.0427	0.1425	0.1439
Bonus	-0.5202	-0.4500	-0.0992	aliased
Make=2	-0.1705	-0.0356	0.0756	0.1375
Make=3	-0.2845	-0.2763	-0.2038	-0.2247
Make=4	-1.0964	-0.7591	-0.6555	-0.4595
Make=5	0.0892	0.1685	0.1468	0.1308
Make=6	-0.5971	-0.5437	-0.3274	-0.2563
Make=7	-0.3330	-0.2900	-0.0405	0.0214
Make=8	-0.0806	-0.0848	0.0233	-0.0584
Make=9	-0.4247	-0.2097	-0.0592	0.0039
Zone=2	-0.3306	-0.2735	-0.2525	-0.1837
Zone=3	-0.5220	-0.3905	-0.4046	-0.3303
Zone=4	-0.8298	-0.5692	-0.5986	-0.5120
Zone=5	-0.4683	-0.3927	-0.3533	-0.2384
Zone=6	-0.7414	-0.5437	-0.5830	-0.4273
Zone=7	-0.8114	-0.8538	-0.7760	-0.6379

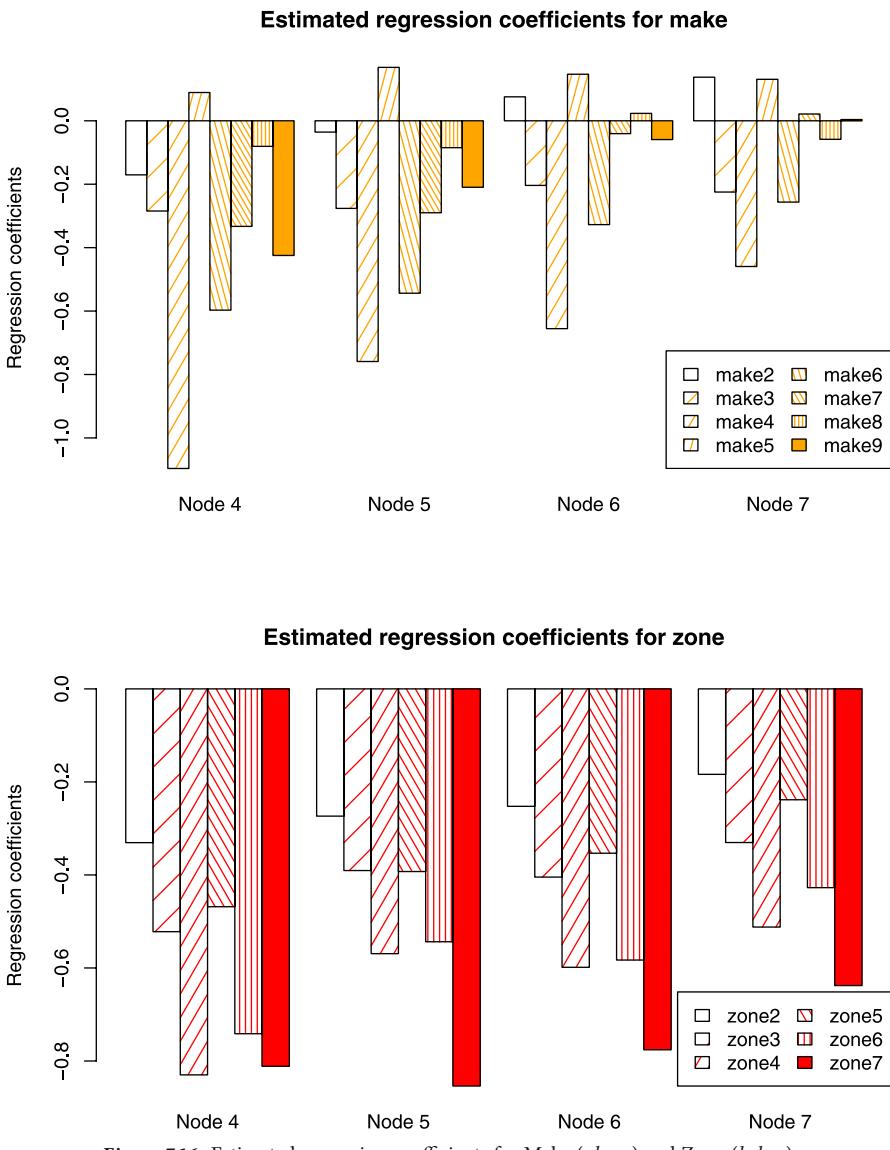


Figure 7.16. Estimated regression coefficients for Make (above) and Zone (below)

Because rate of change of log claim rate with respect to Bonus and Mileage class depends on the levels of Make and Zone, the best way to visualize the effects is to draw a contour plot of the fitted model for each combination of Make and Zone. This is done in Fig. 7.17 for four level combinations, those corresponding to the best and worst levels of Make and Zone. We see that claim rate is highest when Mileage class is 5, Bonus class is 1, Make is 5, and Zone is 1. The lowest claim rates occur for Make level 4 and Zone level 7, more or less independent of Mileage and Bonus class.

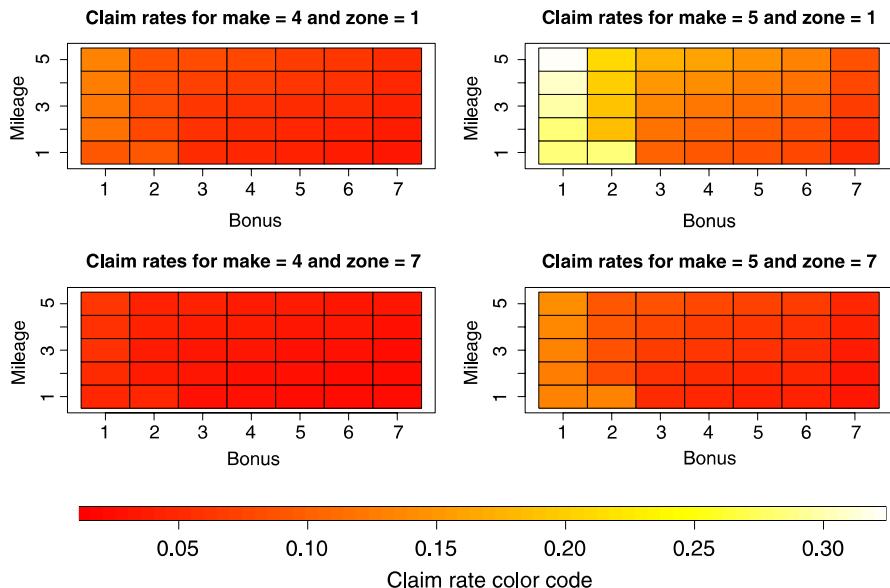


Figure 7.17. Estimated claim rates for selected values of Make and Zone

7.7 Conclusion

We have given four examples to illustrate the uses of GUIDE for building visualizable regression models. We contend that a model is best understood if it can be visualized. But in order to make effective use of current visualization techniques, namely, scatter and contour plots, we will often need to fit models to partitions of a dataset. Otherwise, we simply cannot display a model involving more than two predictor variables in a single 2-D graph. The data partitions, of course, should be chosen to build as parsimonious a model as possible. The GUIDE algorithm does this by finding partitions that break up curvature and interaction effects. As a result, it avoids splitting a partition on a predictor variable whose effect is already linear. Model parsimony as a whole is ensured by pruning, which prevents the number of partitions from being unnecessarily large.

After pruning is finished, we can be quite confident that most of the important effects of the predictor variables are confined within the one or two selected linear predictors. Thus it is safe to plot the data and fitted function in each partition and to draw conclusions from them. As our examples showed, such plots usually can tell us much more about the data than a collection of regression coefficients. An obvious advantage of 2-D plots is that they require no special training for interpretation. In particular, the goodness of fit of the model in each partition can be simply judged by eye instead of through a numerical quantity such as AIC.

The GUIDE computer program is available for Linux, Macintosh, and Windows computers from www.stat.wisc.edu/%7Eloeh/.

Acknowledgement. The author is grateful to a referee for comments that led to improvements in the presentation. This research was partially supported by grants from the US Army Research Office and the National Science Foundation.

References

- Akaike, H. (1973). Information theory and an extension of the maximum likelihood principle. in B. Petrov and F. Csàki (eds), *Second International Symposium on Information Theory*, Akadémia Kiadó, Budapest, pp. 267–281.
- Andrews, D.F. and Herzberg, A.M. (1985). *Data: A Collection of Problems from Many Fields for the Student and Research Worker*, Springer, New York.
- Belsley, D., Kuh, E. and Welsch, R. (1980). *Regression Diagnostics: Identifying Influential Data and Sources of Collinearity*, Wiley, New York.
- Breiman, L., Friedman, J., Olshen, R. and Stone, C. (1984). *Classification and Regression Trees*, Wadsworth, Belmont.
- Camden, M. (1989). The data bundle, New Zealand Statistical Association, Wellington.
- Cook, D. (1998). *Regression Graphics: Ideas for Studying Regression Through Graphics*, Wiley, New York.
- Friedman, L. and Wall, M. (2005). Graphical views of suppression and multicollinearity in multiple linear regression, *American Statistician*, 59:127–136.
- Gilley, O.W. and Pace, R. (1996). On the harrison and rubinfeld data, *Journal of Environmental Economics and Management*, 31:403–405.
- Hallin, M. and Ingenbleek, J.-F. (1983). The swedish automobile portfolio in 1977. A statistical study, *Scandinavian Actuarial Journal* pp. 49–64.
- Harrison, D. and Rubinfeld, D. (1978). Hedonic prices and the demand for clean air, *Journal of Environmental Economics and Management*, 5:81–102.
- Li, K.-C. (1991). Sliced inverse regression for dimension reduction (with discussion), *Journal of the American Statistical Association*, 86:316–342.
- Loh, W.-Y. (2002). Regression trees with unbiased variable selection and interaction detection, *Statistica Sinica*, 12:361–386.
- Miller, A. (2002). *Subset Selection in Regression*, 2nd edn, Chapman & Hall, London.

Structural Adaptive Smoothing III.8 **by Propagation–Separation** **Methods**

Jörg Polzehl, Vladimir Spokoiny

8.1	<i>Nonparametric Regression</i>	472
	Examples	472
	Local Modeling	473
8.2	<i>Structural Adaptation</i>	475
	Adaptive Weights Smoothing	476
	Choice of Parameters: Propagation Condition	477
8.3	<i>An Illustrative Univariate Example</i>	478
8.4	<i>Examples and Applications</i>	480
	Application 1: Adaptive Edge-Preserving Smoothing in 3-D	480
	Examples: Binary and Poisson Data	481
	Example: Denoising of Digital Color Images	483
	Example: Local Polynomial Propagation–Separation (PS) Approach	485
8.5	<i>Concluding Remarks</i>	489

Regression is commonly used to describe and analyze the relation between explanatory input variables X and one or multiple responses Y . In many applications such relations are too complicated to model with a parametric regression function. Classical nonparametric regression (see e.g., Fan and Gijbels, 1996; Wand and Jones, 1995; Loader, 1999; Simonoff, 1996) and varying coefficient models (see e.g., Hastie and Tibshirani, 1993; Fan and Zhang, 1999; Carroll et al., 1998; Cai et al., 2000), allow for a more flexible form. In this article we describe an approach that allows us to efficiently handle discontinuities and spatial inhomogeneities of the regression function in such models.

8.1

Nonparametric Regression

Let us assume that we have a random sample Z_1, \dots, Z_n of the form $Z_i = (X_i, Y_i)$. Every X_i is a vector of explanatory variables which determines the distribution of an observed response Y_i . Let the X_i 's be valued in the finite dimensional Euclidean space $\mathcal{X} = \mathbb{R}^d$ and the Y_i 's belong to $\mathcal{Y} \subseteq \mathbb{R}^q$. The explanatory variables X_i may quantify some experimental conditions, coordinates within an image, or a time. The response Y_i in these cases identifies the observed outcome of the experiment: the gray value or color at the given location and the value of a time series, respectively.

We assume that the distribution of each Y_i is determined by a finite dimensional parameter $\theta = \theta(X_i)$ which may depend on the value X_i of the explanatory variable.

8.1.1

Examples

We use the following examples to illustrate the situation.

1

Example 1: *homoscedastic nonparametric regression model* This model is specified by the regression equation $Y_i = \theta(X_i) + \varepsilon_i$ with a regression function θ and additive i.i.d. Gaussian errors $\varepsilon_i \sim \mathcal{N}(0, \sigma^2)$. We will use this model to illustrate the main properties of our algorithms in a univariate ($d = 1$) setting. The model also serves as a reasonable approximation to many imaging problems. Here the explanatory variables X_i define a two-dimensional ($d = 2$) or three-dimensional ($d = 3$) grid with observed gray values Y_i at each point.

2

Example 2: *inhomogeneous binary response model* Here Y_i is a Bernoulli random variable with parameter $\theta(X_i)$; that is, $P(Y_i = 1 | X_i) = \theta(X_i)$ and $P(Y_i = 0 | X_i) = 1 - \theta(X_i)$. This model occurs in classification. It is also adequate for binary images.

3

Example 3: *inhomogeneous Poisson model* Every Y_i follows a Poisson distribution with parameter $\theta = \theta(X_i)$, i.e., Y_i attains nonnegative integer values and

$P(Y_i = k \mid X_i) = \theta^k(X_i)e^{-\theta(X_i)}/k!$. Such a situation frequently occurs in low-intensity imaging, e.g., confocal microscopy and positron emission tomography. It also serves as an approximation of the density model, obtained by a binning procedure.

Example 4: *color images* In color images, Y_i denotes a vector of values in a three-dimensional color space at pixel coordinates X_i . A fourth component may code transparency information. The observed vectors Y_i can often be modeled as a multivariate Gaussian, i.e., $Y_i \sim N_3(\theta(X_i), \Sigma)$ with some unknown covariance Σ that may depend on θ . Additionally we will usually observe some spatial correlation.

Local Modeling

8.1.2

We now formally introduce our model. Let $\mathcal{P} = (P_\theta, \theta \in \Theta)$ be a family of probability measures on \mathcal{Y} where Θ is a subset of the real line \mathbb{R}^1 . We assume that this family is dominated by a measure P and denote $p(y, \theta) = dP_\theta/dP(y)$. We suppose that each Y_i is, conditionally on $X_i = x$, distributed with density $p(\cdot, \theta(x))$. The density is parameterized by some unknown function $\theta(x)$ on \mathcal{X} which we aim to estimate. A global parametric structure simply means that the parameter θ does not depend on the location; that is, the distribution of every “observation” Y_i coincides with P_θ for some $\theta \in \Theta$ and all i . This assumption reduces the original problem to the classical parametric situation and the well-developed parametric theory is applied here to estimate the underlying parameter θ . In particular, the maximum likelihood estimate $\tilde{\theta} = \tilde{\theta}(Y_1, \dots, Y_n)$ of θ , which is defined by the maximization of the log-likelihood

$$L(\theta) = \sum_{i=1}^n \log p(Y_i, \theta) \quad (8.1)$$

is root-n consistent and asymptotically efficient under rather general conditions.

Such a global parametric assumption is typically too restrictive. The classical non-parametric approach is based on the idea of localization: for every point x , the parametric assumption is only fulfilled locally in a vicinity of x . We therefore use a local model concentrated in some neighborhood of the point x .

The most general way to describe a local model is based on weights. Let, for a fixed x , a nonnegative weight $w_i = w_i(x) \leq 1$ be assigned to the observations Y_i at X_i , $i = 1, \dots, n$. When estimating the local parameter $\theta(x)$, every observation Y_i is used with the weight $w_i(x)$. This leads to the local (weighted) maximum likelihood estimate

$$\tilde{\theta}(x) = \arg \sup_{\theta \in \Theta} \sum_{i=1}^n w_i(x) \log p(Y_i, \theta). \quad (8.2)$$

Note that this definition is a special case of a more general local linear (polynomial) likelihood model when the underlying function θ is modelled linearly (polynomi-

ally) in x ; see e.g., Fan et al. (1998). However, our approach focuses on the choice of localizing weights in a data-driven way rather than on the method of local approximation of the function θ .

A common way to choose the weights $w_i(x)$ is to define them in the form $w_i(x) = K_{\text{loc}}(I_i)$ with $I_i = |\rho(x, X_i)/h|^2$ where h is a bandwidth, $\rho(x, X_i)$ is the Euclidean distance between x and the design point X_i , and K_{loc} is a *location kernel*. This approach is intrinsically based on the assumption that the function θ is smooth. It leads to a local approximation of $\theta(x)$ within a ball with some small radius h centered on the point x , see e.g., Tibshirani and Hastie (1987); Hastie and Tibshirani (1993); Fan et al. (1998); Carroll et al. (1998); Cai et al. (2000).

An alternative approach is termed *localization by a window*. This simply restricts the model to a subset (window) $U = U(x)$ of the design space which depends on x ; that is, $w_i(x) = \mathbf{1}(X_i \in U(x))$. Observations Y_i with X_i outside the region $U(x)$ are not used to estimate the value $\theta(x)$. This kind of localization arises, for example, in the regression tree approach, in change point estimation (see e.g., Müller, 1992; Spokoiny, 1998), and in image denoising (see Qiu, 1998; Polzehl and Spokoiny, 2003), among many other situations.

In our procedure we do not assume any special structure for the weights $w_i(x)$; that is, any configuration of weights is allowed. The weights are computed in an iterative way from the data. In what follows we identify the set $W(x) = \{w_1(x), \dots, w_n(x)\}$ and the local model in x described by these weights and use the notation

$$L(W(x), \theta) = \sum_{i=1}^n w_i(x) \log p(Y_i, \theta).$$

Then $\tilde{\theta}(x) = \arg \sup_{\theta} L(W(x), \theta)$. For simplicity we will assume the case where $\theta(x)$ describes the conditional expectation $E(Y|x)$ and the local estimate is obtained explicitly as

$$\tilde{\theta}(x) = \sum_i w_i(x) Y_i / \sum_i w_i(x). \quad (8.3)$$

The quality of the estimation heavily depends on the localizing scheme we selected. We illustrate this issue by considering kernel weights $w_i(x) = K_{\text{loc}}(|\rho(x, X_i)/h|^2)$ where the kernel K_{loc} is supported on $[0, 1]$. Then the positive weights $w_i(x)$ are concentrated within the ball of radius h at the point x . A small bandwidth h leads to a very strong localization. In particular, if the bandwidth h is smaller than the distance from x to the nearest neighbor, then the resulting estimate coincides with the observation at x . Increasing the bandwidth amplifies the noise reduction that can be achieved. However, the choice of a large bandwidth may lead to estimation bias if the local parametric assumption of a homogeneous structure is not fulfilled in the selected neighborhood.

The classical approach to solving this problem is based on a model selection idea. One assumes a given set of bandwidth candidates $\{h_k\}$, and one of them is selected in a data-driven way to provide the optimal quality of estimation. The global bandwidth selection problem assumes the same kernel structure of localizing schemes $w_i(x)$ for

all points x , and only one bandwidth h has to be specified. In the local model selection approach, the bandwidth h may vary with the point x . See Fan et al. (1998) for more details.

We employ a related but more general approach. We consider a family of localizing models, one per design point X_i , and denote them as $W_i = W(X_i) = \{w_{i1}, \dots, w_{in}\}$. Every W_i is built in an iterative data-driven way, and its support may vary from point to point. The method used to construct such localizing schemes is discussed in the next section.

Structural Adaptation

8.2

Let us assume that for each design point X_i the regression function θ can be well approximated by a constant within a local vicinity $U(X_i)$ containing X_i . This serves as our structural assumption.

Our estimation problem can now be viewed as consisting of two parts. In order to efficiently estimate the function θ in a design point X_i we need to describe a local model, i.e., to assign weights $W(X_i) = \{w_{i1}, \dots, w_{in}\}$. If we knew the neighborhood $U(X_i)$ via an oracle we would define the local weights as $w_{ij} = w_j(X_i) = I_{X_j \in U(X_i)}$ and use these weights to estimate $\theta(X_i)$. However, since θ and therefore $U(X_i)$ are unknown, the assignments will have to depend on the information on θ that we can extract from the observed data. If we have good estimates $\hat{\theta}_j = \hat{\theta}(X_j)$ of $\theta(X_j)$, we can use this information to infer the set $U(X_i)$ by testing the hypothesis

$$H : \theta(X_j) = \theta(X_i). \quad (8.4)$$

A weight w_{ij} can be assigned based on the value of a test statistic T_{ij} , assigning zero weights if $\hat{\theta}_j$ and $\hat{\theta}_i$ are significantly different. This provides us with a set of weights $W(X_i) = \{w_{i1}, \dots, w_{in}\}$ that determines a local model in X_i .

Given the local model we can then estimate our function θ at each design point X_i by (8.2).

We utilize both steps in an iterative procedure. We start with a very local model at each point X_i given by weights

$$w_{ij}^{(0)} = K_{\text{loc}}(I_{ij}^{(0)}) \quad \text{with} \quad I_{ij}^{(0)} = |X_i - X_j|/h^{(0)}. \quad (8.5)$$

The initial bandwidth $h^{(0)}$ is chosen very small. K_{loc} is a kernel function supported on $[-1, 1]$; i.e., weights vanish outside a ball $U_i^{(0)}$ of radius $h^{(0)}$ centered on X_i . We then iterate two steps: estimation and local model refinement. In the k th iteration new weights are generated as

$$w_{ij}^{(k)} = K_{\text{loc}}(I_{ij}^{(k)})K_{\text{stat}}(s_{ij}^{(k)}) \quad \text{with} \quad (8.6)$$

$$I_{ij}^{(k)} = |X_i - X_j|/h^{(k)} \quad \text{and} \quad s_{ij}^{(k)} = T_{ij}^{(k)}/\lambda. \quad (8.7)$$

The kernel function K_{stat} is monotonically nonincreasing over the interval $[0, \infty)$. The bandwidth h is increased by a constant factor with each iteration k . The test statistic for (8.4)

$$T_{ij}^{(k)} = N_i^{(k)} \mathcal{K}(\hat{\theta}_i^{(k-1)}, \hat{\theta}_j^{(k-1)}) \quad (8.8)$$

with $N_i = \sum_j w_{ij}$ is used to specify the penalty $s_{ij}^{(k)}$. This term effectively measures the statistical difference between the current estimates in X_i and X_j . In (8.8) the term $\mathcal{K}(\theta, \theta')$ denotes the Kullback–Leibler distance of the probability measures P_θ and $P_{\theta'}$.

Additionally, we can introduce a kind of memory into the procedure, which ensures that the quality of estimation will not be lost with the iterations. This basically means that we compare a new estimate $\tilde{\theta}_i^{(k)} = \tilde{\theta}^{(k)}(X_i)$ with the previous estimate $\hat{\theta}_i^{(k-1)}$ in order to define a memory parameter $\eta_i = K_{\text{mem}}(\mathbf{m}_i^{(k)})$ using a kernel function K_{mem} and

$$\mathbf{m}_i^{(k)} = \tau^{-1} \sum_j K_{\text{loc}}(\mathbf{l}_{ij}^{(k)}) \mathcal{K}(\tilde{\theta}_i^{(k)}, \hat{\theta}_i^{(k-1)}). \quad (8.9)$$

This leads to an estimate

$$\hat{\theta}_i^{(k)} = \eta_i \tilde{\theta}_i^{(k)} + (1 - \eta_i) \hat{\theta}_i^{(k-1)}. \quad (8.10)$$

8.2.1 Adaptive Weights Smoothing

We now formally describe the resulting algorithm.

Initialization: Set the initial bandwidth $h^{(0)}$, $k = 0$ and compute, for every i , the statistics

$$N_i^{(k)} = \sum_j w_{ij}^{(k)}, \quad \text{and} \quad S_i^{(k)} = \sum_j w_{ij}^{(k)} Y_j \quad (8.11)$$

and the estimates

$$\hat{\theta}_i^{(k)} = S_i^{(k)} / N_i^{(k)} \quad (8.12)$$

using $w_{ij}^{(0)} = K_{\text{loc}}(\mathbf{l}_{ij}^{(0)})$. Set $k = 1$ and $h^{(1)} = c_h^{(0)}$.

Adaptation: For every pair i, j , compute the penalties

$$\mathbf{l}_{ij}^{(k)} = |X_i - X_j| / h^{(k)}, \quad (8.13)$$

$$\mathbf{s}_{ij}^{(k)} = \lambda^{-1} T_{ij}^{(k)} = \lambda^{-1} N_i^{(k-1)} \mathcal{K}(\hat{\theta}_i^{(k-1)}, \hat{\theta}_j^{(k-1)}). \quad (8.14)$$

Now compute the weights $w_{ij}^{(k)}$ as

$$w_{ij}^{(k)} = K_{\text{loc}}(\mathbf{l}_{ij}^{(k)}) K_{\text{stat}}(\mathbf{s}_{ij}^{(k)})$$

and specify the local model by $W_i^{(k)} = \{w_{i1}^{(k)}, \dots, w_{in}^{(k)}\}$.

Local estimation: Now compute new local MLE estimates $\tilde{\theta}_i^{(k)}$ of $\theta(X_i)$ as

$$\tilde{\theta}_i^{(k)} = S_i^{(k)} / \tilde{N}_i^{(k)} \quad \text{with} \quad \tilde{N}_i^{(k)} = \sum_j w_{ij}^{(k)}, \quad S_i^{(k)} = \sum_j w_{ij}^{(k)} Y_j.$$

Adaptive control: Compute the memory parameter as $\eta_i = K_{\text{mem}}(\mathbf{m}_i^{(k)})$. Define

$$\hat{\theta}_i^{(k)} = \eta_i \tilde{\theta}_i^{(k)} + (1 - \eta_i) \hat{\theta}_i^{(k-1)} \quad \text{and}$$

$$N_i^{(k)} = \eta_i \tilde{N}_i^{(k)} + (1 - \eta_i) N_i^{(k-1)}$$

Stopping: Stop if $h^{(k)} \geq h_{\max}$, otherwise set $h^{(k)} = c_h h^{(k-1)}$, increase k by 1, and continue with the adaptation step.

Choice of Parameters: Propagation Condition

8.2.2

The proposed procedure involves several parameters. The most important one is the scale parameter λ in the statistical penalty s_{ij} . The special case $\lambda = \infty$ simply leads to a kernel estimate with bandwidth h_{\max} . We propose to choose λ as the smallest value satisfying a propagation condition. This condition requires that, if the local assumption is valid globally (i.e., $\theta(x) \equiv \theta$ does not depend on x), then with high probability the final estimate for $h_{\max} = \infty$ coincides at every point with the global estimate. More formally we request that, in this case, for each iteration k ,

$$E|\hat{\theta}^{(k)}(X) - \check{\theta}^{(k)}(X)| < \alpha E|\check{\theta}^{(k)}(X) - \theta| \quad (8.15)$$

for a specified constant $\alpha > 0$. Here

$$\check{\theta}^{(k)}(X_i) = \sum_j K_{\text{loc}}(I_{ij}^{(k)}) Y_j / \sum_j K_{\text{loc}}(I_{ij}^{(k)}) \quad (8.16)$$

denotes the nonadaptive kernel estimate employing the bandwidth $h^{(k)}$ from step k . The value λ provided by this condition does not depend on the unknown model parameter θ and can therefore be approximately found by simulation. This allows us to select default values for λ depending on the specified family of the probability distribution $\mathcal{P} = (P_\theta, \theta \in \Theta)$. Default values for λ in the examples below are selected for a value of $\alpha = 0.2$.

The second parameter of interest is the maximal bandwidth h_{\max} , which controls both the numerical complexity of the algorithm and the smoothness within homogeneous regions.

The scale parameter τ in the memory penalty \mathbf{m}_i can also be chosen to meet the propagation condition (8.15). The special case $\tau = \infty$ turns off the adaptive control step.

Additionally we specify a number of parameters and kernel functions that have less influence on the resulting estimates. As a default, the kernel functions are chosen as $K_{\text{loc}}(x) = K_{\text{mem}}(x) = (1 - x^2)_+$ and $K_{\text{stat}}(x) = e^{-x} I_{x < 5}$. If the design is on

a grid (e.g., for images), the initial bandwidth $h^{(0)}$ is chosen as the distance between neighboring pixels. The bandwidth is increased after each iteration by a default factor $c_h = 1.25^{1/d}$.

An Illustrative Univariate Example

We use a simple example to illustrate the behavior of the algorithm. The data in the upper left of Fig. 8.1 follow a univariate regression model

$$Y_i = \theta(X_i) + \varepsilon_i. \quad (8.17)$$

The unknown parameter (i.e., the regression function θ) is piecewise constant, the errors ε_i are i.i.d. $N(0,1)$, and the observed $X_i = i$ form a univariate grid. In this situation the statistical penalty takes the form

$$\mathbf{s}_{ij}^{(k)} = \frac{N_i^{(k-1)}}{2\sigma^2\lambda} (\hat{\theta}_i^{(k-1)} - \hat{\theta}_j^{(k-1)})^2 \quad (8.18)$$

where $\sigma^2 = 1$ denotes the variance of the errors. A robust estimate of the variance is obtained from the data using the interquartile range (IQR) as

$$\hat{\sigma}^2 = (IQR(\{Y_{i+1} - Y_i\}_{i=1,\dots,n-1})/1.908)^2 \quad (8.19)$$

and this is used as a plug-in for σ^2 . The propagation condition (8.15) suggests a value of $\lambda = 4.7$. We employ a value of $\tau = \infty$, disabling the adaptive control step.

We have four regions, differing in size and contrast, where the function θ is constant. The regression function is displayed as a black line in the upper right of Fig. 8.1.

The lower part of Fig. 8.1 illustrates the evolution of the weights w_{ij} as the number of iterations increases. The horizontal and vertical axes correspond to indices i and j , respectively. The upper row provides $K_{\text{loc}}(\mathbf{l}_{ij}^{(k)})$ for iterations $k = 0$ ($h = 1$), $k = 7$ ($h = 5$), $k = 13$ ($h = 18$) and $k = 23$ ($h = 169$). The central row shows the corresponding values $K_{\text{stat}}(\mathbf{s}_{ij}^{(k)})$. The grayscale ranges from black for 0 to white for 1. The weights $w_{ij}^{(k)}$ (lower row) used in the algorithm are the products of both terms.

The left column corresponds to the initialization step. Here the location penalty effectively restricts the local model in X_i to the point X_i itself. If computed, the stochastic penalty would contain some weak information about the structure of the regression function. When we reach step $k = 7$, the location penalty allows for positive weights for up to 9 observations, and therefore less variable estimates. At this stage the test (8.4) shows a significant difference between estimates at points within the third homogeneous interval and estimates at locations outside this interval. This is reflected in the statistical penalty and therefore the weights. In step $k = 13$ the second interval is also clearly identified. The last column, referring to the 23rd iteration and a final bandwidth of $h = 169$, shows the final situation, where the statistical penalty

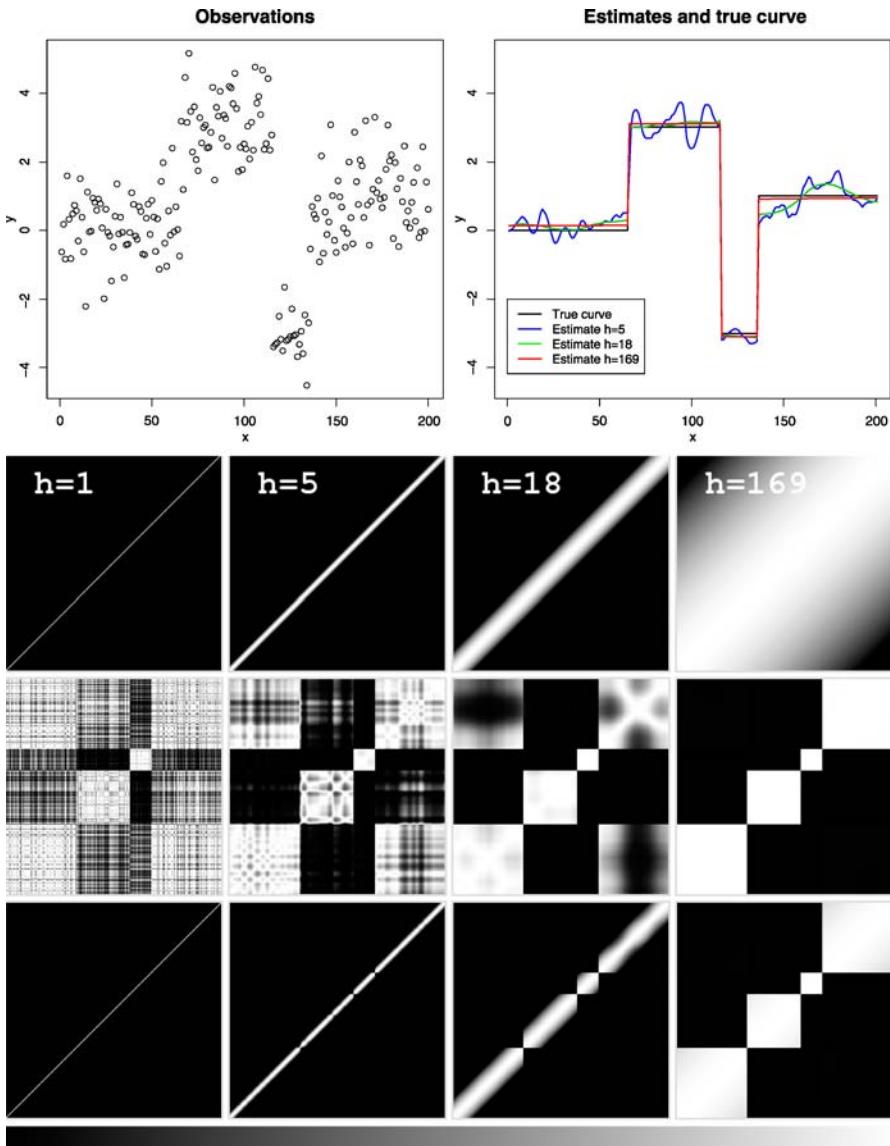


Figure 8.1. Adaptive weights smoothing for a simple univariate regression problem: data (*upper left*), regression function, and estimates $\hat{\theta}^{(k)}$ for $k = 7$ ($h = 5$), $k = 13$ ($h = 18$) and $k = 23$ ($h = 169$) (*upper right*). The lower part displays the contributions from the location penalty (*top row*) and the stochastic penalty (*middle row*) to the weights (*bottom row*) w_{ij} in iteration steps $k = 0, 7, 13$ and 23 (*columns*). A linear grayscale is provided at the bottom

reflects all of the information about the structure and determines the weights. The influence of the location penalty has almost vanished.

What we observe during the iteration process is the unrestricted propagation of weights within homogeneous regions. Two regions with different values of the parameter are separated as values of the statistical penalty s_{ij} increase with decreasing variance of the estimates $\hat{\theta}_i$ and $\hat{\theta}_j$ and a large enough contrast $|\theta(X_i) - \theta(X_j)|^2$. The iteration k at which this occurs depends on the sizes of the homogeneous regions, i.e., the potential variance reduction, and the contrast.

The upper right plot in Fig. 8.1 additionally displays the intermediate estimates $\hat{\theta}^{(k)}$, $k = 7, 13, 23$ corresponding to the weighting schemes illustrated.

8.4 Examples and Applications

We now provide a series of examples for adaptive weights smoothing in various setups.

8.4.1 Application 1: Adaptive Edge-Preserving Smoothing in 3-D

The algorithm described in Sect. 8.2.1 is essentially dimension-free. It can easily be applied to reconstruct 2-D and 3-D images. We illustrate this using a 3-D MR image of a head. The upper left image in Fig. 8.2 shows the 130th slice of the noisy data cube, consisting of $256 \times 192 \times 256$ voxels. The image is modeled as

$$Y_i = \theta(X_i) + \varepsilon_i, \quad (8.20)$$

where X_i are coordinates on the 3-D grid and the errors ε_i are again assumed to be i.i.d. Gaussian with unknown variance σ^2 . The parameter of interest $\theta(X_i)$ describes a tissue-dependent underlying gray value at voxel X_i . There is special interest in using these images to identify tissue borders. Denoising, preferably using an edge-preserving or edge-enhancing filter, is a prerequisite step here.

We apply the AWS algorithm from Sect. 8.2.1 using a maximal bandwidth $h_{\max} = 5$. The error variance is estimated from the data. The default value of λ provided by condition (8.15) for smoothing in 3-D with Gaussian errors is $\lambda = 5.9$. The upper right image provides the resulting reconstruction. Note that in the smoothed image the noise is removed while the detailed structure corresponding to tissue borders is preserved. Some deterioration of the image is caused by the structural assumption of a local constant model. This leads to some flattening where $\theta(X_i)$ is smooth.

In the bottom row of Fig. 8.2, we provide the absolute values obtained after applying a Laplacian filter to the original noisy image and to the reconstruction obtained by AWS, respectively. We observe an enhancement of the tissue borders.

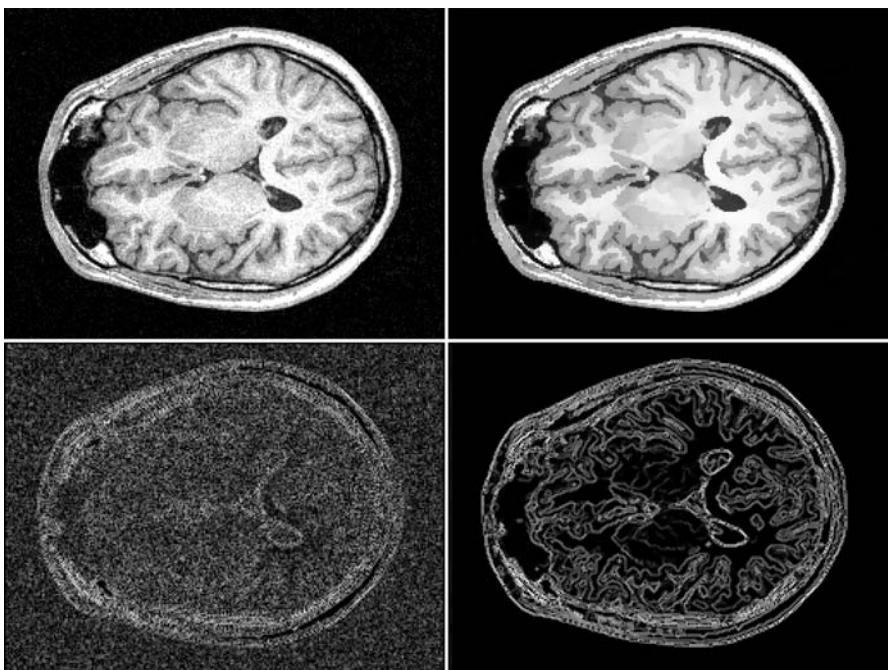


Figure 8.2. 3-D Magnetic resonance imaging (MRI): Slice 130 from a 3-D MR image (*upper left*) and its 3-D reconstruction by AWS (*upper right*). The *bottom row* shows the result of applying an edge detection filter to both images

Examples: Binary and Poisson Data

8.4.2

For non-Gaussian data, the stochastic penalty s_{ij} takes a different form in (8.14). The definition is based on the Kullback–Leibler distance \mathcal{K} between the probability measures $P_{\hat{\theta}_i}$ and $P_{\hat{\theta}_j}$. For binary data this leads to

$$s_{ij}^{(k)} = \frac{N_i^{(k-1)}}{\lambda} \left(\hat{\theta}_i^{(k-1)} \log \frac{\hat{\theta}_i^{(k-1)}}{\hat{\theta}_j^{(k-1)}} + (1 - \hat{\theta}_i^{(k-1)}) \log \frac{1 - \hat{\theta}_i^{(k-1)}}{1 - \hat{\theta}_j^{(k-1)}} \right) \quad (8.21)$$

while for Poisson data we get

$$s_{ij}^{(k)} = \frac{N_i^{(k-1)}}{\lambda} \left(\hat{\theta}_i^{(k-1)} \log \frac{\hat{\theta}_i^{(k-1)}}{\hat{\theta}_j^{(k-1)}} - \hat{\theta}_i^{(k-1)} + \hat{\theta}_j^{(k-1)} \right). \quad (8.22)$$

In both cases a special problem occurs. If the estimates $\hat{\theta}_i$ or $\hat{\theta}_j$ attain a value at the boundary of the parameter space, i.e., 0 or 1 for binary data or 0 in the case of Poisson data, then the Kullback–Leibler distance between the probability measures $P_{\hat{\theta}_i}$ and $P_{\hat{\theta}_j}$ will equal ∞ . Such a situation can be avoided by modifying the algorithm. One solution is to initialize the estimates with the value obtained by the global estimate

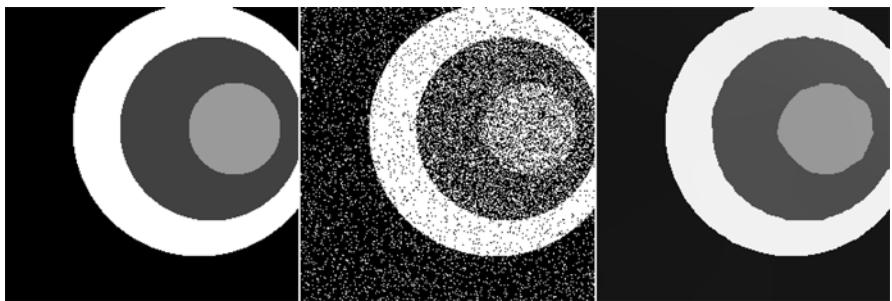


Figure 8.3. Binary images: artificial image containing four circles with different gray values (*left*), binary image generated by Bernoulli experiments with pointwise probabilities proportional to the gray values in the left image (*center*) and reconstructed image of pointwise probabilities (*right*)

and to replace the estimate $\hat{\theta}_j^{(k-1)}$ in (8.21, 8.22) by $\hat{\theta}_{ij}^{(k-1)} = (1 - 0.5/N_i^{(k-1)})\hat{\theta}_j^{(k-1)} + 0.5/N_i^{(k-1)}\hat{\theta}_i^{(k-1)}$ for all subsequent iteration steps.

We use a simple artificial example to demonstrate the performance of the procedure. We start with the image displayed on the left of Fig. 8.3. The image of size 256×256 is composed of four regions with distinct gray values. The central image is generated by pixelwise Bernoulli experiments with probabilities of 0.08, 0.3, 0.6 and 0.94, respectively, for the four regions. The image on the right of Fig. 8.3 provides the reconstruction obtained by AWS using a maximal bandwidth $h_{\max} = 100$. The value of λ selected by our propagation condition is $\lambda = 5.9$.

The noisy image on the left of Fig. 8.4 is constructed using the same image structure. Now each gray value is a Poisson count of intensity 0.4, 1.5, 3 and 4.7, depending on the location of the pixel within the image. The right image again provides the reconstruction. A maximal bandwidth $h_{\max} = 50$ and the value $\lambda = 5.4$ provided by the propagation condition (8.15) for 2-D Poisson images are used.

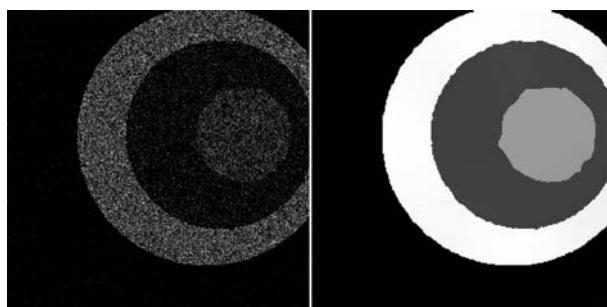


Figure 8.4. Poisson images: image generated by Poisson experiments with pointwise intensities proportional to the gray values in the left image of Fig. 8.3 (*left*) and reconstructed image of pointwise intensities (*right*)

Note that the underlying structure is completely recovered and therefore near-optimal estimates of the probabilities and intensities are obtained for both our binary and our Poisson image.

Example: Denoising of Digital Color Images

8.4.3

In digital color images, the information in each pixel consists of a vector of three values. Each value is an intensity in one channel of a three-dimensional color space, usually the RGB space. Additionally, each pixel may carry some transparency information. Ideally the image is recorded in RAW format (minimally processed data from the image sensor of a digital camera, see Wikipedia RAW (2006)), and then transformed to TIFF to avoid artifacts caused by lossy image compression and discretization to a low number of color values.

If the image was recorded under bad light conditions, using a high sensor sensitivity, such images can carry substantial noise. This noise is usually spatially correlated (i.e., colored). We also observe a correlation between the noise components in the three RGB channels. RGB is an additive color model in which red, green and blue light are combined in various ways to reproduce other colors; see Wikipedia RGB (2006) or Gonzales and Woods (2001).

An appropriate model for describing such a situation is given by

$$Y_{i_h, i_v} = \theta(X_i) + \varepsilon_{i_h, i_v}, \quad (8.23)$$

where the components of $X_i = (i_h, i_v)$ are the horizontal and vertical image coordinates. Y_{i_h, i_v} , $\theta(X_i)$ and ε_{i_h, i_v} take values in R^3 . The errors follow a distribution with $E\varepsilon_{i_h, i_v} = 0$, $\text{Var } \varepsilon_{i_h, i_v} = \Sigma$ and $E\varepsilon_{i_h, i_v}^c \varepsilon_{i_h+1, i_v}^c = E\varepsilon_{i_h, i_v}^c \varepsilon_{i_h, i_v+1}^c = \rho$ for each color channel c . The covariance matrix Σ may vary with the value of θ_{i_h, i_v} .

The algorithm from Sect. 8.2.1 can be applied in this situation with a statistical penalty

$$s_{ij}^{(k)} = \frac{N_i^{(k-1)}}{2\lambda} (\hat{\theta}_i^{(k-1)} - \hat{\theta}_j^{(k-1)})^\top \Sigma^{-1} (\hat{\theta}_i^{(k-1)} - \hat{\theta}_j^{(k-1)}). \quad (8.24)$$

The model can often be simplified by transforming the image to a suitable color space. We observe that a transformation to the YUV space decorrelates the noise between channels, so that a diagonal form of Σ seems appropriate under such a transformation. The YUV model defines a color space in terms of one luminance and two chrominance components, see Wikipedia YUV (2006) or Gonzales and Woods (2001). In this case, error variance can be estimated separately in the three color channels, accounting for the spatial correlation.

Figures 8.5 and 8.6 provide an example. The upper left image was obtained by deteriorating a digital image showing the Concert Hall at the Gendarmenmarkt in Berlin. The image resolution is 1600×1200 pixels.

The original image was transformed from RGB into YUV space. In YUV space, the values in the three channels are scaled to fall into the range $(0, 1)$, $(-0.24, 0.19)$ and $(-0.17, 0.46)$, respectively. In each YUV channel colored noise with $\rho = .36$ and



Figure 8.5. Color images: image of the Concert Hall at Gendarmenmarkt in Berlin, Germany. The image has been deteriorated by colored noise in all color channels (*upper left*). Shown are MAE-optimal reconstructions by AWS (*upper right*) and nonadaptive kernel smoothing (*lower left*). The *lower right* image illustrates for each pixel X_i the sum of weights $N_i = \sum_j w_{ij}$ that arises for the final estimate. Luminance images, see online version for colors



Figure 8.6. Color images: detail from the images in Fig. 8.5, noisy original (*left*), AWS reconstruction (*center*) and kernel smoothing (*right*). See online version for colors

Table 8.1. MAE and MSE in RGB space for the images in Fig. 8.5 and Fig. 8.9

	Noisy image	AWS reconstruction	Kernel smoothing	local quadratic PS
MAE	3.62×10^{-2}	1.91×10^{-2}	2.25×10^{-2}	1.70×10^{-2}
MSE	2.06×10^{-3}	8.34×10^{-4}	1.12×10^{-3}	6.03×10^{-4}

standard deviations of $\sigma = 0.08, 0.01$ and 0.012 , respectively, was added. The resulting noisy image, in RGB space, is shown in the upper left of Fig. 8.5. The upper right image shows the reconstruction obtained using our procedure, using a maximal bandwidth $h_{\max} = 6$. It is assumed that the spatial correlation $\rho = 0.36$ is known. The error variance is estimated from the image, taking the spatial correlation into account. The statistical penalty selected by the propagation condition (8.15) for color images with spatially independent noise is $\lambda = 6.90$. This parameter is corrected for the effect of spatial correlation at each iteration.

Each pixel X_i in the lower right image contains the value N_i ; that is, the sum of the weights defining the local model in X_i , for the last iteration. We can clearly see how the algorithm adapts to the structure in the image, effectively using a large local vicinity of X_i if the pixel belongs to a large homogeneous region and very small local models if the pixel X_i belongs to a very detailed structure.

Finally, we also provide the results from the corresponding nonadaptive kernel smoother (i.e., with $\lambda = \infty$ and a bandwidth of $h = 3.1$) for comparison at the lower left of Fig. 8.5. The bandwidth was chosen to provide the minimal mean absolute error. Table 8.1 provides the mean absolute error (MAE) and the mean squared error (MSE) for the three images in Fig. 8.5.

Figure 8.6 provides a detail, with a resolution of 340×545 pixels, from the noisy original (left), the AWS reconstruction (center) and the image obtained by nonadaptive kernel smoothing. The AWS reconstruction produces a much-enhanced image at the cost of flattening some smooth areas due to its local constant approximation. On the other hand, the nonadaptive kernel smoother suffers from a bad compromise between variance reduction and introduction of blurring, or bias, near edges.

Example: Local Polynomial Propagation–Separation (PS) Approach

8.4.4

Models (8.17) and (8.23) assume that the gray or color value is locally constant. This assumption is essentially used in the form of the stochastic penalty s_{ij} . The effect can be viewed as a regularization in the sense that in the limit for $h_{\max} \rightarrow \infty$, the reconstructed image is forced to a locally constant gray value or color structure even if the true image is locally smooth. This is clearly apparent in the detailed reconstruction in the center of Fig. 8.6, where the sculpture looks particularly cartoon-like. Such effects can be avoided if a local polynomial structural assumption is employed. Due to the increased flexibility of such models, this comes at the price of a decreased sensitivity to discontinuities.

The propagation–separation approach from Polzehl and Spokoiny (2004) assumes that within a homogeneous region containing $X_i = (i_h, i_v)$, i.e., for $X_j \in U(X_i)$, the gray value or color Y_{j_h, j_v} can be modeled as

$$Y_{j_h, j_v} = \theta(X_i)^\top \Psi(j_h - i_h, j_v - i_v) + \varepsilon_{j_h, j_v}, \quad (8.25)$$

where the components of $\Psi(\delta_h, \delta_v)$ contain values of basis functions

$$\psi_{m_1, m_2}(\delta_h, \delta_v) = (\delta_h)^{m_1} (\delta_v)^{m_2} \quad (8.26)$$

for integers $m_1, m_2 \geq 0$, $m_1 + m_2 \leq p$ and some polynomial order p . For a given local model $W(X_i)$, estimates of $\theta(X_i)$ are obtained by local least squares as

$$\tilde{\theta}(X_i) = B_i^{-1} \sum_j w_{ij} \Psi(j_h - i_h, j_v - i_v) Y_{j_h, j_v}, \quad (8.27)$$

with

$$B_i = \sum_j w_{ij} \Psi(j_h - i_h, j_v - i_v) \Psi(j_h - i_h, j_v - i_v)^\top. \quad (8.28)$$

The parameters $\theta(X_i)$ are defined with respect to a system of basis functions centered on X_i . Parameter estimates $\hat{\theta}(X_{j,i})$ in the local model $W(X_j)$ with respect to basis functions centered at X_i can be obtained by a linear transformation from $\hat{\theta}(X_j)$, see Polzehl and Spokoiny (2004). At iteration k , a statistical penalty can now be defined as

$$s_{ij}^{(k)} = \frac{1}{\lambda 2\sigma^2} (\hat{\theta}^{(k-1)}(X_i) - \hat{\theta}^{(k-1)}(X_{j,i}))^\top B_i (\hat{\theta}^{(k-1)}(X_i) - \hat{\theta}^{(k-1)}(X_{j,i})). \quad (8.29)$$

In a similar way, a memory penalty is introduced as

$$m_{ij}^{(k)} = \frac{1}{\tau 2\sigma^2} (\tilde{\theta}^{(k)}(X_i) - \hat{\theta}^{(k-1)}(X_i))^\top \tilde{B}_i^{(k)} (\tilde{\theta}^{(k)}(X_i) - \hat{\theta}^{(k-1)}(X_i)) \quad (8.30)$$

where \tilde{B}_i is constructed like B_i , employing location weights $K_I(l_{ij}^{(k)})$. The main parameters λ and τ are again chosen by a propagation condition requiring the free propagation of weights in the specified local polynomial model. A detailed description and discussion of the resulting algorithm and corresponding theoretical results can be found in Polzehl and Spokoiny (2004).

We use an artificial example to illustrate the behavior of the resulting algorithm. The left image in Fig. 8.7 contains gray values

$$f(x, y) = 0.5 [1 + \text{sign}(x^2 - y^2) \{ \sin(7\phi) \mathbf{1}_{\{r \geq 0.5\}} + \sin(\pi r/2) \mathbf{1}_{\{r < 0.5\}} \}]$$

with $x = i/127.5 - 1$, $y = j/127.5 - 1$, $r = \sqrt{x^2 + y^2}$ and $\phi = \arcsin(x/r)$ in locations $i, j = 0, \dots, 255$. The image is piecewise smooth with sharp discontinuities along diagonals and a discontinuity of varying strength around a circle. The noisy image on the right of Fig. 8.7 contains additive white noise with standard deviation $\sigma = .2$.

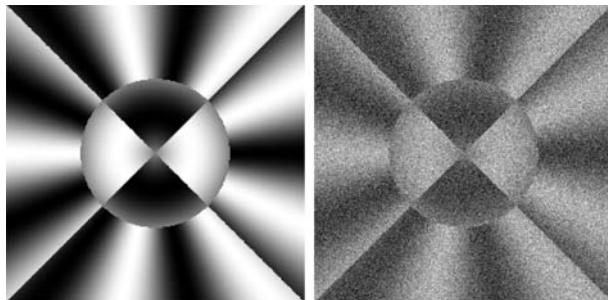


Figure 8.7. Artificial locally smooth image, original (*left*) and noisy version (*right*)

The upper row of Fig. 8.8 provides results obtained by (nonadaptive) kernel, local linear and local quadratic smoothing (from left to right), employing mean absolute error (MAE) optimal bandwidths. The second row gives the reconstructions obtained by the corresponding AWS and propagation–separation approaches, again with MAE optimal maximal bandwidths h_{\max} . The mean absolute error and mean squared error (MAE) for all six reconstructions together with the employed values of h or h_{\max} are shown in Table 8.2. Adaptive control was not used ($\tau = \infty$) for the adaptive procedures. The local constant AWS reconstruction, although clearly an improvement on all nonadaptive methods, exhibits clear artifacts resulting from the inappropriate structural assumption used. Also, the quality of this result heavily depends on the chosen value of h_{\max} . Both local linear and local quadratic PS allow for more flexibility when describing smooth changes of gray values. This enables us to use much larger maximal bandwidths, and therefore to obtain more variance reduction without compromising the separation of weights at the edges. The best results are obtained by the local quadratic propagation–separation algorithm. The bottom row of Fig. 8.8 again illustrates the sum of weights in each pixel generated in the final step of the adaptive procedure.

We now revisit the example from Fig. 8.5. The reconstruction in Fig. 8.9 is obtained by applying the local quadratic propagation–separation algorithm with parameters adjusted for the spatial correlation present in the noisy image. The maximal bandwidth used is $h_{\max} = 20$. The statistical penalty selected by the propagation condition for color images with spatially independent noise is $\lambda = 35$. This parameter is again corrected for the effect of spatial correlation at each iteration. Both the MAE

Table 8.2. MAE optimal value of h , MAE and MSE for the images in Fig. 8.8

	local constant		local linear		local quadratic	
	nonadapt.	AWS ($p = 0$)	nonadapt.	PS ($p = 1$)	nonadapt.	PS ($p = 2$)
h, h_{\max}	5.5	6	5.5	15	10	25
MAE $\times 10^2$	3.27	3.02	3.30	2.10	3.44	1.88
MSE $\times 10^3$	3.52	2.17	3.52	1.64	3.52	1.64

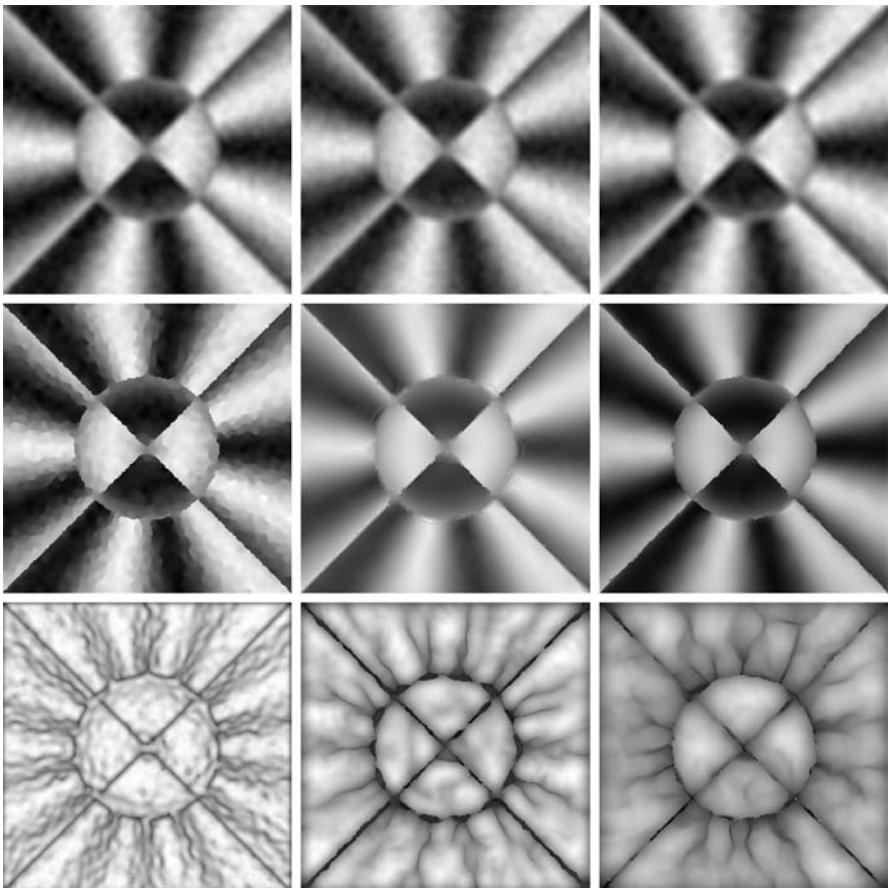


Figure 8.8. Reconstructions of the noisy image from Fig. 8.7. *Upper row:* nonadaptive smoothing; *central row:* structurally adaptive reconstructions; *bottom row:* pointwise sum of weights used in the structurally adaptive reconstructions. *Left column:* local constant smoothing, e.g., kernel smoothing and AWS; *central column:* local linear models; *right column:* local quadratic models. All reconstructions use MAE optimal values for the bandwidth or maximal bandwidth, respectively

and the MSE for the reconstruction are significantly smaller than those obtained for local constant AWS, see Table 8.2.

The detailed view offered by Fig. 8.10 allows for a more precise judgment of the image quality for one of the most structured regions in the image. We provide the same segment of size 300×300 pixels of the original image, its noisy version, and both the local constant and local quadratic reconstructions. The local constant reconstruction generally provides more contrast, at the cost of introducing artifacts into the smooth regions, such as the sculpture. Local quadratic PS gives a better result with respect to optical impression, MAE and MSE.



Figure 8.9. Local quadratic reconstruction of the noisy image from Fig. 8.5. See online version for colors

Concluding Remarks

8.5

In this chapter we have presented a novel adaptive smoothing procedure that has some remarkable properties and a potentially wide range of applications. We have illustrated, using a variety of examples, that the approach is essentially dimension-free; it works in 1-D, 2-D and even 3-D situations. It automatically recovers regions of homogeneity with respect to a local constant or local polynomial model. As a consequence, borders between homogeneous regions are preserved and even enhanced. If the specified local model allows for a good approximation of the unknown function θ , this also permits a significant reduction in variance without introduction of bias.

In areas where the function θ is smooth, the procedure based on a local constant model is, for large h_{\max} , likely to produce a local constant approximation. Nevertheless, such a bias introduced at a certain iteration k will be balanced by the variability of the estimates at this iteration. The effect can also be avoided by choosing an appropriate value of h_{\max} .

In Polzehl and Spokoiny (2006), theoretical results are obtained for the case where \mathcal{P} is a one-parameter exponential family. This includes results on the propagation, or free extension, of weights within interior sets of homogeneous regions and rates of estimation in regions where the parameter function θ is smooth. Conditions are given for the separation of two homogeneous regions depending on their sizes and con-



Figure 8.10. *Upper row:* detail of the original image and the same detail from the noisy version. *Bottom row:* local constant and local quadratic reconstructions. See online version for colors

trast. It was also shown that, up to a constant, the procedure retains the best quality of estimation reached within the iteration process at any point. Related results for the local polynomial propagation–separation approach can be found in Polzehl and Spokoiny (2004).

In the form presented here, the procedure is, for dimension $d > 1$, entirely isotropic. It can be significantly improved by introducing anisotropy adaptively (i.e., depending on the information about θ obtained in the iterative process) in the definition of the location penalty.

A reference implementation for the adaptive weights procedure described in Sect. 8.2.1 is available as a package (`aws`) from the R-Project for Statistical Computing (R Development Core Team, 2005) at <http://www.r-project.org/>. Image processing is implemented in R-package `adimpro`, see Polzehl and Tabelow (2007).

References

- Cai, Z., Fan, J. and Li, R. (2000). *Efficient estimation and inference for varying coefficients models.*, J. Amer. Statist. Assoc. 95:888–902.
- Cai, Z., Fan, J. and Yao, Q. (2000). *Functional-coefficient regression models for nonlinear time series*, J. Amer. Statist. Assoc. 95:941–956.
- Carroll, R.J., Ruppert, D. and Welsh, A.H. (1998). *Nonparametric estimation via local estimating equation*, J. Amer. Statist. Assoc. 93:214–227.
- Fan, J., Farmen, M. and Gijbels, I. (1998). *Local maximum likelihood estimation and inference*, J. Roy. Statist. Soc. Ser. B 60:591–608.
- Fan, J. and Gijbels, I. (1996). *Local polynomial modelling and its applications*, Chapman & Hall, London.
- Fan, J. and Zhang, W. (1999). *Statistical estimation in varying coefficient models*, Ann. Statist. 27:1491–1518.
- Gonzales, R.C., and Woods, R.E. (2001). *Digital image processing*, 2nd ed., Prentice Hall, Upper Saddle River, NJ.
- Hastie, T.J. and Tibshirani, R.J. (1993). *Varying-coefficient models (with discussion)*, J. Roy. Statist. Soc. Ser. B 55:757–796.
- Loader, C. (1999). *Local regression and likelihood*, Springer, New York.
- Müller, H. (1992). *Change-points in nonparametric regression analysis*, Ann. Statist. 20:737–761.
- Polzehl, J. and Spokoiny, V. (2000). *Adaptive weights smoothing with applications to image restoration*, J. Roy. Statist. Soc. Ser. B 62:335–354.
- Polzehl, J. and Spokoiny, V. (2001). *Functional and dynamic magnetic resonance imaging using vector adaptive weights smoothing*, J. Roy. Statist. Soc. Ser. C 50:485–501.
- Polzehl, J. and Spokoiny, V. (2003). *Image denoising: pointwise adaptive approach*, Ann. Statist. 31:30–57.
- Polzehl, J. and Spokoiny, V. (2004). *Spatially adaptive regression estimation: Propagation-separation approach*, Preprint 998, WIAS.
- Polzehl, J. and Spokoiny, V. (2006). *Local likelihood modeling by adaptive weights smoothing*, Probab. Theor. Relat. Fields 12:335–362.
- Polzehl, J. and Tabelow, K. (2007). *Adaptive Smoothing of Digital Images: The R Package adimpro*, Journal of Statistical Software, 19(1).
- Qiu, P. (1998). *Discontinuous regression surface fitting*, Ann. Statist. 26:2218–2245.
- R Development Core Team (2005). *R: A language and environment for statistical computing*, R Foundation for Statistical Computing, Vienna, Austria, ISBN 3-900051-07-0.
- Simonoff, J. (1996). *Smoothing methods in statistics*, Springer, New York.
- Spokoiny, V. (1998). *Estimation of a function with discontinuities via local polynomial fit with an adaptive window choice*, Ann. Statist. 26:1356–1378.
- Tibshirani, R. and Hastie, T.J. (1987). *Local likelihood estimation*, J. Amer. Statist. Assoc. 82:559–567.
- Wikimedia Foundation (2006). *Wikipedia: RAW image format*, [Online; accessed 2006-03-21], http://en.wikipedia.org/wiki/RAW_image_format.

- Wikimedia Foundation (2006). *Wikipedia: RGB*, [Online; accessed 2006-03-21],
<http://en.wikipedia.org/wiki/RGB>.
- Wikimedia Foundation (2006). *Wikipedia: YUV*, [Online; accessed 2006-03-21],
<http://en.wikipedia.org/wiki/YUV>.
- Wand, M.P. and Jones, M.C. (1995). *Kernel smoothing*, Chapman & Hall, London.

Smoothing Techniques for Visualisation

III.9

Adrian W. Bowman

9.1	<i>Introduction</i>	494
9.2	<i>Smoothing in One Dimension</i>	496
9.3	<i>Smoothing in Two Dimensions</i>	502
9.4	<i>Additive Models</i>	507
9.5	<i>Discussion</i>	511

Introduction

Graphical displays are often constructed to place principal focus on the individual observations in a dataset, and this is particularly helpful in identifying both the typical positions of datapoints and unusual or influential cases. However, in many investigations, principal interest lies in identifying the nature of underlying trends and relationships between variables, and so it is often helpful to enhance graphical displays in ways which give deeper insight into these features. This can be very beneficial both for small datasets, where variation can obscure underlying patterns, and large datasets, where the volume of data is so large that effective representation inevitably involves suitable summaries.

These issues are particularly prominent in a regression setting, where it is the nature of the relationships between explanatory variables and the mean value of a response which is the focus of attention. Nonparametric smoothing techniques are extremely useful in this context as they provide an estimate of the underlying relationship without placing restrictions on the shape of the regression function, apart from an assumption of smoothness.

This is illustrated in Fig. 9.1, where the left-hand panel displays a scatterplot of data collected by the Scottish Environment Protection Agency on the level of dissolved oxygen close to the start of the Clyde estuary. Data from a substantial section of the River Clyde are analysed in detail by McMullan et al. (2005), who give the background details. Water samples have been taken at irregular intervals over a long period. The top left-hand panel plots the data against time in years. The large amount of variation in the plot against year makes it difficult to identify whether any underlying trend is present. The top right-hand panel adds a smooth curve to the plot, estimating the mean value of the response as a function of year. Some indication of improvement in DO emerges, with the additional suggestion that this improvement is largely restricted to the earlier years. The smooth curve therefore provides a significant enhancement of the display by drawing attention to features of some potential importance which are not immediately obvious from a plot of the raw data. However, these features required further investigation to separate real evidence of change from the effects of sampling variation.

In exploring the effect of an individual variable, it is also necessary to consider the simultaneous effects of others. The lower left-hand panel shows the data plotted against day of the year. Water samples are not taken every day but, when the samples are plotted by day of the year across the entire time period, a very clear relationship is evident. This seasonal effect is a periodic one and so this should be reflected in an appropriate estimate. The smooth curve added to the lower right panel has this periodic property. It also suggests that a simple trigonometric shape may well be adequate to describe the seasonal effect. Once a suitable model for this variable has been constructed, it will be advisable to reexamine the relationship between DO and year, adjusted for the seasonal effect.

The aim of this chapter is to discuss the potential benefits of enhancing graphical displays in this manner, and to illustrate the insights which this can bring to a vari-

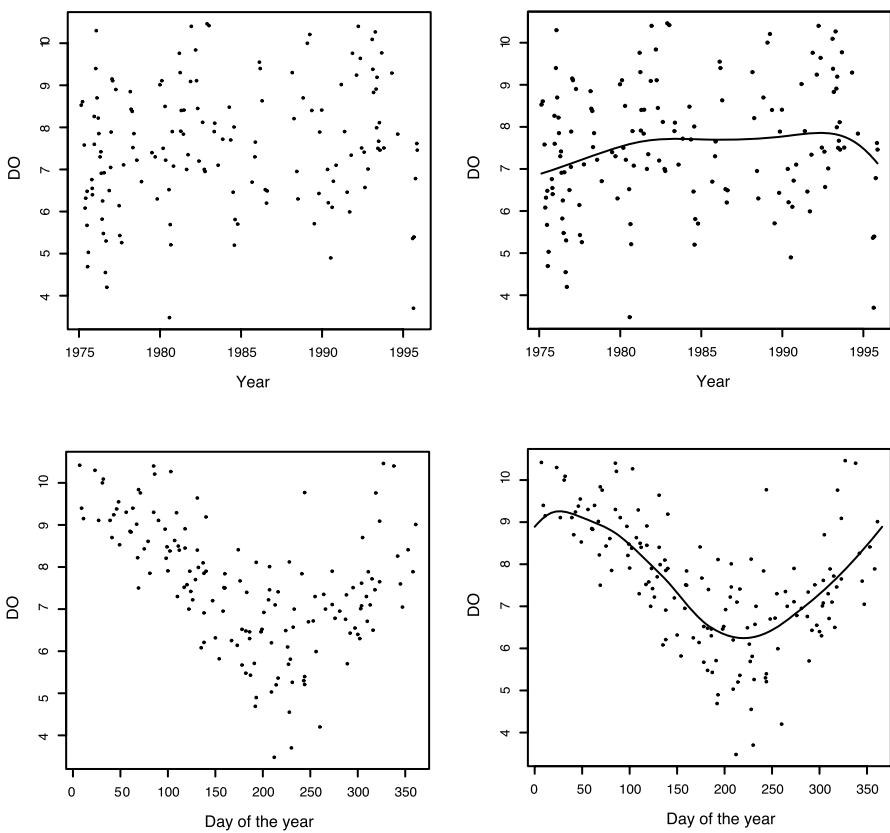


Figure 9.1. The *left-hand panels* shows data on dissolved oxygen (DO) in the Clyde estuary, plotted against year and day within the year. The *right-hand panels* add smooth curves as estimates of the underlying regression functions

ety of types of regression data. From this perspective, graphics are oriented towards the exploration of appropriate *models* for data, as well as towards the display of the observed data themselves. In Sect. 9.2, simple methods of constructing smooth estimates are described and illustrated. The ideas are developed in the context of response data on a continuous measurement scale, but the more general applicability of the concept is indicated by an extension to binary response data. The graphical methods employed are also extended beyond simple displays of the underlying regression estimate to include indications of variability and of the suitability of simple parametric models. These concepts are extended further in Sect. 9.3, where displays of nonparametric regression surfaces relating a response variable to two explanatory variables are discussed. The addition of information on variability and the suitability of parametric models are revisited in this setting. Situations involving several covariates are discussed in Sect. 9.4, where additive models are used to provide descriptions of each separate regression component. Some final discussion is given in Sect. 9.5.

Smoothing in One Dimension

There are many ways in which a nonparametric regression curve can be constructed. These include orthogonal basis functions, a wide variety of approaches based in splines and, more recently, methods based on wavelets. While there are important differences between these approaches from a technical perspective, the particular choice of technique for the construction of a nonparametric regression curve is less important in a graphical setting. The principal issue is how an estimate can be used to best effect, rather than the details of its construction.

For convenience, this chapter will make use of local linear methods of nonparametric regression. These have the advantages of being simple to explain and easy to implement, as well as having theoretical properties which are amenable to relatively straightforward analysis. A further advantage lies in the link with the chapter on smoothing by Loader (2004) in an earlier *Computational Statistics Handbook*, where many of the technical details can be found. The basic ideas of the method are described below, but the emphasis thereafter is on the use of the technique to enhance graphical displays.

With regression data of the form $\{(x_i, y_i) : i = 1, \dots, n\}$, where y denotes a response variable and x a covariate, a general prescription of a model is provided by

$$y_i = m(x_i) + \varepsilon_i,$$

where m denotes a regression function and the ε_i denote independent errors. A parametric form for m can easily be fitted by the method of least squares. A nonparametric estimate of m can be constructed simply by fitting a parametric model locally. For example, an estimate of m at the covariate value x arises from minimising the weighted least squares

$$\sum_{i=1}^n \{y_i - \alpha - \beta(x_i - x)\}^2 w(x_i - x; h) \quad (9.1)$$

over α and β . The estimate $\hat{m}(x)$ is the fitted value of the regression at x , namely $\hat{\alpha}$. By choosing the weight function $w(x_i - x; h)$ to be decreasing in $|x_i - x|$, the linear regression is fitted locally as substantial weight is placed only on those observations near x . Unless otherwise noted, this chapter will adopt a weight function w , which is a normal density centred on 0 with standard deviation h . The parameter h controls the width of the weight function and therefore the extent of its local influence. This in turn dictates the degree of smoothness of the estimate. For this reason, h is usually referred to as the *smoothing parameter* or *bandwidth*.

Computationally, the solution of the weighted least squares (9.1) is straightforward, leading to an estimate of the form $\hat{m}(x) = v^\top y$, where the vector v is a simple function of x , the covariate values x_i and the weights $w(x_i - x; h)$. Specifically, the i th element of v is

$$v_i = \frac{1}{n} \frac{\{s_2(x; h) - s_1(x; h)(x_i - x)\} w(x_i - x; h)}{s_2(x; h)s_0(x; h) - s_1(x; h)^2},$$

where $s_r(x; h) = \{\sum(x_i - x)^r w(x_i - x; h)\}/n$. The estimate can therefore be computed at a set of covariate values through the expression Sy , where S denotes a *smoothing matrix* whose rows contain the vectors v required to construct the estimate at the points of interest.

This representation emphasises the important fact that the estimation process is linear in the response data y . It also suggests useful analogies with standard linear modelling techniques. In particular, the degrees of freedom associated with a linear model can be identified as the trace of the projection matrix P which creates the fitted values as $\hat{y} = Py$. It is therefore convenient to define the *approximate degrees of freedom* associated with a nonparametric estimate as $v = \text{tr}\{S\}$, where S is the smoothing matrix which creates the fitted values at the observed covariate values $\{x_i; i = 1, \dots, n\}$. As the smoothing parameter h is increased, the influence of the weight function extends across a greater range of the covariate axis and the flexibility of the estimate is reduced. This corresponds to a reduction in the approximate degrees of freedom associated with the estimate.

The approximate degrees of freedom therefore provide a helpful alternative scale on which degree of smoothness can be expressed. The estimate for year shown in Fig. 9.1 was produced with a smoothing parameter corresponding to four degrees of freedom, namely $h = 3.16$. This allows a moderate degree of flexibility in the curve beyond the two degrees of freedom associated with a simple linear shape.

The choice of smoothing parameter h , or equivalently of the approximate degrees of freedom v , is therefore of some importance. From a graphical and exploratory perspective it is helpful to plot the estimates over a wide range of smoothing parameters, to view the effects of applying different degrees of local fitting. This is particularly effective in the form of an interactive animation. However, it is also worth considering ways of automatically identifying suitable choices of smoothing parameter.

Some very effective methods of doing this have been developed for particular types of regression problem, but other proposals have the advantage of very wide applicability. One of the most popular of these has been cross-validation, where h is chosen to minimise

$$\sum_{i=1}^n \{y_{-i} - \hat{m}_i(x_i)\}^2.$$

The subscript on \hat{m}_{-i} indicates that the estimate is constructed from the dataset with the i th observation omitted, and so the criterion being minimised represents the prediction error of the estimate. There is some evidence that this approach produces substantial variation in its selected smoothing parameters. This chapter will therefore use an alternative criterion, proposed by Hurvich et al. (1998), based on Akaike's information criterion (AIC). This chooses h to minimise

$$\log(\text{RSS}/n) + 1 + \frac{2(v+1)}{(n-v-2)}, \quad (9.2)$$

where RSS denotes the residual sum-of-squares $\sum_{i=1}^n \{y_i - \hat{m}(x_i)\}^2$ and, as described above, $v = \text{tr}\{S\}$. In general, this method offers a very useful and usually very effective

means of selecting an appropriate degree of smoothness. However, any method of automatic selection must be used carefully.

In the initial plots of the DO data in Fig. 9.1 it was noted that the day effect is a periodic one. This can easily be accommodated in the construction of a smooth estimate by employing a periodic weight function. Since a linear model is not appropriate for periodic data, a locally weighted mean offers a simple solution. A smooth estimate is then available as the value of α which minimises the weighted least squares

$$\sum_{i=1}^n \{y_i - \alpha\}^2 \exp \left\{ \frac{1}{h} \cos(2\pi(x_i - x)/366) \right\}.$$

This uses an unscaled von Mises density as a weight function, with a period of 366 days to allow for leap years. In order to allow the estimate to express shapes beyond a standard trigonometric pattern, the approximate degrees of freedom were set to the slightly higher value of 6 in constructing the estimate of the seasonal effect in Fig. 9.1.

Nonparametric curve estimates are very useful as a means of highlighting the potential shapes of underlying regression relationships. However, like any estimate based on limited information, they are subject to the effects of variability. Indeed, the flexibility which is the very motivation for a nonparametric approach will also increase the sensitivity of the estimate to sampling variation in the data. It is therefore important not only to examine curve estimates but also to examine their associated variability.

The linear representation of the estimate as $\hat{m}(x) = v^\top y$, where v is a known vector as discussed above, means that its variance is readily available as $\text{var}\{\hat{m}(x)\} = (\sum_{i=1}^n v_i^2) \sigma^2$, where σ^2 is the common variance of the errors ε_i . The calculation of standard errors then requires an estimate of σ^2 . Pursuing the analogy with the linear models mentioned above leads to proposals such as $\hat{\sigma}^2 = \text{RSS}/df$, where df is an appropriate value for the degrees of freedom for error. Other approaches are based on local differencing. A particularly effective proposal of Gasser et al. (1986) is based on the deviations of each observation from the linear interpolation between its neighbours, as $\tilde{\varepsilon}_i = y_i - a_i y_{i-1} - (1 - a_i) y_{i+1}$, where $a_i = (x_{i+1} - x_i)/(x_{i+1} - x_{i-1})$, under the assumption that the data have been ordered by increasing x value. This leads to the estimate

$$\hat{\sigma}^2 = \frac{1}{n-2} \sum_{i=2}^{n-1} \frac{\tilde{\varepsilon}_i^2}{1 + a_i^2 + (1 - a_i)^2}.$$

The standard error of $\hat{m}(x)$ is then available as $\sqrt{(\sum_{i=1}^n v_i^2)} \hat{\sigma}$.

The left-hand plot of Fig. 9.2 shows the estimate of the seasonal effect in the Clyde data, with curves indicating a distance of two standard errors from the estimated regression function. Some care has to be exercised in the interpretation of this band. The smoothing inherent in the construction of a nonparametric regression curve inevitably leads to bias, as discussed by Loader (2004). The band cannot therefore be given a strict interpretation in terms of confidence. However, it does give a good indication of the degree of variability in the estimate and so it is usually referred to as a *variability band*.

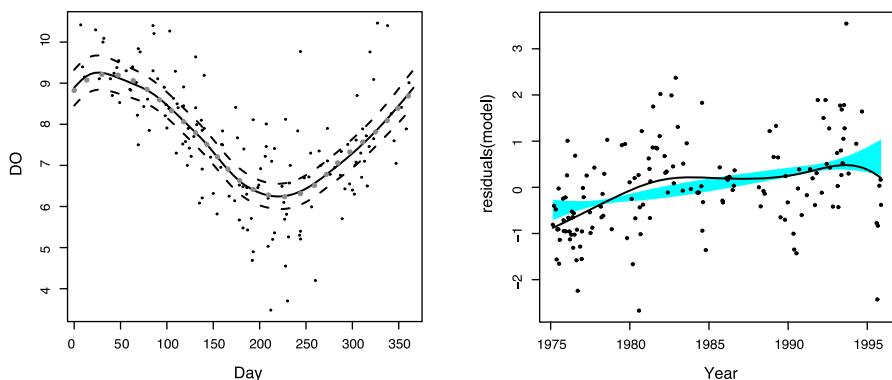


Figure 9.2. The *left-hand panel* shows a smooth curve as an estimate of the underlying regression function for the seasonal effect in the Clyde data, with variability bands to indicate the precision of estimation. The *dotted line* denotes a smoothed version of a shifted and scaled cosine model. The *right-hand panel* shows an estimate of the year effect after adjustment for the seasonal effect. A reference band has been added to indicate where a smooth curve is likely to lie if the underlying relationship is linear

A natural model for the seasonal effect is a shifted and scaled cosine curve, of the form

$$y_i = \alpha + \beta \cos \left\{ 2\pi \frac{(x_i - \theta)}{366} \right\} + \varepsilon_i,$$

where the smaller effect across years, if present at all, is ignored at the moment. McMullan et al. (2003) describe this approach. A simple expansion of the cosine term allows this model to be written in simple linear form, which can then be fitted to be observed data very easily.

However, some thought is required in comparing a parametric form with a nonparametric estimate. As noted above, bias is an inevitable consequence of nonparametric smoothing. We should therefore compare our nonparametric estimate with what we expect to see when a nonparametric estimate is constructed from data generated by the cosine model. This can easily be done by considering $E\{\hat{m}(x)\}$, where the expectation is calculated under the cosine model. The simple fact that $E\{Sy\} = SE\{y\}$ suggests that we should compare the nonparametric estimate Sy with a smoothed version of the vector of fitted values \hat{y} from the cosine model, namely $S\hat{y}$. This curve has been added to the left hand plot of Fig. 9.2 and it agrees very closely with the nonparametric estimate. The cosine model can therefore be adopted as a good description of the seasonal effect.

Since the seasonal effect in the Clyde data is so strong, it is advisable to reexamine the year effect after adjustment for this. Nonparametric models involving more than one covariate will be discussed later in the chapter. For the moment, a simple expedient is to plot the residuals from the cosine model against year, as shown in the right hand panel of Fig. 9.2. The reduction in variation over the marginal plot of DO against year is immediately apparent.

It is now also natural to consider whether a simple linear model might be adequate to describe the underlying relationship, with the curvature exhibited by the nonparametric estimate attributable to sampling variation. Variability bands provide one way of approaching this. However, a more direct way of assessing the evidence is through a *reference band*, which indicates where a nonparametric estimate is likely to lie if the underlying regression is indeed linear. Since bias in the nonparametric estimate depends on the curvature of the underlying regression function, it follows that a nonparametric estimate, fitted by the local linear method, is unbiased in the special case of data from a linear model. If the fitted linear model at the covariate value x is represented as $\sum_{i=1}^n l_i y_i$, then it is straightforward to see that the variance of the difference between the linear and nonparametric models is simply $\sum_{i=1}^n (v_i - l_i)^2 \sigma^2$. On substituting an estimate of σ^2 , a reference band extending for a distance of two standard errors above and below the fitted linear model can then easily be constructed.

This is illustrated in the right-hand panel of Fig. 9.2, where the evidence against a simple linear model is confirmed in this graphical manner. This addition to the plot has therefore identified an important feature which is not easily spotted from plots of the raw data.

A formal, global test can also be carried out, as described by Bowman and Azzalini (1997), but the discussion here will be restricted to graphical aspects. It should also be noted that the calculations for the reference band have been adjusted to account for the correlations in the residuals from the fitted cosine model. However, this effect is a very small one.

The idea of local fitting of a relevant parametric model is a very powerful one which can be extended to a wide variety of settings and types of data. For example, nonparametric versions of generalised linear models can be constructed simply by adding suitable weights to the relevant log-likelihood function. Under an assumption of independent observations, the log-likelihood for a generalised linear model can be represented as $\sum_{i=1}^n l(\alpha, \beta)$. A local likelihood for nonparametric estimation at the covariate value x can then be constructed as

$$\sum_{i=1}^n l(\alpha, \beta) w(x_i - x; h)$$

and the fitted value of the model at x extracted to provide the nonparametric estimate $\hat{m}(x)$.

An example is provided by the data displayed in the top left-hand panel of Fig. 9.3, which indicate whether the dissolved oxygen in water samples, taken from two well-separated monitoring points on the river, was below (1) or above (0) a threshold of 5mg/l, which is the level required for healthy resident and migratory fish populations. In order to standardise for the year and seasonal effects which were noted in the earlier analysis, the measurements considered here are restricted to the years from 1982 onwards and to the summer months of May and June. The level of dissolved oxygen is likely to be related to temperature, which has therefore been used as a covariate. It is particularly difficult to assess the nature of any underlying relationship when the response variable is binary, even when some random variation has been added to the response values to allow the density of points to be identified more clearly.

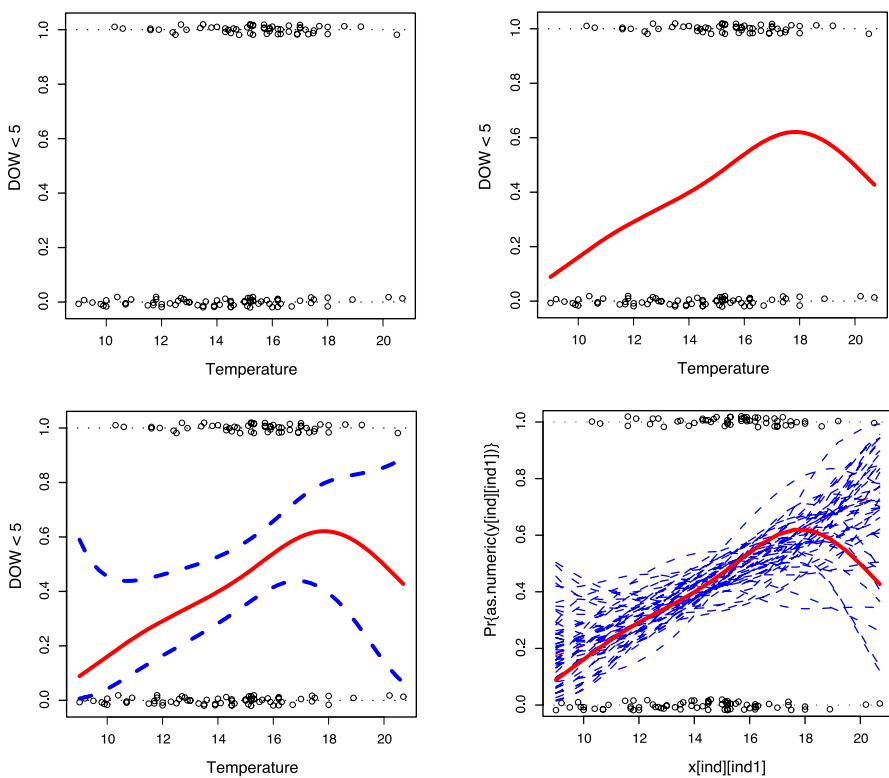


Figure 9.3. The top left-hand panel shows data on the occurrence of very low (< 5 %) levels of dissolved oxygen in the River Clyde, related to the temperature of the water. The top right-hand panel shows a smooth nonparametric regression curve to display the pattern of change in the probability of very low dissolved oxygen as a function of temperature. The bottom left-hand panel displays standard error bands to indicate the variability in the nonparametric estimate. The bottom right-hand panel shows nonparametric regression curves simulated from the fitted logistic model, together with the nonparametric curve from the observed data

A natural model in this setting is a simple logistic regression where the probability $p(x)$ of observing a measurement below the threshold is assumed to be described by $\exp(\alpha + \beta x)/(1 + \exp(\alpha + \beta x))$, where α and β are unknown parameters. The log-likelihood contribution for an observation (x_i, y_i) can be written as $y_i \log\{p(x_i)\} + (1 - y_i) \log\{1 - p(x_i)\}$. Maximising the weighted log-likelihood, as described above, and using a smoothing parameter $h = 2$, produces the nonparametric estimate shown in the top right-hand panel of Fig. 9.3. The technical details of this process are described by Fan et al. (1995).

The generally increasing nature of the relationship between low measurements and temperature is apparent. However, the downturn in the estimate of the probability for the highest temperature is a surprise. Again, it is helpful to add information on variability to the plot. The bottom left hand panel of Fig. 9.3 shows variability bands

for the estimate, constructed by carrying the weights through the usual process of deriving standard errors in generalised linear models. These indicate a high degree of variability at high temperatures.

The suitability of a linear logistic model can be assessed more directly by constructing a reference band. A simple way of doing this here is to simulate data from the fitted linear logistic model and construct a nonparametric estimate from each set of simulated data. The results from repeating this 50 times are displayed in the bottom right-hand panel of Fig. 9.3. The appearance of some other estimates with curvature similar to that exhibited in the original estimate offers reassurance that the data are indeed consistent with the linear logistic model and prevents an inappropriate interpretation of a feature in the nonparametric estimate which can reasonably be attributed to sampling variation.

9.3

Smoothing in Two Dimensions

The implementation of smoothing techniques with two covariates is a particularly important application because of the wide variety of types of data where it is helpful to explore the combined effects of two variables. Spatial data provide an immediate example, where the local characteristics of particular regions lead to measurement patterns which are often not well described by simple parametric shapes. As in the univariate case, a variety of different approaches to the construction of a smooth estimate of an underlying regression function is available. In particular, the extension of the local linear approach is very straightforward. From a set of data $\{(x_{1i}, x_{2i}, y_i) : i = 1, \dots, n\}$, where y denotes a response variable and x_1, x_2 are covariates, an estimate of m at the covariate value x arises from minimising the weighted least squares

$$\sum_{i=1}^n \{y_i - \alpha - \beta_1(x_{1i} - x_1) - \beta_2(x_{2i} - x_2)\}^2 w(x_{1i} - x_1; h_1) w(x_{2i} - x_2; h_2)$$

over α, β_1 and β_2 . The estimate $\hat{m}(x)$ is the fitted value of the regression at x , namely $\hat{\alpha}$. More complex forms of weighting are possible and Härdle et al. (2004) give a more general formulation. However, the product form shown above is particularly attractive in the simplicity of its construction.

From the form of this weighted sum-of-squares, it is immediately obvious that the estimator $\hat{m}(x)$ again has a linear form $v^\top y$, for a vector of known constants v . The concept of approximate degrees of freedom then transfers immediately, along with automatic methods of smoothing parameter selection such as the AIC method described in (9.2). Estimation of the underlying error variance needs more specific thought, although the same principles of local differencing apply, as described by Munk et al. (2005). For the particular case of two covariates, a method based on a very small degree of local smoothing is also available, as described by Bock et al. (2005), and this is used in the illustrations of this chapter.

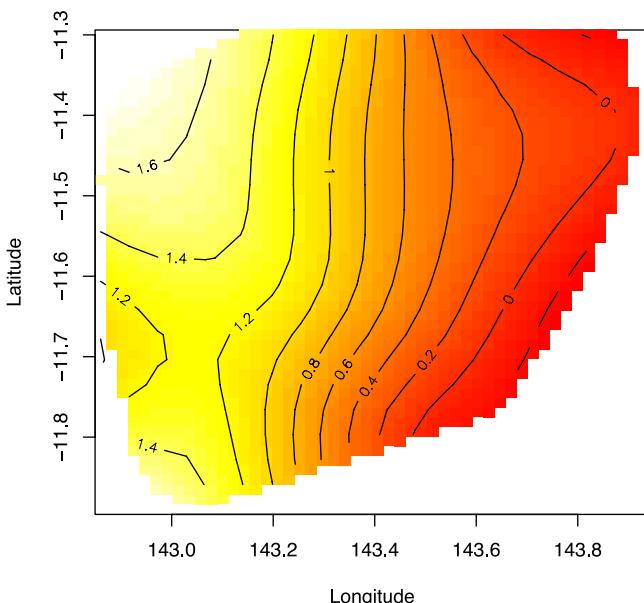


Figure 9.4. A nonparametric estimate of a regression surface relating the mean level of a catch score to latitude and longitude, using data from the Great Barrier Reef

Figure 9.4 displays a smooth estimate of a regression surface derived from data on a catch score, representing the abundance of marine life on the sea bed at various sampling points in a region near the Great Barrier Reef, as a function of latitude and longitude. Poiner et al. (1997) describe the background to these data and Bowman and Azzalini (1997) illustrate the application of smoothing techniques on various subsets. Here, the data for two successive years are examined to investigate the relationship between the catch score and the covariates latitude and longitude.

Several types of graphical display are available for the resulting estimate. Figure 9.4 uses both colour shading and contour levels to indicate the height of the estimated regression surface. The simultaneous use of both is helpful in enhancing the interpretation in a form familiar to users of geographical maps. However, three-dimensional projections are also easy to construct with many software packages and the ability to render surfaces to a high degree of visual quality is now commonly available. The display is further enhanced by animated rotation, providing a very realistic perception of a real three-dimensional object. Figure 9.5 displays this kind of representation in static form.

Figures 9.4 and 9.5 were produced with the smoothing parameters $(h_1, h_2) = (0.18, 0.09)$ selected by AIC and equivalent to 12 degrees of freedom. This choice was based on a single underlying parameter h , which was then scaled by the sample standard deviations, s_1, s_2 , of the covariates to provide a pair of smoothing parameters, (hs_1, hs_2) . A common smoothing parameter for each dimension or unrestricted choices of h_1 and h_2 could also be allowed.

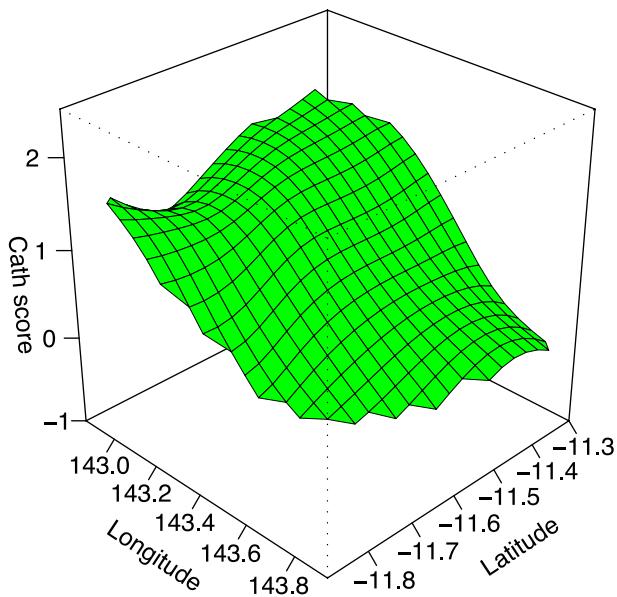


Figure 9.5. A smooth surface representing an estimate of the regression function of catch score on latitude and longitude simultaneously

A clear drop in mean catch score as longitude increases is indicated. However, more detailed insight is available from the estimated surface, with clear indication of a nonlinear pattern with increasing longitude. In fact, due to the orientation of the coast in this region, longitude broadly corresponds to distance offshore, leading to a natural biological interpretation, with relatively constant levels of marine life abundance near the coast followed by rapid decline in deeper water. The smooth surface therefore provides a significant enhancement of the display by drawing attention to features of some potential importance which are not immediately obvious from plots of the raw data.

It is natural to revisit in the setting of two covariates the discussion of the benefits of adding further information, particularly on variability and reference models, to displays of nonparametric estimates. The graphical issues are now rather different, given the very different ways in which the estimate itself must be represented. One possibility is to mix the different types of representation (colour, contours, three-dimensional surfaces) with an estimate represented in one way and information on variability in another. For example, the colour and contour displays in Fig. 9.4 might be used for these two different purposes, although this particular combination can be rather difficult to interpret.

One attractive option is to combine surface and colour information and Fig. 9.6 illustrates this by painting the estimated regression surface to indicate the variations in standard error in different locations. Discrete levels of colour have been used for simplicity, with a separate colour associated with each individual polygonal surface

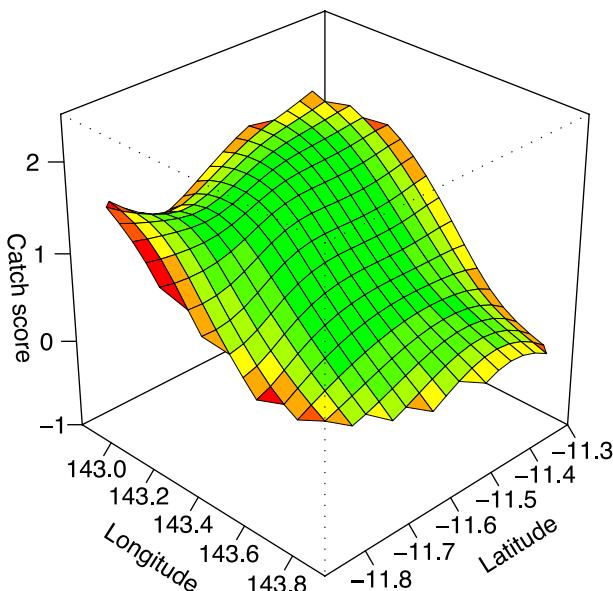


Figure 9.6. An estimate of the regression function of catch score on latitude and longitude, with colour coding to add information on the relative size of the standard error of estimation across the surface

panel. This highlights the areas where precision in the estimate is low. It is no surprise to find these at the edges of the surface, where information is less plentiful. However, it is particularly helpful to be able to identify the relatively high standard errors at low latitude.

This idea extends to the assessment of reference models, such as a linear trend across latitude and longitude. This is shown in the right-hand panel of Fig. 9.7, where the surface panels are painted according to the size of the standardised difference $(\hat{m} - \hat{p})/\text{s.e.}(\hat{m} - \hat{p})$, to assess graphically the plausibility of a linear model \hat{p} in two covariates. The red panels indicate a difference of more than 2, and the blue panels of less than -2, standard errors (s.e.). This gives an immediate graphical indication that the curvature in the surface is not consistent with a linear shape.

Surfaces are three-dimensional objects and software to display such objects in high quality is now widely available. The OpenGL system is a good example of this, and access to these powerful facilities is now possible from statistical computing environments such as R, through the `rgl` package described by Adler (2005). The left hand plot of Fig. 9.7 gives one example of this, showing a regression surface for the Reef data with additional wire mesh surfaces to define a reference region for a linear model. The protrusion of the estimated surface through this reference region indicates the substantial lack-of-fit of the linear model. This higher quality of three-dimensional representation, together with the ability to rotate the angle of view interactively, provides a very attractive and useful display.

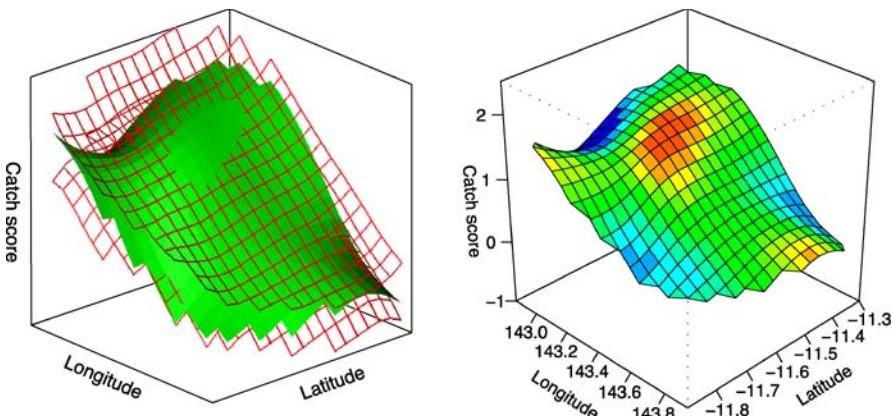


Figure 9.7. Estimates of the regression function of catch score on latitude and longitude. The *left-hand panel* displays a reference band for linearity while the *right-hand panel* uses colour coding to indicate the size of the difference between the estimate and a linear regression function, in units of standard error

The two plots of Fig. 9.8 give an example of a further extension of this type of display to the comparison of two different surfaces, referring in this case to two different years of sampling. The surface panels are now painted by the values of the standardised distance $(\hat{m}_1 - \hat{m}_2)/\text{s.e.}(\hat{m}_1 - \hat{m}_2)$, and so the added colour assesses the evidence for differences between the two underlying surfaces m_1 and m_2 . The very small regions where the estimates are more than two standard errors apart indicate that there are only relatively small differences between the catch score patterns in the two years.

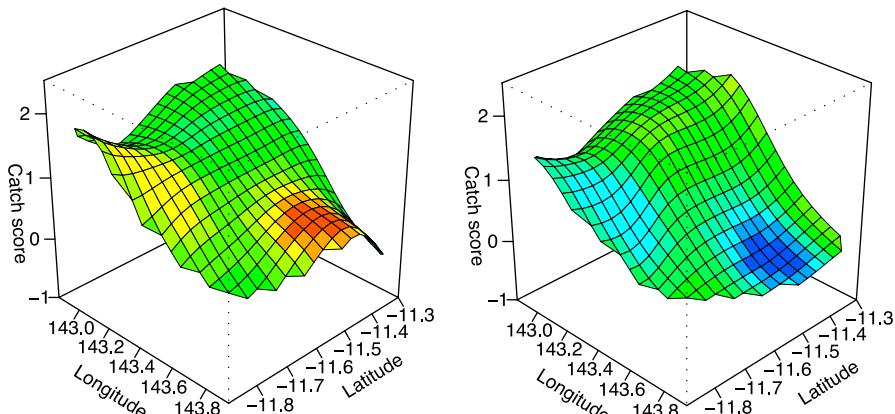


Figure 9.8. [This figure also appears in the color insert.] Estimates of regression functions of catch score on latitude and longitude for two different years of data collection. Colour coding has been used to indicate the standard differences between the two surfaces

A more detailed description of the construction of displays of this type, together with techniques for more global assessment of the evidence for differences between surfaces and methods for incorporating correlated data, are provided in Bowman (2005).

Additive Models

In order to be useful statistical tools, regression models need to be able to incorporate arbitrary numbers of covariates. In principle, the local fitting approach described above could be extended to any number of covariates. However, in practice, the performance of such simultaneous estimation deteriorates rapidly as the dimensionality of the problem increases. A more parsimonious and powerful approach is offered by *additive models*, developed by Friedman and Stuetzle (1981), Hastie and Tibshirani (1990) and many other authors. These allow each covariate to contribute to the model in a nonparametric manner but assume that the effects of these are additive, so that a model for data $\{(x_{1i}, \dots, x_{pi}, y_i); i = 1, \dots, n\}$ is given by

$$y_i = \alpha + m_1(x_{1i}) + \dots + m_p(x_{pi}) + \varepsilon_i.$$

This extends the usual linear regression model by allowing the effects of the covariates to be nonparametric in shape. To ensure that the model is identifiable, the constraint that each component function m_j averages to zero across the observed covariate values can be adopted.

Additive models can be fitted to observed data through the *backfitting algorithm*, where the vectors of estimates $\hat{m}_j = (\hat{m}_j(x_{j1}), \dots, \hat{m}_j(x_{jn}))^\top$ are updated from iteration r to $r + 1$ as

$$\hat{m}_j^{(r+1)} = S_j \left(y - \hat{\alpha} \mathbf{1} - \sum_{k < j} \hat{m}_k^{(r+1)} - \sum_{k > j} \hat{m}_k^{(r)} \right). \quad (9.3)$$

This applies a smoothing operation, expressed in the smoothing matrix S_j for the j th covariate, to the partial residuals constructed by subtracting the current estimates of all the other model components from the data vector y . The estimate of the intercept term α can be held fixed at the sample mean \bar{y} throughout. The identifiability constraint on each component function can be incorporated by adjusting the vectors \hat{m}_j to have mean zero after each iteration.

The backfitting algorithm described above is not the only way in which an additive model can be fitted to observed data. In particular, Mammen et al. (1999) proposed a smooth backfitting algorithm which has a number of attractive properties. Nielsen and Sperlich (2005) give a clear exposition of this approach, with practical proposals for bandwidth selection.

Hastie and Tibshirani (1990) discuss how the standard errors of the estimates can also be constructed. Computationally, the end result of the iterative scheme (9.3) can be expressed in matrix form as $\hat{y} = Py = (P_0 + \sum_{j=1}^p P_j)y$, where P_0 is filled with

the values $1/n$ to estimate α , and the remaining matrices construct the component estimates as $\hat{m}_j = P_j y$. The standard errors of \hat{m}_j at each observed covariate value are then available as the square roots of the diagonal entries of $P_j P_j^\top \hat{\sigma}^2$, where the estimate of error variance $\hat{\sigma}^2$ can be constructed as RSS/df and df denotes the degrees of freedom for error. Hastie and Tibshirani give the details on how that can be constructed, again by analogy with its characterisation in linear models.

The left-hand panels of Fig. 9.9 show the results of fitting an additive model in latitude and longitude to the Reef data. The top two panels show the estimated functions for each covariate, together with the partial residuals, while the bottom panel shows the resulting surface. The additive nature of the surface is apparent as slices across latitude always show the same shape of longitude effect and vice versa. (Notice that colour has been used here simply to emphasise the heights of the surface at different locations.)

The level of smoothing was determined by setting the number of approximate degrees of freedom to four for each covariate. An alternative approach, advocated by Wood (2000), applies cross-validation as an automatic method of smoothing parameter selection at each iteration of the estimation process defined by (9.3). The effects of this strategy on the Reef data are displayed in the right-hand panels of Fig. 9.9. The estimate of the longitude effect is very similar but a very large smoothing parameter has been selected for latitude, leading to a linear estimate. Based on the earlier estimate for the latitude effect, using four degrees of freedom, a linear model for this term is a reasonable strategy to adopt. This leads to a *semiparametric* model, where one component is linear and the other is nonparametric. This hybrid approach takes advantage of the strength of parametric estimation where model components of this type are justified.

For a further example of additive models, the Clyde data are revisited. When water samples are collected, a variety of measurements are made on these. This includes temperature and salinity and it is interesting to explore the extent to which the DO level in the samples can be explained by these physical parameters. Clearly, temperature has a strong relationship with the day of the year. In fact, salinity also has a strong relationship with this variable, as it measures the extent to which fresh water from the river and salt water from the sea mix together, and this has a strong seasonal component related to the volume of river flow. It is therefore inappropriate to use all three of these variables in a model for DO. As Hastie and Tibshirani (1990) observe, the effect of *concurvity*, where explanatory variables have strong curved relationships, creates difficulties analogous to those associated with collinearity in a linear model. The three explanatory variables to be considered are therefore year, temperature and salinity, with the latter variable on a $\log(\text{salinity} + 1)$ scale to reduce substantial skewness.

The top two panels of Fig. 9.10 show nonparametric curve estimates based on regressions of DO on temperature and salinity separately. The lower three panels of the figure show the effects of fitting an additive model which expresses the DO values as a sum of year, temperature and salinity components simultaneously. One striking feature expressed in the partial residuals is the substantial reduction in the variability of the data compared to that displayed in the marginal scatterplots, as each component

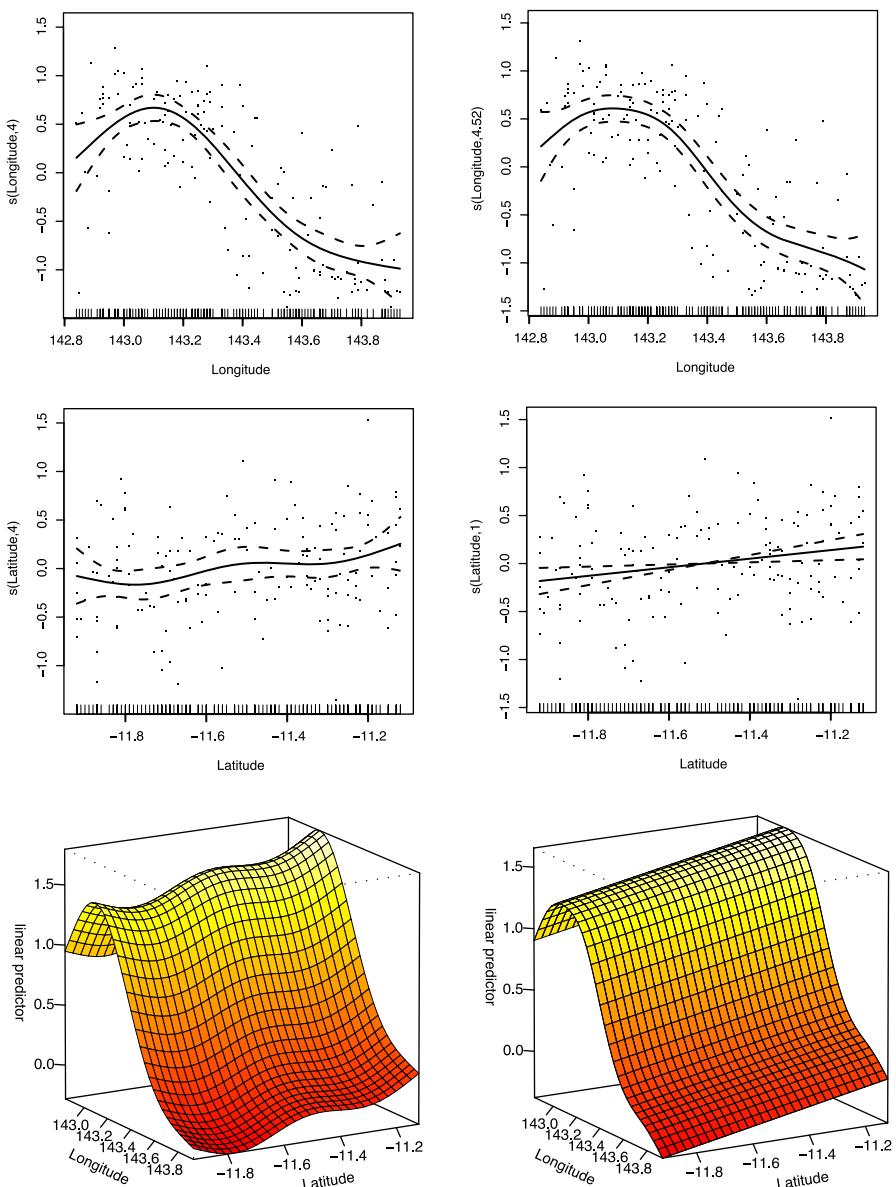


Figure 9.9. The left-hand panels show the components and fitted surface of an additive model for the Reef data, using four degrees of freedom for each covariate. The right-hand panels show the results for an additive model when cross-validation is used to select the degree of smoothing at each step of the backfitting algorithm

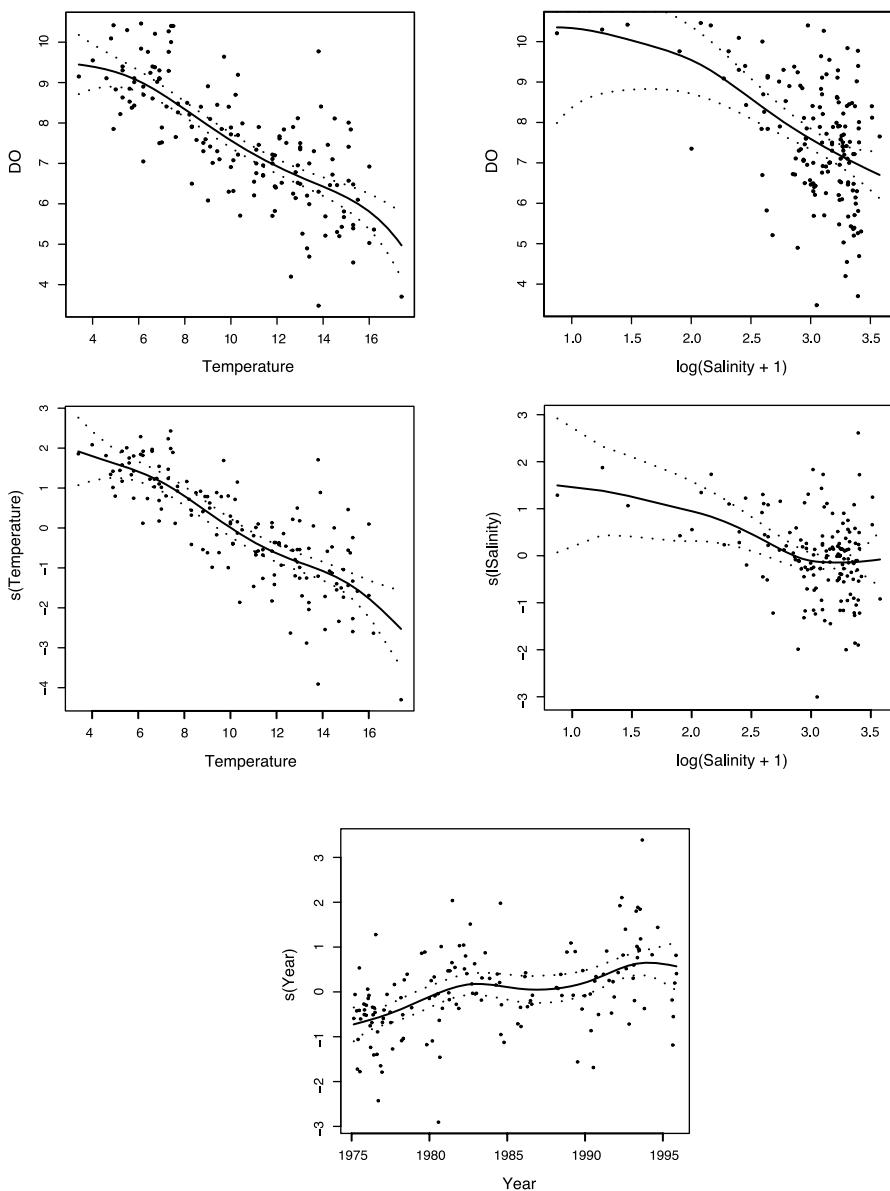


Figure 9.10. The *top two panels* show plots of dissolved oxygen against temperature and log salinity with the Clyde data. The *middle two panels* show the fitted functions for temperature and log salinity, and the *lower panel* for year, from an additive model

focuses only on the variation that is not explained by the others. Some interesting features are displayed in the curve estimates, with DO declining in a linear manner as temperature increases while DO is elevated at low salinity but constant elsewhere. Reassuringly, the trend across years remains very similar to the patterns displayed in earlier analysis, when adjustment involved only the day of the year. This collection of graphs therefore provides a very powerful summary of the data across all the covariates involved and brings considerable insight into the factors which influence the observed values of DO.

Discussion

9.5

The material of this chapter has aimed to introduce the concepts and aims of nonparametric regression as a means of adding significant value to graphical displays of data. Technical details have been limited only to those required to give a general explanation of the methods. However, a great deal of technical work has been carried out on this topic, which is well represented in the statistical research literature. There are several books in this area and these provide good starting points for further information. Hastie and Tibshirani (1990) give a good general overview of smoothing techniques as well as a detailed treatment of additive models. Green and Silverman (1994) give a very readable and integrated view of the penalty function approach to smoothing models. Fan and Gijbels (1996) gives considerable theoretical insight into the local linear approach to smoothing, while Simonoff (1996) is particularly strong in providing extensive references to the literature on nonparametric regression and is therefore a very good starting point for further reading.

Bowman and Azzalini (1997) give a treatment which aligns most closely with the style of exposition in this chapter and focuses particular attention on smoothing over one and two covariates and on graphical methods. Schimek (2000) provides an collection of contributions from a wide variety of authors on different aspects of the topic. Härdle et al. (2004) give a further general overview of nonparametric modelling, while Ruppert et al. (2003) give an authoritative treatment of semiparametric regression in particular. Wood (2006) provides an excellent introduction to, and overview of, additive models, focussing in particular on the penalized regression splines framework and with a great deal of helpful practical discussion. An alternative wavelet view of modelling is provided by Percival and Walden (2000) in the context of time series analysis. On specifically graphical issues, Cleveland (1993) makes excellent use of smoothing techniques in the general context of visualising data. Material provided by Loader (2004) on local regression techniques and Horowitz (2004) on semiparametric models in an earlier *Handbook of Computational Statistics* are also highly relevant to the material of this chapter.

The role of smoothing techniques in visualisation has been indicated by specific regression examples in this chapter. However, the principles behind this approach allow it to be applied to a very wide range of data structures and application areas. For example, Cole and Green (1992) discuss the estimation of quantile curves, while Kim

and Truong (1998) describe how nonparametric regression can accommodate censored data. Diblasi and Bowman (2001) use smoothing to explore the shape of an empirical variogram constructed from spatial data, examining in particular the evidence for the presence of spatial correlation. Simonoff (1996) discusses the smoothing of ordered categorical data, while Härdle et al. (2004) describe single index models which aim to condense the information in several potential explanatory variables into an index which can then be related to the response variable, possibly in a nonparametric manner. Cook and Weisberg (1994) address the general issue of identifying and exploring the structure of regression data, with particular emphasis on the very helpful roles of smoothing and graphics in doing so. These references indicate the very wide variety of ways in which smoothing techniques can be used to great effect to highlight the patterns in a wide variety of data types, with appropriate visualization forming a central part of the process.

Software to implement smoothing techniques is widely available and many standard statistical packages offer facilities for nonparametric regression in some form. The examples and illustrations in this chapter have all been implemented in the R statistical computing environment (R Development Core Team, 2004) which offers a very extensive set of tools for nonparametric modelling of all types. The one- and two-covariate models of this chapter were fitted with the *sm* (Bowman and Azzalini, 2005) package associated with the monograph of Bowman and Azzalini (1997). The *mgcv* (Wood, 2005) and *gam* (Hastie, 2005) packages provide tools for generalised additive models which can deal with a much wider distributional family beyond the simple illustrations of this chapter.

The website associated with this handbook provides R software which will allow the reader to reproduce the examples of the chapter and, by doing so, offers encouragement for the reader to investigate the potential benefits of nonparametric regression modelling as a tool in the exploration of other regression datasets.

Acknowledgement. The assistance of Dr. Brian Miller of the Scottish Environment Protection Agency in gaining access to, and advising on, the data from the River Clyde is gratefully acknowledged.

References

- Adler, D. (2005) The R package *rgl*: 3D visualization device system (OpenGL) Version 0.65, available from cran.r-project.org.
- Bock, M., Bowman, A.W. and Ismail, B. (2005) Estimation and inference for error variance in bivariate nonparametric regression. Technical report, Department of Statistics, The University of Glasgow.
- Bowman, A.W. and Azzalini, A. (1997) *Applied Smoothing Techniques for Data Analysis*. Oxford University Press, Oxford.
- Bowman, A.W. and Azzalini, A. (2005) The R package *sm*: Smoothing methods for nonparametric regression and density estimation. Version 2.1-0, available from cran.r-project.org.

- Bowman, A. (2005) Comparing nonparametric surfaces. Technical report, Department of Statistics, The University of Glasgow. (Available from www.stats.gla.ac.uk/~adrian).
- Cleveland, W.S. (1993) *Visualising Data*. Hobart Press, Summit, NJ.
- Cole, T.J. and Green, P.J. (1992) Smoothing reference centile curves: the LMS method and penalised likelihood. *Statistics in Medicine* 11:1305–1319.
- Cook, R.D. and Weisberg, S. (1994). *An Introduction to Regression Graphics*. Wiley, New York.
- Diblasi, A. and Bowman, A.W. (2001) On the use of the variogram for checking independence in a Gaussian spatial process. *Biometrics*, 57:211–218.
- Fan, J. and Gijbels, I. (1996). *Local polynomial modelling and its applications*. Chapman & Hall, London.
- Fan, J., Heckmann, N.E. and Wand, M.P. (1995). Local polynomial kernel regression for generalized linear models and quasi likelihood functions. *J. Amer. Statist. Assoc.*, 90:141–50.
- Friedman, J.H. and Stuetzle, W. (1981) Projection pursuit regression. *J. Amer. Statist. Assoc.*, 76:817–23.
- Gasser, T., Sroka, L. and Jennen-Steinmetz, C. (1986) Residual variance and residual pattern in nonlinear regression. *Biometrika*, 73:625–33.
- Green, P.J. and Silverman, B.W. (1994) *Nonparametric Regression and Generalized Linear Models: A Roughness Penalty Approach*. Chapman & Hall, London.
- Härdle, W., Müller, M., Sperlich, S., Werwatz, A. (2004) *Nonparametric and Semiparametric Models*. Springer-Verlag, Berlin.
- Hastie, T. (2005) The R package gam: Generalized Additive Models, Version 0.94, available from cran.r-project.org.
- Hastie, T. and Tibshirani, R. (1990) *Generalized Additive Models*. Chapman & Hall, London.
- Horowitz, J.L. (2004) Semiparametric models. In: Gentle, J.E., Härdle, W. and Mori, Y. (eds) *Handbook of Computational Statistics: concepts and methods*. Springer, Berlin.
- Hurvich, C.M., Simonoff, J.S. and Tsai, C.-L. (1998). Smoothing parameter selection in nonparametric regression using an improved Akaike information criterion, *J.Roy.Stat.Soc., Series B*, 60:271–293.
- Kim, H.T. and Truong, Y.K. (1998) Nonparametric regression estimates with censored data: local linear smoothers and their applications, *Biometrics* 54:1434–1444.
- Loader, C. (2004) Smoothing: local regression techniques. In: Gentle, J.E., Härdle, W. and Mori, Y. (eds) *Handbook of Computational Statistics: concepts and methods*. Springer, Berlin.
- McMullan, A., Bowman, A.W. and Scott, E.M. (2003). Non-linear and nonparametric modelling of seasonal environmental data. *Computational Statistics*, 18:167–183.
- McMullan, A., Bowman, A.W. and Scott, E.M. (2005). Additive models with correlated data: an application to the analysis of water quality data. Technical report, Department of Statistics, The University of Glasgow. (Available from www.stats.gla.ac.uk/~adrian).

-
- Mammen, E., Linton, O.B. and Nielsen, T.J. (1999) The existence and asymptotic properties of a backfitting projection algorithm under weak conditions. *Annals of Statistics* 27:1443–1490.
- Munk, A., Bissantz, N., Wagner, T. and Freitag, G. (2005) On difference-based variance estimation in nonparametric regression when the covariate is high-dimensional. *J. Roy. Statistic. Soc., Series B*, 67:19–41.
- Nielsen, T.J. and Sperlich, S. (2005) Smooth backfitting in practice. *J. Roy. Statistic. Soc., Series B*, 67:43–61.
- Percival, D.B. and Walden, A.T. (2000) *Wavelet Methods for Time Series Analysis*. Cambridge University Press, Cambridge.
- Poiner, I.R., Blaber, S.J.M., Brewer, D.T., Burridge, C.Y., Caesar, D., Connell, M., Dennis, D., Dews, G.D., Ellis, A.N., Farmer, M., Fry, G.J., Glaister, J., Gribble, N., Hill, B.J., Long, B.G., Milton, D.A., Pitcher, C.R., Proh, D., Salini, J.P., Thomas, M.R., Toscas, P., Veronise, S., Wang, Y.G., Wassenberg, T.J. (1997) The effects of prawn trawling in the far northern section of the Great Barrier Reef. Final report to GBRMPA and FRDC on 1991–96 research. CSIRO Division of Marine Research, Queensland Dept. of Primary Industries.
- R Development Core Team (2004) R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL <http://www.R-project.org>.
- Ruppert, D., Wand, M.P. and Carroll, R.J. *Semiparametric Regression*. Cambridge University Press, Cambridge.
- Schimek, M. (2000) *Smoothing and Regression: Approaches, Computation and Application*. Wiley, New York.
- Simonoff, J.S. (1996) *Smoothing Methods in Statistics*. Springer-Verlag, New York.
- Wood, S.N. (2000) Modelling and smoothing parameter estimation with multiple quadratic penalties. *J.Roy.Stat.Soc., Series B*, 62:413–428.
- Wood, S.N. (2005) The R package mgcv: GAMs with GCV smoothness estimation and GAMMs by REML/PQL. Version, 1.3-9 available from cran.r-project.org.
- Wood, S.N. (2006) *Generalized Additive Models: An Introduction with R*. CRC Press, London.

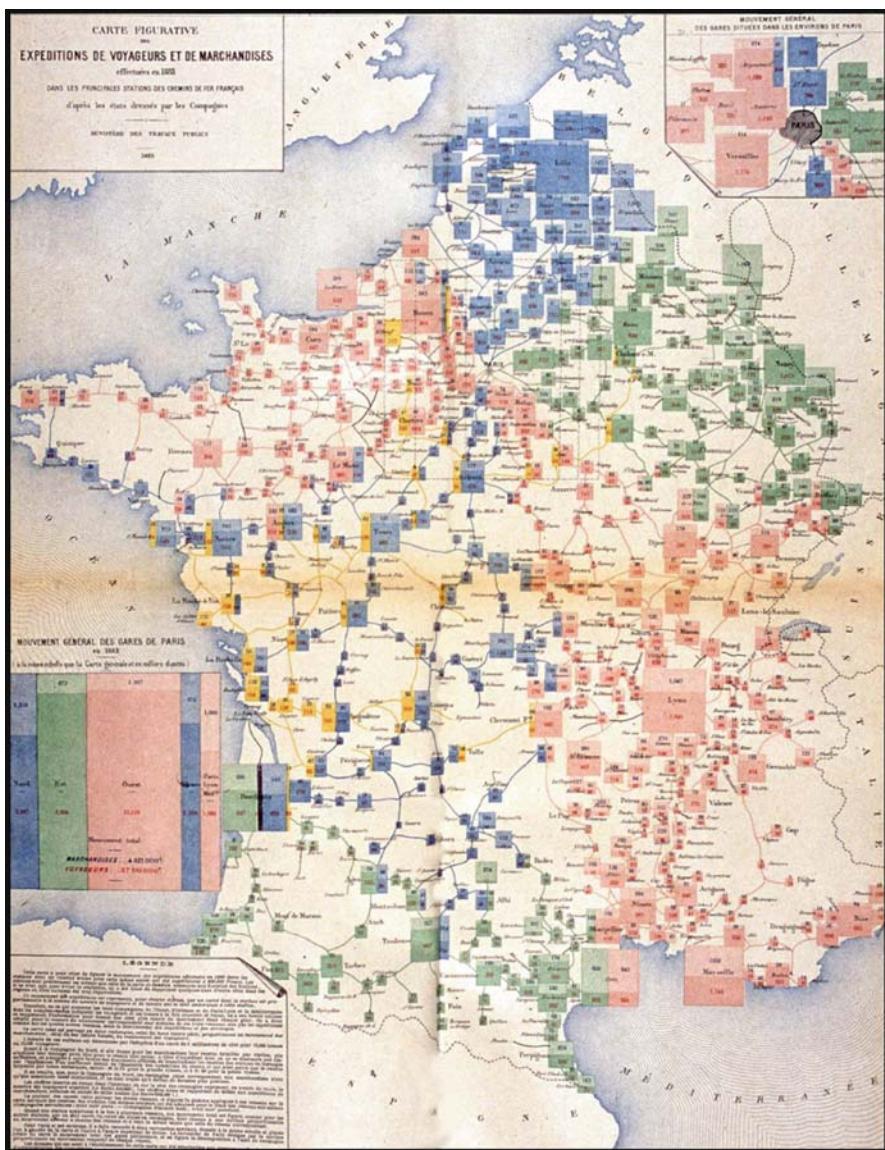


Figure II.1.12. Mouvement des voyageurs et des marchandises dans les principales stations de chemins de fer en 1882. Scale: 2 mm² = 10 000 passagers or tons of freight. Source: Album, 1884, Plate 11 (author's collection)

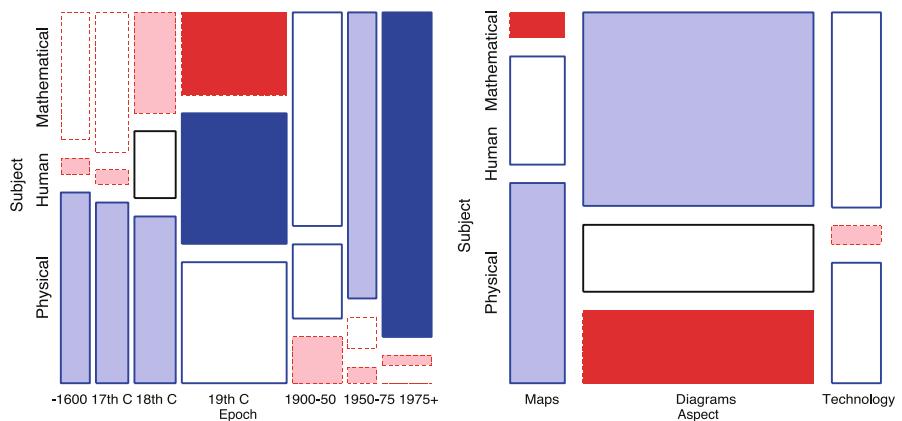


Figure II.1.17. Mosaic plots for milestones items, classified by Subject, Aspect and Epoch. Cells with greater (less) frequency than expected under independence are coloured blue (red), with intensity proportional to the deviation from independence

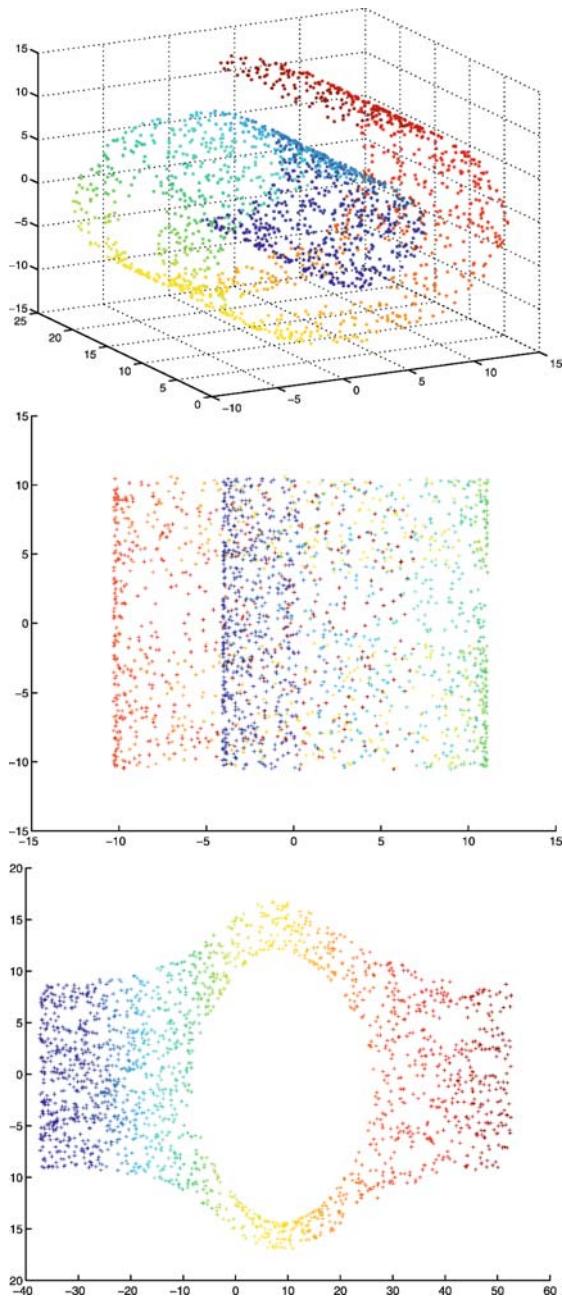


Figure II.4.6. Top panel: original data (2000 data points) arranged along a nonlinear surface (Swiss Roll). Middle panel: 2-D MDS representation based on a complete weighted graph. Bottom panel: 2-D MDS representation based on 20 nearest-neighbor graph

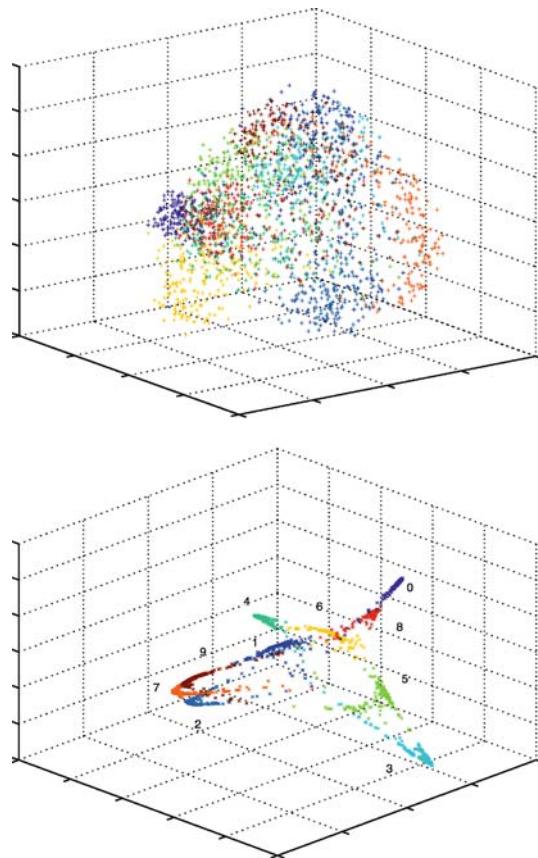


Figure II.4.8. PCA layout of digits dataset (*top panel*) and the 3-D graph layout (*bottom panel*)

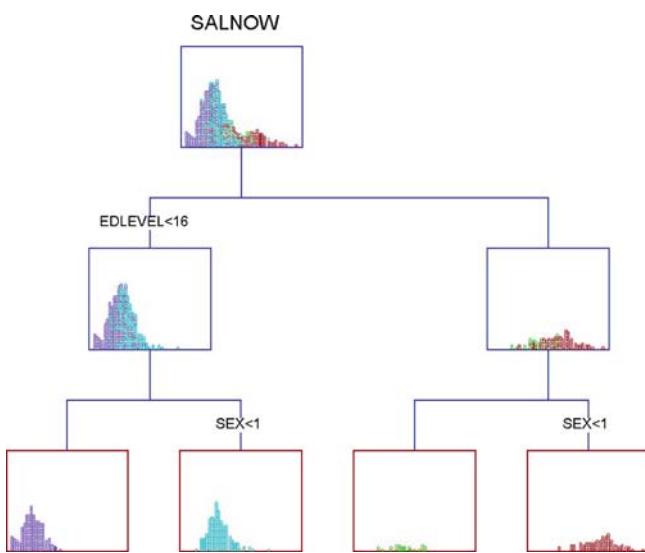


Figure II.5.6. Mobile of bank employee data

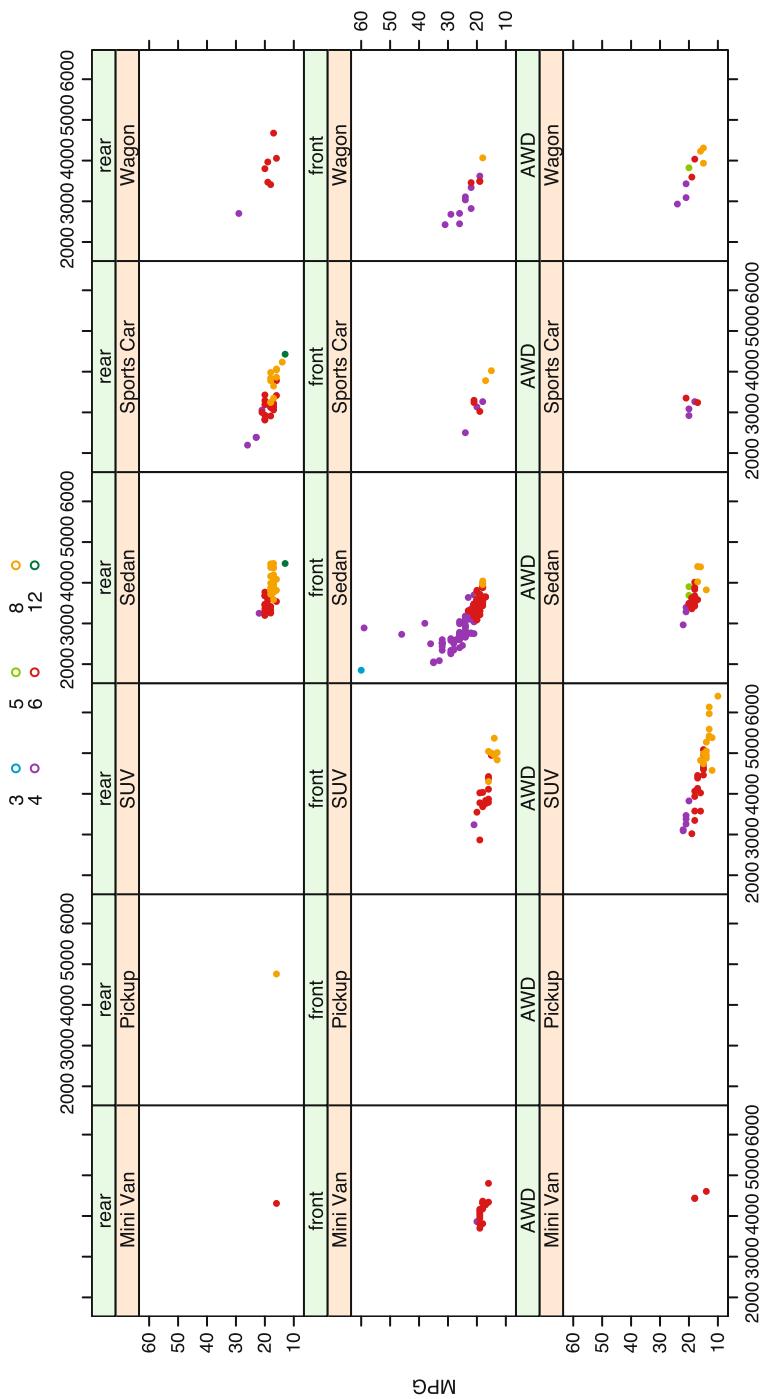


Figure II.6.5. A trellis display incorporating five variables of the cars data set
 weight [pounds]



Figure II.7.4. Exchange rate data using the original ordering of dimensions and then ordered by the first data record. Significant features in the ordered version, such as the sudden rise in value of one of the lower currencies during the third year and the progressive alignment of several of the inner currencies, are difficult to detect in the original ordering

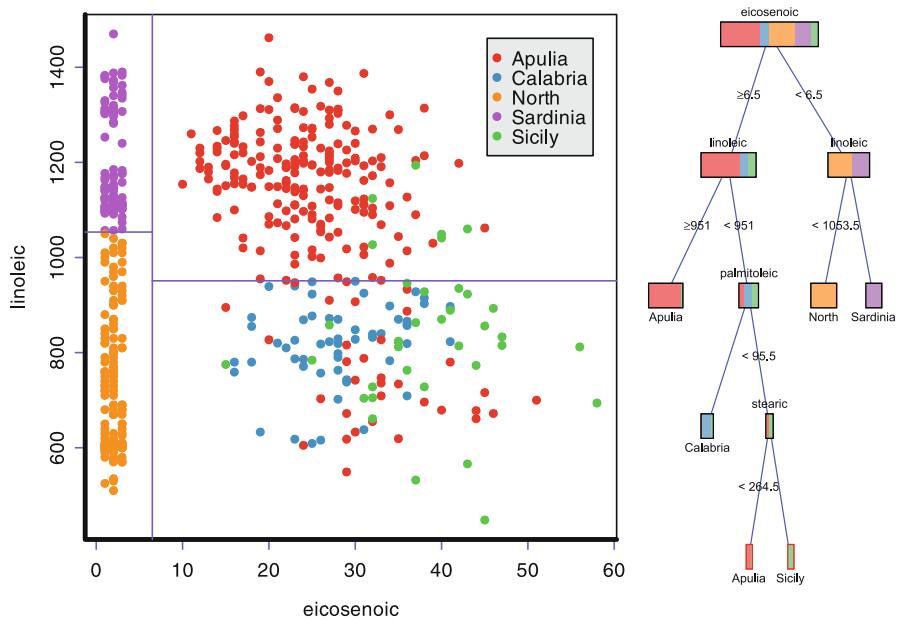


Figure II.10.3. Sectioned scatterplot (left) showing root splits and splits in its children of a classification tree (right)

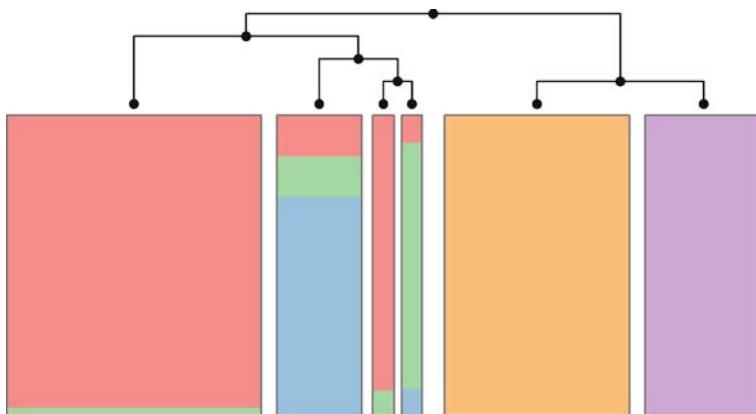


Figure II.10.7. Spineplot of leaves brushed by response categories with superimposed tree model. The associated tree is sketched on top for easier identification of individual leaves

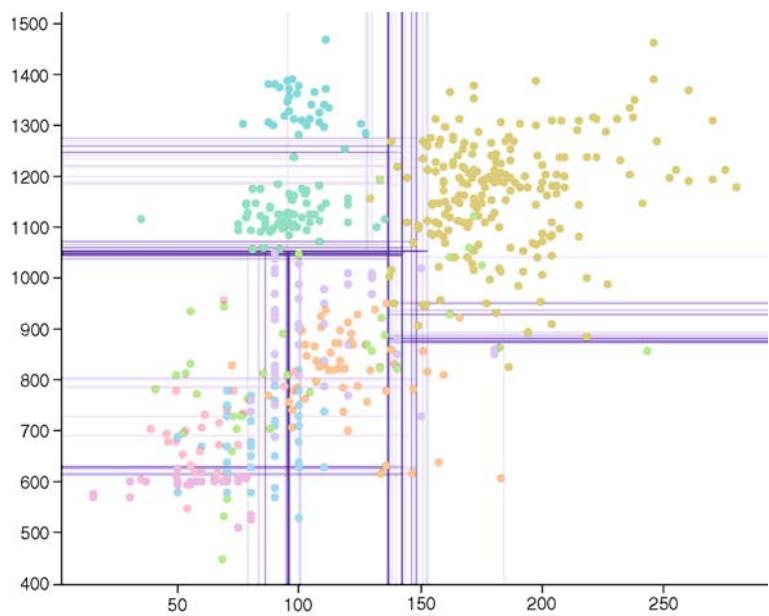


Figure II.10.12. Sectioned scatterplot of a forest of 100 trees

Soybean Statistics by State, 1997 Census of Agriculture

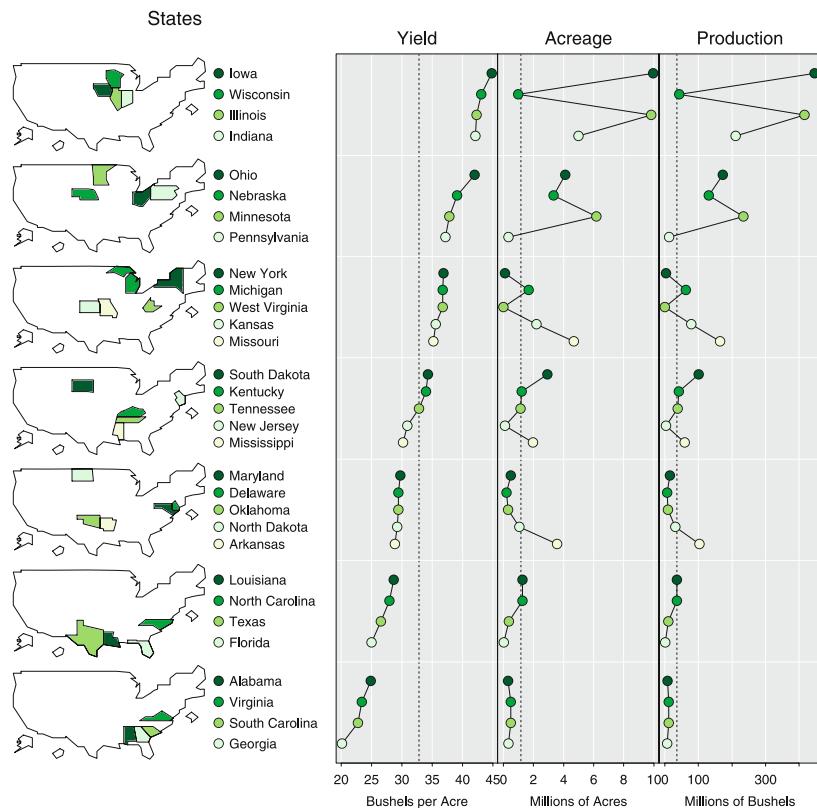


Figure III.1.3. LM plots of the 1997 Census of Agriculture, showing soybean yield (in bushels per acre), acreage (in millions of acres), and production (in millions of bushels) by state. The data are sorted by yield and show the 31 US states where soybeans were planted. The “US Average” represents the median, i.e., the value that splits the data in half such that half of the states have values below the median and the other half of the states have values above the median. For example, Tennessee is the state with the median yield. This figure has been republished from <http://www.nass.usda.gov/research/gmsoyyap.htm> without any modifications (and ideally should contain much less white space in the lower part)

White Male Lung Cancer Mortality Rates

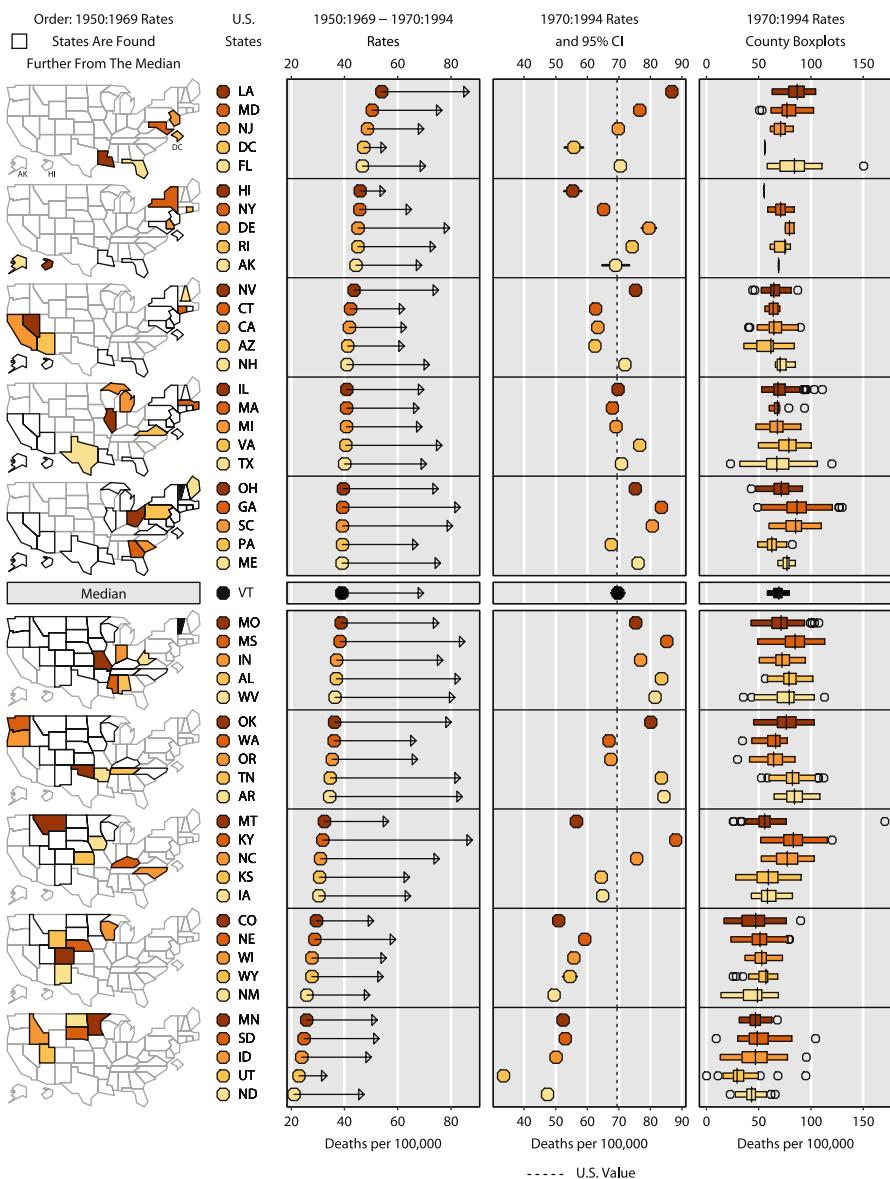


Figure III.1.6. LM plots, based on data from the NCI Web page, showing summary values for the years 1950 to 1969 and for the years 1970 to 1994 in the *left data panel*, rates and 95 % confidence intervals in the *middle data panel*, and boxplots for each of the counties of each state in the *right data panel*

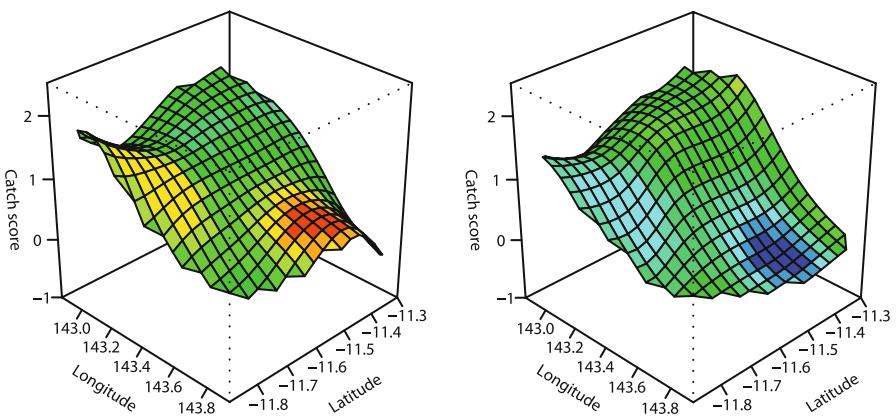


Figure III.9.8. Estimates of regression functions of catch score on latitude and longitude for two different years of data collection. Colour coding has been used to indicate the standard differences between the two surfaces

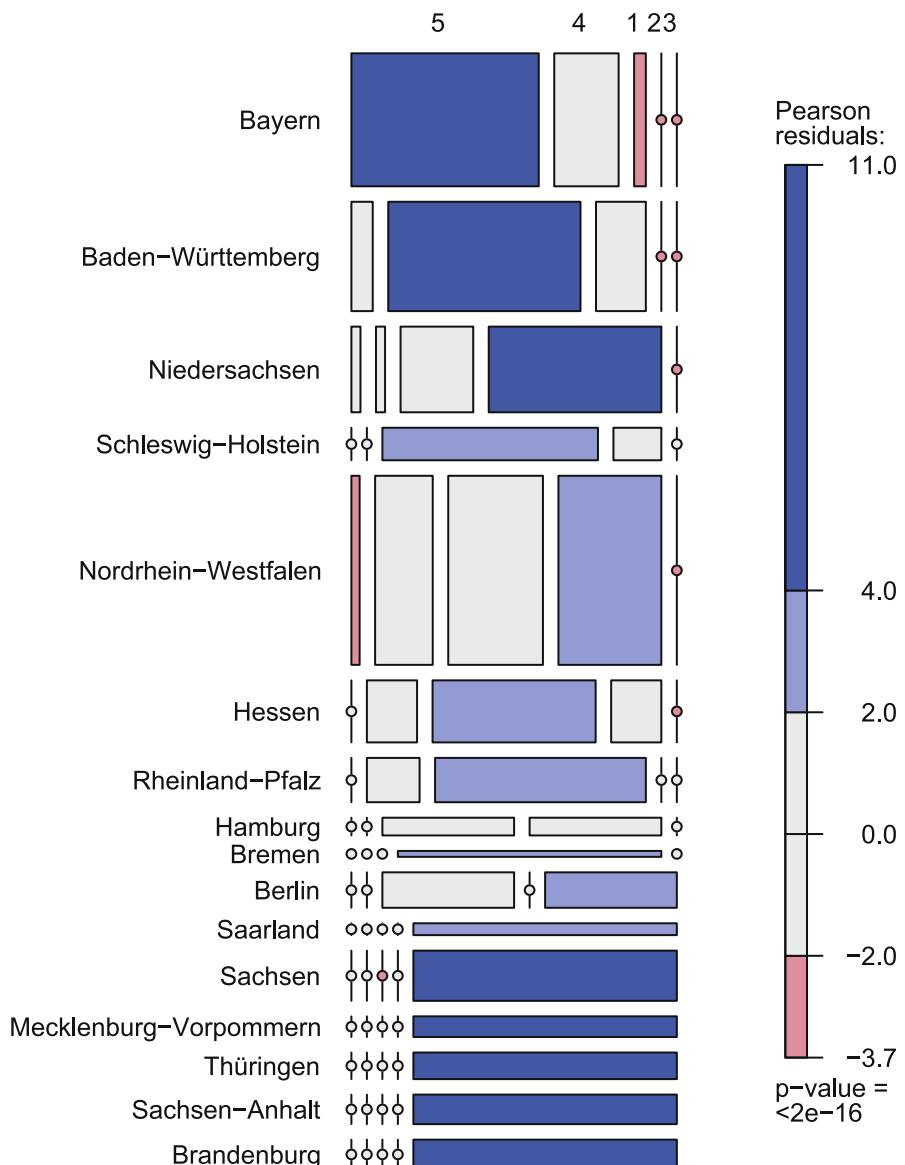


Figure III.11.11. Mosaic plot for Table 11.3

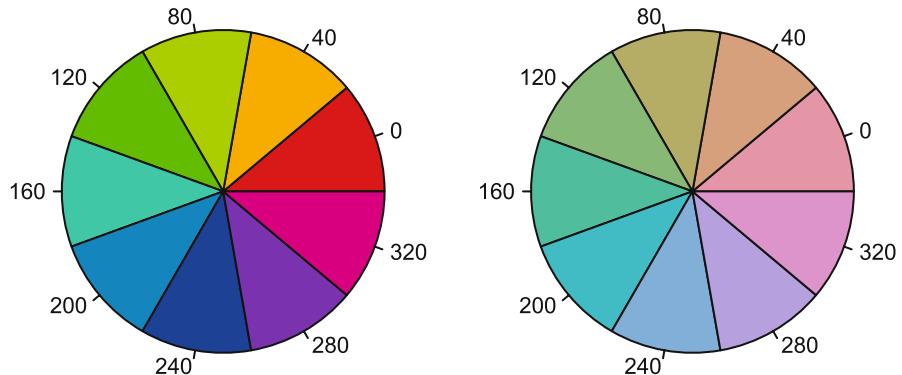


Figure III.12.8. Qualitative color palettes for the HSV (*left*) and HCL (*right*) spaces. The HSV colors are $(H, 100, 100)$ and the HCL colors $(H, 50, 70)$ for the same hues H . Note that in a monochrome version of this paper, all pies in the right wheel will be shaded with the same gray, i.e., they will appear virtually identical

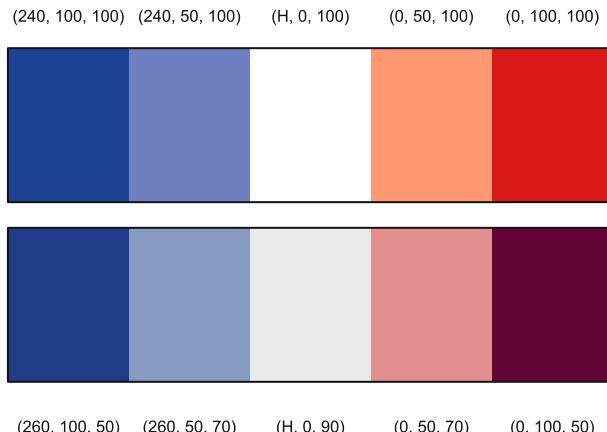


Figure III.12.9. Diverging color palettes for the HSV space (*upper part*) and the HCL space (*lower part*), ranging from blue to a neutral color to red. The triples indicate the settings for the three dimensions: (hue, saturation, value) in the upper part, and (hue, chroma, luminance) in the lower part. In a monochrome version of the paper, the right- and left-hand sides of the HCL color palette will appear to be identical, unlike the HSV color palette

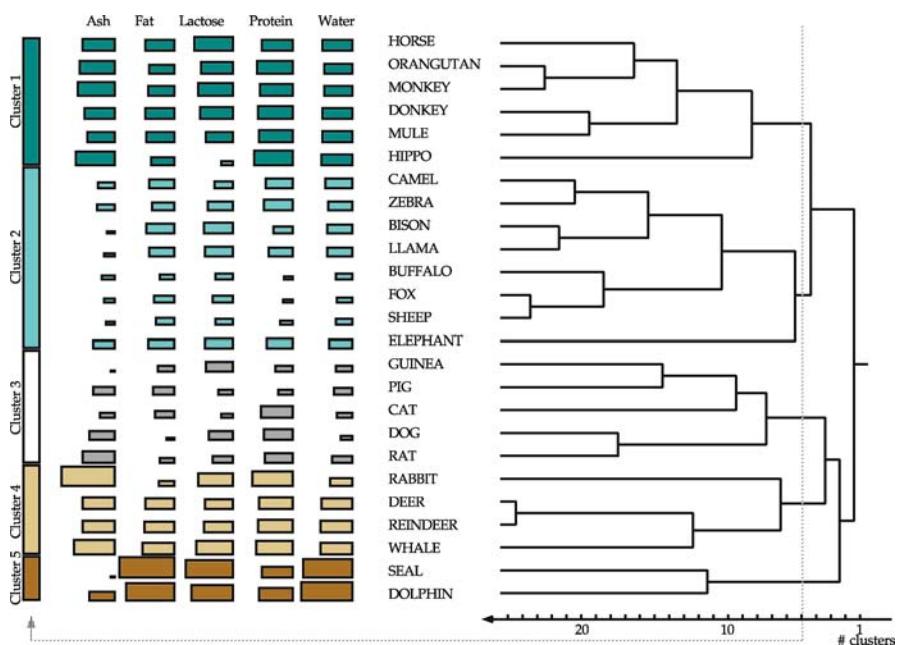


Figure III.13.12. (Weighted) mosaic plot and dendrogram of the Mammals data clustered via complete linkage

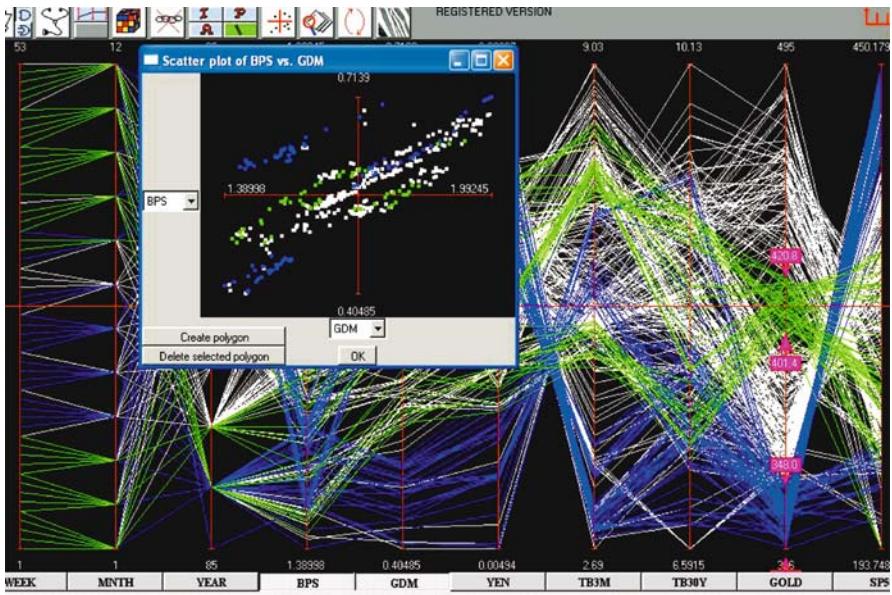


Figure III.14.16. Two price ranges for Gold. The Sterling vs. Dmark plots for them show no regularity

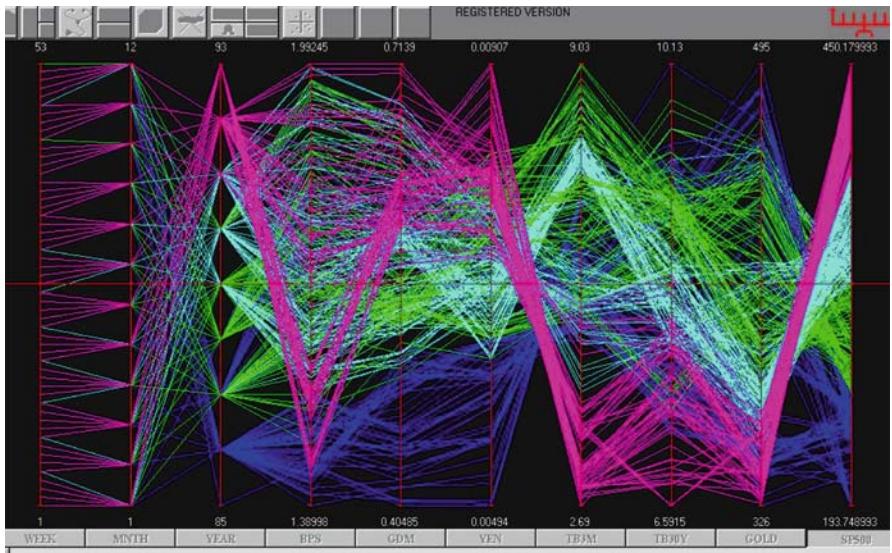


Figure III.14.19. The zebra partitions and colors the parts differently. A variable, here the SP500 axis, is divided into equal (four in this case) intervals. This quickly reveals interrelationships. In particular, note those for the highest SP500 range and review the next figure...

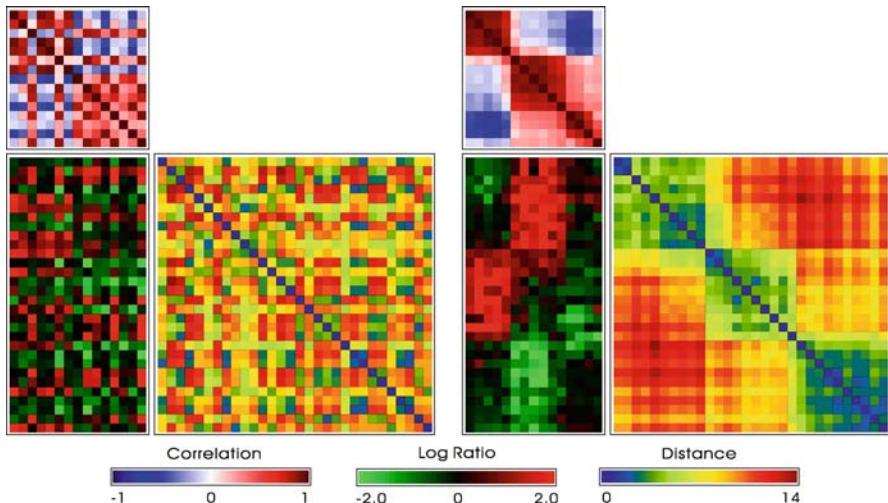


Figure III.15.1. *Left:* unsorted data matrix (log ratio gene expression) map with two proximity matrix (Pearson correlation for arrays and Euclidean distance for genes) maps for Dataset 1. *Right:* application of elliptical seriations to the three matrix maps on the left panel

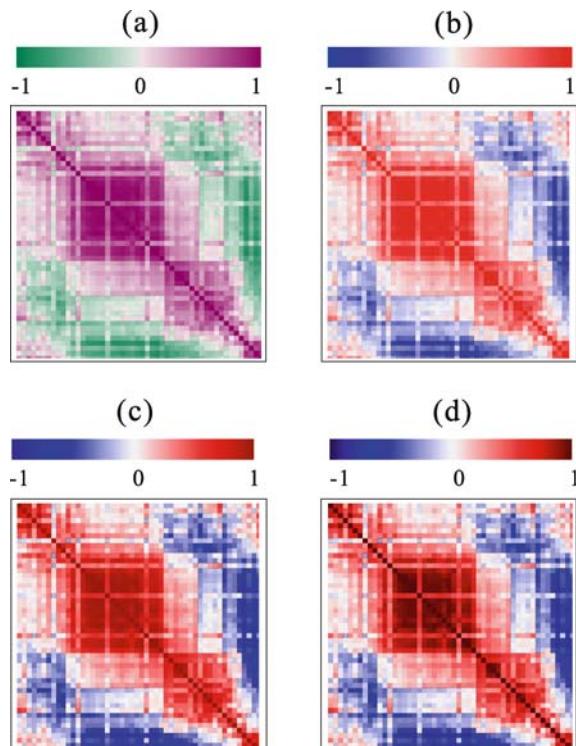


Figure III.15.2. Four color spectra applied to the same correlation matrix map for fifty psychosis disorder variables (Chen, 2002)

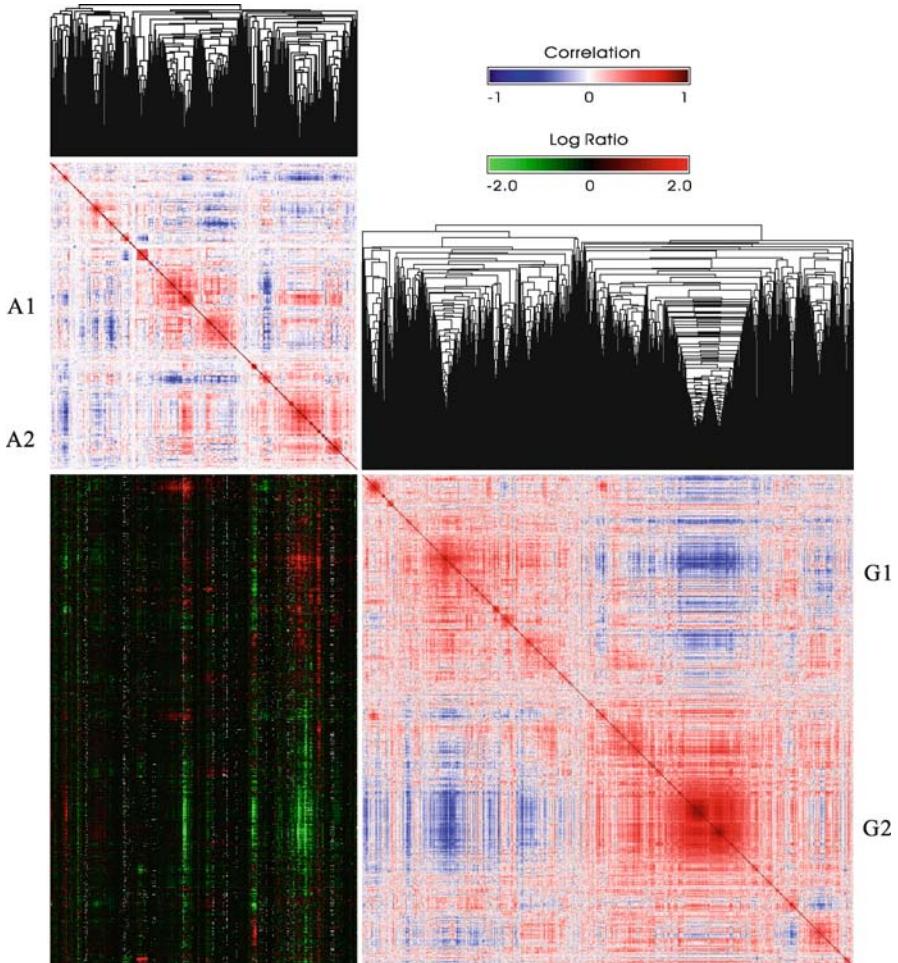


Figure III.15.10. Data matrix map (\log_2 ratio gene expression) with two proximity matrix maps (Pearson correlation for both genes and arrays) for Dataset 2 permuted by two average linkage trees (for genes (*rows*) and arrays (*columns*))

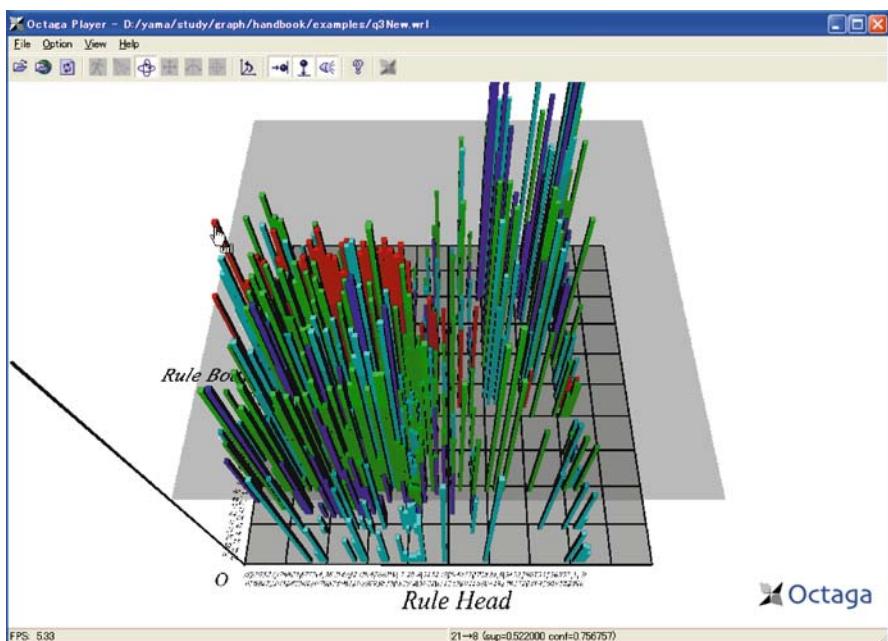


Figure III.18.16. A representation of association rules

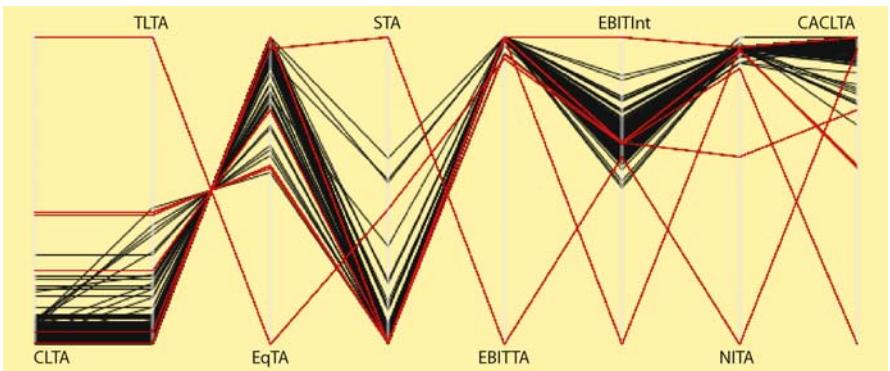


Figure IV.3.8. Parallel coordinate plot of financial ratios with skewed distributions. Seven outliers have been selected

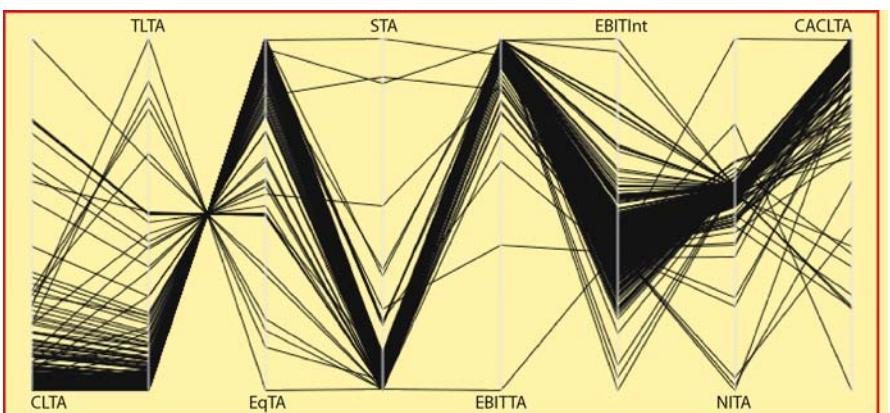


Figure IV.3.10. Parallel coordinate plot of the financial ratios with skewed distributions. The seven outliers selected in Fig. 3.8 have been removed. The plot's (red) border is a sign that not all data are displayed

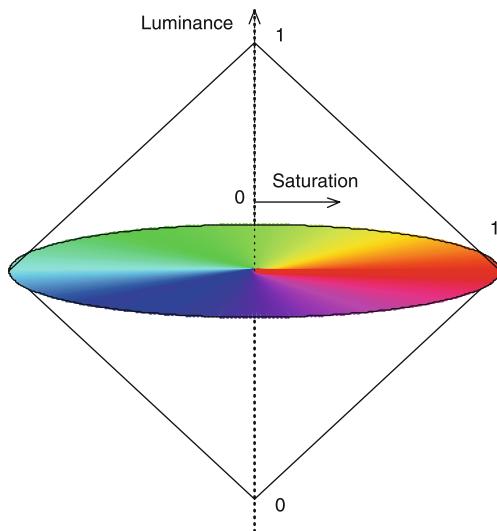


Figure IV.4.13. The luminance and saturation dimensions of the HLS colour space. We will keep luminance and saturation constant and encode PD information with the hue

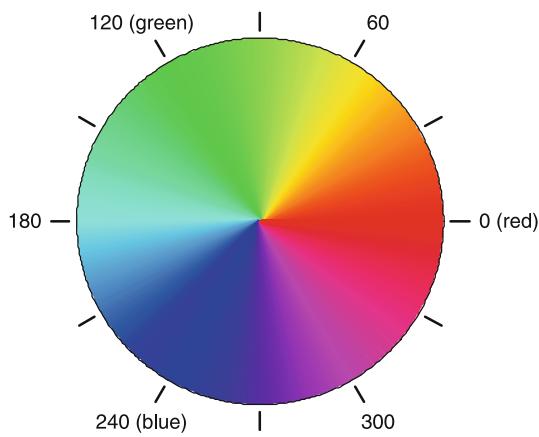


Figure IV.4.14. The hue dimension of the HLS colour space

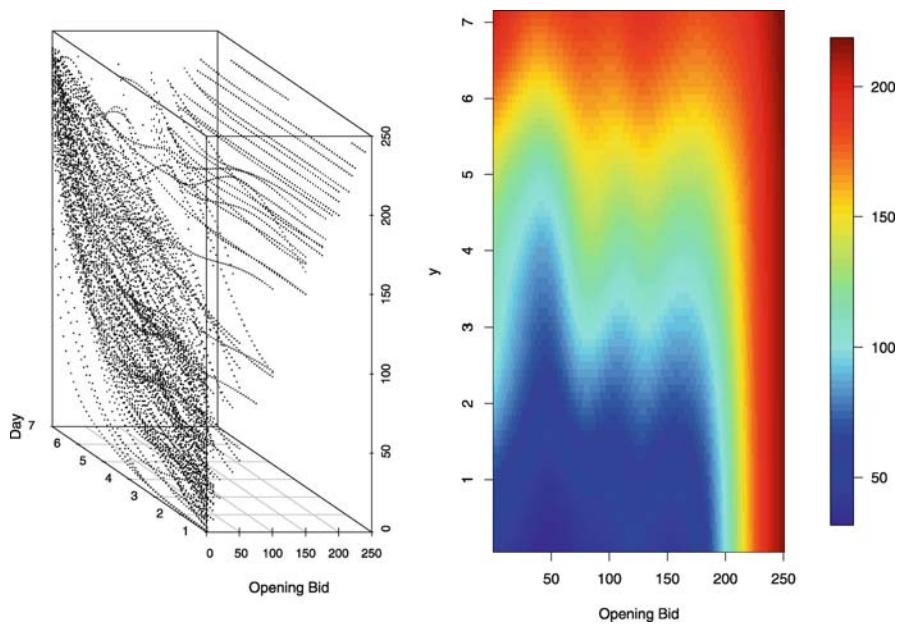


Figure IV.5.9. Relationships among functional objects: the *left panel* shows a 3-D scatterplot of opening bid (x), day of the auction (y) and price (z). The right panel shows a smoother version of the price surface, obtained using a Nadaraya–Watson smoother. In that plot, the x -axis is the opening bid and the y -axis is the day of the auction

Price Evolution Curves Vs. Calendar Time

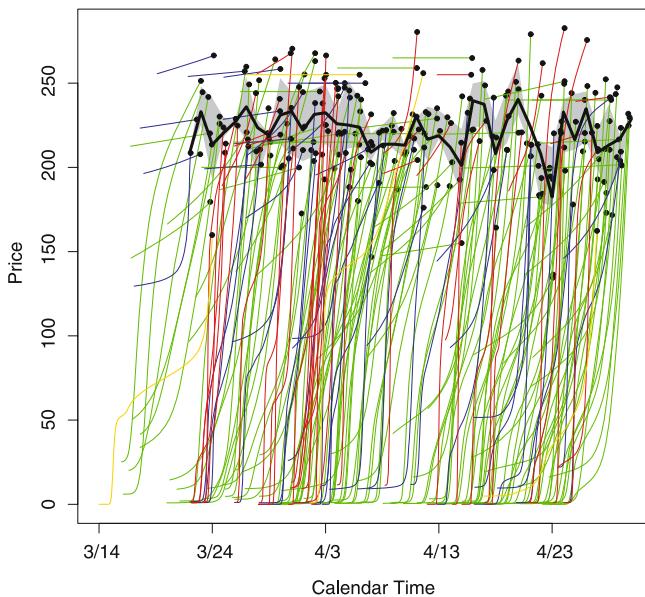


Figure IV.5.11. Rug plot displaying the price evolution (y-axis) of 217 online auctions over calendar time (x-axis) during a three-month period. The colored lines show the price path of each auction, with color indicating auction length (yellow, three days; blue, five days; green, seven days; red, ten days). The dot at the end of each line indicates the final price of the auction. The black line represents the average of the daily closing price, and the gray band is the interquartile range

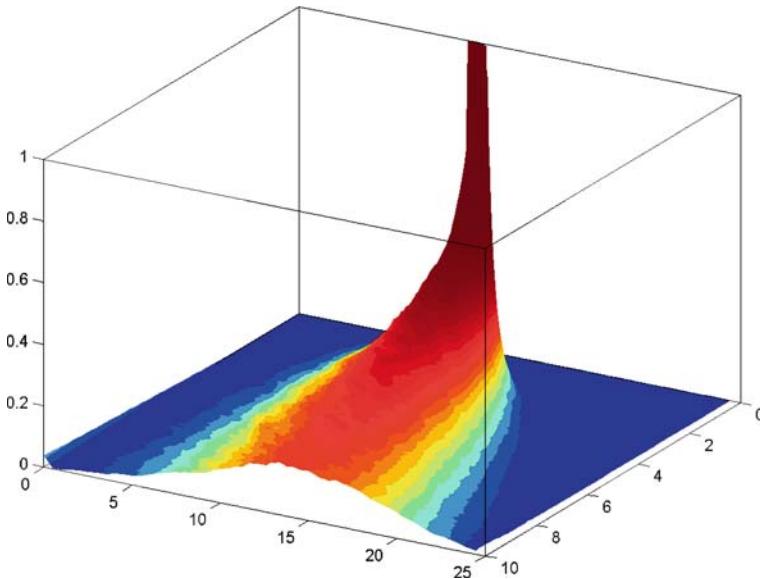


Figure IV.6.12. Three-dimensional visualization of the density evolution of a risk process with respect to the risk process value R_t (left axis) and time t (right axis). The parameters of the risk process are the same as in Fig. 6.11. From the Ruin Probabilities Toolbox

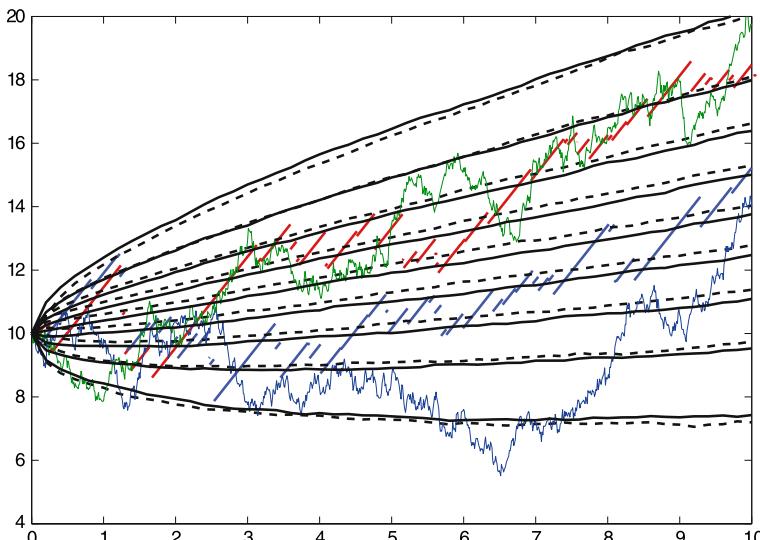


Figure IV.6.14. A Poisson-driven risk process (discontinuous thin lines) and its Brownian motion approximation (continuous thin lines). The quantile lines enable an easy and fast comparison of the processes. The thick solid lines represent the sample 0.1, ..., 0.9-quantile lines based on 10 000 trajectories of the risk process, whereas the thick dashed lines correspond to their approximation counterparts. The parameters of the risk process are the same as in Fig. 6.11. From the Ruin Probabilities Toolbox

Data Visualization via Kernel Machines

Yuan-chin Ivan Chang, Yuh-Jye Lee, Hsing-Kuo Pao, Mei-Hsien Lee,
Su-Yun Huang

10.1	<i>Introduction</i>	540
10.2	<i>Kernel Machines in the Framework of an RKHS</i>	541
10.3	<i>Kernel Principal Component Analysis</i>	543
	Computation of KPCA	544
10.4	<i>Kernel Canonical Correlation Analysis</i>	551
10.5	<i>Kernel Cluster Analysis</i>	554

Introduction

Due to the rapid development of information technology in recent years, it is common to encounter enormous amounts of data collected from diverse sources. This has led to a great demand for innovative analytic tools that can handle the kinds of complex data sets that cannot be tackled using traditional statistical methods. Modern data visualization techniques face a similar situation and must also provide adequate solutions.

High dimensionality is always an obstacle to the success of data visualization. As well as this problem, explorations of the information and structures hidden in complicated data can be very challenging. Parametric models, on the one hand, are often inadequate for complicated data; on the other hand, traditional nonparametric methods can be far too complex to implement in a stable and affordable way due to the “curse of dimensionality.” Thus, the development of new nonparametric methods for analyzing massive data sets is a highly demanding but important task. Following the recent successes in many fields of machine learning, kernel methods (e.g., Vapnik, 1995) can certainly provide us with powerful tools for such analyses. Kernel machines facilitate the flexible and versatile nonlinear analysis of data in a very high-dimensional (often with infinite dimensions) reproducing kernel Hilbert space (RKHS). The rich mathematical theory as well as topological and geometric structures associated with reproducing kernel Hilbert spaces enable probabilistic interpretation and statistical inference. They also provide a convenient environment that is suitable for massive computation.

In many classical approaches, statistical procedures are carried out directly on sample data in Euclidean space \mathbb{R}^p . In kernel methods, data are first mapped to a high-dimensional Hilbert space via a certain kernel or its spectrum, and classical statistical procedures are then applied to these kernel-transformed data. Kernel transformations provide us with a new way of specifying a “distance” or “similarity” metric between different elements.

After preparing the raw data in kernel form, standard statistical and/or mathematical software can be used to explore nonlinear data structures. For instance, we can perform nonlinear dimension reduction by kernel principal component analysis (KPCA), which can be used to construct high-quality classifiers as well as to provide new angles of view in data visualization. That is, we are able to view the more complicated (highly nonlinear) structures of massive data sets without the need to overcome the computational difficulties of building complex models. Many multivariate methods can also be extended to cover highly nonlinear cases through a kernel machine framework.

In this article, by combining the classical methods of multivariate analysis – such as PCA, canonical correlation analysis (CCA) and cluster analysis – with kernel machines, we introduce their kernelized counterparts, which enable more versatile and flexible data visualization.

Kernel Machines in the Framework of an RKHS

The goal of this section is twofold. First, it serves as an introduction to some basic RKHS theory that is relevant to kernel machines. Secondly, it provides a unified framework for kernelizing some classical linear methods, such as PCA, CCA, support vector clustering (SVC), etc., to allow for nonlinear structure exploration. For further details, we refer the reader to Aronszajn (1950) for the theory of reproducing kernels and reproducing kernel Hilbert spaces and Berlinet and Thomas-Agnan (2004) for their usage in probability, statistics and machine learning. Listed below are some definitions and basic properties.

- Let $\mathcal{X} \subset \mathbb{R}^p$ be the sample space of the data, which serves here as an index set. A real symmetric function $\kappa : \mathcal{X} \times \mathcal{X} \mapsto \mathbb{R}$ is said to be positive definite if, for any positive integer m , any sequence of numbers $\{a_1, a_2, \dots, a_m \in \mathbb{R}\}$, and points $\{x_1, x_2, \dots, x_m \in \mathcal{X}\}$, we have $\sum_{i,j=1}^m a_i a_j \kappa(x_i, x_j) \geq 0$.
- An RKHS is a Hilbert space of real valued functions on \mathcal{X} that satisfy the property that all evaluation functionals are bounded linear functionals. Note that an RKHS is a Hilbert space of pointwise-defined functions, where the \mathcal{H} -norm convergence implies pointwise convergence.
- For every positive definite kernel κ on $\mathcal{X} \times \mathcal{X}$ there is a corresponding unique RKHS, denoted by \mathcal{H}_κ , of real valued functions on \mathcal{X} . Conversely, for every RKHS \mathcal{H} there is a unique positive definite kernel κ such that $\langle f(\cdot), \kappa(x, \cdot) \rangle_{\mathcal{H}} = f(x)$, $\forall f \in \mathcal{H}$, $\forall x \in \mathcal{X}$, which is known as the reproducing property. We say that this RKHS admits the kernel κ . A positive definite kernel is also termed a “reproducing kernel.”
- A reproducing kernel κ that satisfies the condition $\int_{\mathcal{X} \times \mathcal{X}} \kappa^2(x, u) dx du < \infty$ has a countable discrete spectrum given by

$$\kappa(x, u) = \sum_q \lambda_q \phi_q(x) \phi_q(u), \text{ or } \kappa := \sum_q \lambda_q \phi_q \otimes \phi_q \text{ for short.} \quad (10.1)$$

The main idea behind kernel machines is to first map the data into an Euclidean space $\mathcal{X} \subset \mathbb{R}^p$ into an infinite-dimensional Hilbert space. Next, a particular classical statistical procedure, such as PCA, is carried out in this feature Hilbert space. Such a hybrid model of a classical statistical procedure and a kernel machine is nonparametric in nature, but when fitting the data it uses the underlying parametric procedure (for example, the PCA finds some of the main linear components). The extra effort involved is the preparation of the kernel data before they are fed into some classical procedures. Below we will introduce two different but isomorphic maps that are used to embed the underlying Euclidean sample space into a feature Hilbert space. Consider the transformation

$$\Phi : x \mapsto (\sqrt{\lambda_1} \phi_1(x), \sqrt{\lambda_2} \phi_2(x), \dots, \sqrt{\lambda_q} \phi_q(x), \dots)' . \quad (10.2)$$

Let $\mathcal{Z} := \Phi(\mathcal{X})$, with is termed the *feature space*. The inner product in \mathcal{Z} is given by

$$\Phi(x) \cdot \Phi(u) = \sum_q \lambda_q \phi_q(x) \phi_q(u) = \kappa(x, u). \quad (10.3)$$

The kernel trick (10.3) of turning inner products in \mathcal{Z} into kernel values allows us to carry out many linear methods in the spectrum-based feature space \mathcal{Z} without needing to know the spectrum Φ itself explicitly. Therefore, it makes it possible to construct nonlinear (from the Euclidean space viewpoint) variants of linear methods. Consider another transformation,

$$\gamma : \mathcal{X} \mapsto \mathcal{H}_\kappa \text{ given by } \gamma(x) := \kappa(x, \cdot), \quad (10.4)$$

which brings a point in \mathcal{X} to an element in \mathcal{H}_κ . The original sample space \mathcal{X} is thus embedded into a new sample space \mathcal{H}_κ . The map is called an Aronszajn map in Hein and Bousquet (2004). We connect these two maps (10.2) and (10.4) via $\mathcal{J} : \Phi(\mathcal{X}) \mapsto \gamma(\mathcal{X})$, given by $\mathcal{J}(\Phi(x)) = \kappa(x, \cdot)$. Note that \mathcal{J} is a one-to-one linear transformation satisfying

$$\|\Phi(x)\|_{\mathcal{Z}}^2 = \kappa(x, x) = \|\kappa(x, \cdot)\|_{\mathcal{H}_\kappa}^2 = \|\gamma(x)\|_{\mathcal{H}_\kappa}^2.$$

Thus, $\Phi(\mathcal{X})$ and $\gamma(\mathcal{X})$ are isometrically isomorphic, and these two feature representations (10.2) and (10.4) are equivalent in this sense. Since they are equivalent, mathematically there is no distinction between them. However, from a data visualization perspective, there is a difference. As the feature map (10.2) is not explicitly known, there is no way of visualizing the feature data in \mathcal{Z} . In this article, for data visualization purposes, data or extracted data features are placed in the framework of \mathcal{H}_κ . We will use the feature map (10.4) for the later KPCA and kernel canonical correlation analysis (KCCA). Since the data cluster will be visualized in the original sample space \mathcal{X} for the SVC, we will use the spectrum-based feature map (10.2) for ease.

Given data $\{x_1, \dots, x_n\}$, let us write, for short, the corresponding new data in the feature space \mathcal{H}_κ as

$$\gamma(x_j) := \gamma_j (\in \mathcal{H}_\kappa). \quad (10.5)$$

As can be seen later, via these new data representations (10.4) and (10.5), statistical procedures can be solved in this kernel feature space \mathcal{H}_κ in a parallel way using existing algorithms of classical procedures such as PCA and CCA. This is the key idea behind kernelization. The kernelization approach can be regarded, from the original sample space viewpoint, as a nonparametric method, since it adopts a model via kernel mixtures. It still has the computational advantage of keeping the process analogous to a parametric method, as its implementation only involves solving a parametric-like problem in \mathcal{H}_κ . The resulting kernel algorithms can be interpreted as running the original parametric (often linear) algorithms on kernel feature space \mathcal{H}_κ . For the KPCA and KCCA in this article, we use existing PCA and CCA codes on kernel data. One may choose to use codes from MATLAB, R, Splus or SAS. The extra programming effort involved is the preparation of data in an appropriate kernel form.

Let us discuss another computational issue. Given a particular training data set of size n , and by applying the kernel trick in (10.3), we can generate an $n \times n$ kernel data matrix according to a chosen kernel independent of the statistical algorithms to be used. We then apply the classical algorithm we are interested in, which depends only on the dot product, to the kernel matrix directly. Now the issue is that the generation of the full kernel matrix will become a stumbling block when the data size is huge due to the high computational cost (including CPU time and memory space). Moreover, the time complexity of the algorithm may depend on the size of this full kernel matrix. For example, the complexity of SVM is $O(n^3)$. To overcome these difficulties, Lee and Mangasarian (2001a) proposed the “reduced kernel” idea. Instead of using the full square kernel matrix \mathbb{K} , they randomly chose only a small portion of columns from \mathbb{K} to form a thin rectangular kernel matrix, called a reduced kernel. The use of partial columns corresponds to the use of partial kernel bases in \mathcal{H}_κ , while all data points are used for model fitting when the full rows are retained. The idea of a reduced kernel is applied to the smooth support vector machine (Lee and Mangasarian, 2001a,b), and according to their numerical experiments, the reduced kernel method can dramatically reduce the computational load and memory usage without sacrificing much of the prediction accuracy. The heuristic is that the reduced kernel method regularizes the model complexity by reducing the number of kernel bases without sacrificing the number of data points that are used in model-fitting. This idea has also been successfully applied to smooth ϵ -insensitive support vector regression (Lee et al., 2005), which is a penalized ϵ -insensitive least squares fit that results in an adaptive ridge-type support vector regression estimator. For a theoretical study of the reduced kernel method, we refer the reader to Lee and Huang (2007). A comparison study of the empirical behavior of the eigenvalues and eigenvectors of the full kernel versus the reduced kernel can also be found therein. Also note that the feature representation (10.4) conveniently allows the related optimization problems to be solved in its primal form as opposed to the dual form. Primal optimization has several advantages which are especially prominent for large-scale problems. Primal optimization directly optimizes the objective function, the solution can often be obtained in only a couple of gradient steps, and it can easily accommodate the reduced kernel approach or another low-rank approximation approach for large-scale problems. While kernel machine packages are available and conveniently included in many software packages, such as R, Matlab, etc., this reduced kernel method allows us to utilize kernel machines with reduced computational effort, especially for large data sets. In this article, the reduced kernel is adopted in conjunction with the algorithms for KPCA and KCCA. Some reference papers and Matlab code are available at <http://dmlab1.csie.ntust.edu.tw/downloads>.

Kernel Principal Component Analysis

10.3

When dealing with high-dimensional data, methods of projection pursuit play a key role. Among various approaches, PCA is probably the most basic and commonly used

for dimension reduction. As unsupervised method, it looks for an r -dimensional linear subspace with $r < p$ that carries as much information (in terms of data variability) as possible. It sequentially finds new coordinate axes that provide the “best” ways to view the data, and along which the data exhibit the most variance. The set of all of the new coordinate axes, the so-called principal components, forms the basis set for the r -dimensional subspace. It is often the case that a small number of principal components are sufficient to account for most of the relevant data structure and information. These are sometimes called the factors or latent variables of the data. In classical PCA, we try to find the leading eigenvectors by solving an eigenvalue problem in the original sample space. We refer the reader to Mardia et al. (1979) and Alpaydin (2004) for further details. Due to its nature, PCA can only find linear structures in the data. But what if we are interested in both linear and nonlinear features? Inspired by the success of kernel machines, Schölkopf et al. (1998) and Schölkopf et al. (1999) raised the idea of kernel principal component analysis. In their papers, they apply the idea of PCA to the feature data in \mathcal{Z} via the feature map (10.2). Their method allows the analysis of higher-order correlations between input variables. In practice, the transformation does not need to be specified explicitly, and the whole operation can be done by computing the dot products in (10.3). In this chapter, our formulation of KPCA is constructed in terms of its equivalent variant in the framework of RKHS \mathcal{H}_κ given by (10.4).

Actually, given any algorithm that can be expressed solely in terms of dot products (i.e., without explicit usage of the variables $\Phi(x)$ themselves), the kernel method enables us to create nonlinear versions of the given algorithm; see Aizerman et al. (1964) and Boser et al. (1992) for example. This general fact is well known to the machine learning community, and is gradually gaining popularity in the statistical community too. Here we give some examples of the application of this method to the domain of unsupervised learning, in order to obtain a nonlinear form of PCA. Some data sets from UCI Machine Learning Benchmark data archives are used for illustration.

10.3.1 Computation of KPCA

Before getting into KPCA, we briefly review the computational procedure of classical PCA. Let $X \in \mathbb{R}^p$ be a random vector with covariance matrix $\Sigma := \text{Cov}(X)$. To find the first principal component, we must find a unit vector $w \in \mathbb{R}^p$ such that the variance of the projection of X along w is maximized, i.e.,

$$\max_w w' \Sigma w \text{ subject to } \|w\| = 1. \quad (10.6)$$

This can be rewritten as a Lagrangian problem:

$$\max_{\alpha, w} w' \Sigma w - \alpha(w' w - 1), \quad (10.7)$$

where α is the Lagrange multiplier. Taking derivatives with respect to α and w and setting them to zero, we then solve for α and w . The solutions are denoted α_1 and w_1 ,

and they must satisfy $\sum w_1 = \alpha_1 w_1$, and $w_1' w_1 = 1$. Therefore, w_1 is obtained by finding the eigenvector associated with the leading eigenvalue α_1 . For the second principal component, we look for a unit vector w_2 which is orthogonal to w_1 and maximizes the variance of the projection of X along w_2 . That is, in terms of a Lagrangian problem, we solve for α_2 , w_2 and β in the following optimization formula:

$$\max_{\alpha_2, \beta, w_2} w_2' \Sigma w_2 - \alpha_2 (w_2' w_2 - 1) - \beta (w_1' w_2). \quad (10.8)$$

Using a similar procedure, we are able to find the leading principal components sequentially.

Assume for simplicity that the data $\{x_1, \dots, x_n\}$ are already centered on their mean, and so the sample covariance matrix is given by $\Sigma_n = \sum_{j=1}^n x_j x_j' / n$. By applying the above sequential procedure to the sample covariance Σ_n , we can obtain the empirical principal components.

For KPCA using the feature representation (10.4), the data mapped in the feature space \mathcal{H}_κ are $\{\gamma_1, \dots, \gamma_n\}$. The sample covariance (which is also known as a covariance operator in \mathcal{H}_κ) is given by

$$C_n := \frac{1}{n} \sum_{j=1}^n (\gamma_j - \bar{\gamma}) \otimes (\gamma_j - \bar{\gamma}), \quad (10.9)$$

where $f \otimes g$ is a linear operator defined by $(f \otimes g)(h) := \langle g, h \rangle_{\mathcal{H}_\kappa} f$ for $f, g, h \in \mathcal{H}_\kappa$. Applying similar arguments to before, we aim to find the leading eigencomponents of C_n . That is, we solve for h in the following optimization problem:

$$\max_{h \in \mathcal{H}_\kappa} \langle h, C_n h \rangle_{\mathcal{H}_\kappa} \text{ subject to } \|h\|_{\mathcal{H}_\kappa} = 1. \quad (10.10)$$

It can be shown that the solution to this is of the form $h = \sum_{j=1}^n \beta_j \gamma_j \in \mathcal{H}_\kappa$, where β_j 's are scalars. As

$$\langle h, C_n h \rangle_{\mathcal{H}_\kappa} = \sum_{i,j=1}^n \beta_i \beta_j \langle \gamma_i, C_n \gamma_j \rangle_{\mathcal{H}_\kappa} = \beta' \mathbb{K} \left(I_n - \frac{\mathbf{1}_n \mathbf{1}'_n}{n} \right) \mathbb{K} \beta / n,$$

and $\|h\|_{\mathcal{H}_\kappa}^2 = \beta' \mathbb{K} \beta$, where $\mathbb{K} = [\kappa(x_i, x_j)]$ denotes the $n \times n$ kernel data matrix, the optimization problem can be reformulated as

$$\max_{\beta \in \mathbb{R}^n} \beta' \mathbb{K} \left(I_n - \frac{\mathbf{1}_n \mathbf{1}'_n}{n} \right) \mathbb{K} \beta / n \text{ subject to } \beta' \mathbb{K} \beta = 1. \quad (10.11)$$

The Lagrangian of the above optimization problem is

$$\max_{\alpha \in \mathbb{R}, \beta \in \mathbb{R}^n} \beta' \mathbb{K} \left(I_n - \frac{\mathbf{1}_n \mathbf{1}'_n}{n} \right) \mathbb{K} \beta / n - \alpha (\beta' \mathbb{K} \beta - 1),$$

where α is the Lagrange multiplier. Taking derivatives with respect to the β 's and setting them to zero, we get

$$\mathbb{K} \left(I_n - \frac{\mathbf{1}_n \mathbf{1}'_n}{n} \right) \mathbb{K} \beta / n = \alpha \mathbb{K} \beta, \text{ or } \left(I_n - \frac{\mathbf{1}_n \mathbf{1}'_n}{n} \right) \mathbb{K} \beta = n \alpha \beta. \quad (10.12)$$

This leads to the eigenvalues–eigenvectors problem for $\left(I_n - \frac{\mathbf{1}_n\mathbf{1}'_n}{n}\right)\mathbb{K}$. Denoting its largest eigenvalue by α_1 (note that the multiplicative factor n is absorbed into the eigenvalue) and its associated eigenvector by β_1 , then the corresponding first kernel principal component is given by $h_1 = \sum_{j=1}^n \beta_{1j} \gamma_j$ in the feature space \mathcal{H}_κ . We can then sequentially find the second, third, etc., principal components. From (10.12), we have that β_k , $k = 1, 2, \dots$, are orthogonal to $\mathbf{1}_n$ and so the normalization $\beta'_k \mathbb{K} \beta_k = 1$ is equivalent to $\beta'_k \left(I_n - \frac{\mathbf{1}_n\mathbf{1}'_n}{n}\right) \mathbb{K} \beta_k = 1$. Thus, β_k is normalized according to $\alpha_k \beta'_k \beta_k = 1$.

For an $x \in \mathbb{R}^p$ and its feature image $\gamma(x) \in \mathcal{H}_\kappa$, the projection of $\gamma(x)$ along the k th eigencomponent of C_n is given by

$$\langle \gamma(x), h_k \rangle_{\mathcal{H}_\kappa} = \langle \gamma(x), \sum_{j=1}^n \beta_{kj} \gamma_j \rangle_{\mathcal{H}_\kappa} = \sum_{j=1}^n \beta_{kj} \kappa(x_j, x), \quad (10.13)$$

where β_k is the k th eigenvector of $\left(I_n - \frac{\mathbf{1}_n\mathbf{1}'_n}{n}\right)\mathbb{K}$. Therefore, the projection of $\gamma(x)$ onto the dimension reduction linear subspace spanned by the leading r eigencomponents of C_n is given by

$$\left(\sum_{j=1}^n \beta_{1j} \kappa(x_j, x), \dots, \sum_{j=1}^n \beta_{rj} \kappa(x_j, x) \right) \in \mathbb{R}^r.$$

Let us demonstrate the idea behind KPCA using a few examples. There are three data sets in this demonstration, the synthesized “two moon” data set, the “Pima diabetes” data set, and the “image segmentation” data set.

5

Example 5 First, we compare PCA and KPCA using the synthesized “two moons” data set shown in Fig. 10.1. The original data set is located in a 2-D space in (a). We can see that the two classes of data are not well-separated along any one-dimensional component. Therefore, upon applying PCA, we are not going to see good separation along the first principal coordinate axis. In the histogram (b1), the horizontal axis is the first principal coordinate from the PCA and the vertical axis is the frequency. As we can see, there is a great deal of overlap between two classes. A kernelized projection can provide a solution to this problem. In the histogram (b2), the horizontal axis is the first principal coordinate given by KPCA (with the polynomial kernel of degree 3) and the vertical axis is the frequency. Note that the KPCA with polynomial kernel also does not give good separation. However, in the histogram (b3), the results obtained from KPCA using a radial basis function (RBF, also known as the “Gaussian kernel”) with $\sigma = 1$ are shown, and they exhibit a good separation. If either PCA or KPCA is to be used as a preprocessing step before a classification task, clearly KPCA using the radial basis function with $\sigma = 1$ is the best choice. The ROC curves for these approaches are shown in (c), with the area under the curve (AUC) reported as $A_{\text{PCA}} = 0.77$, $A_{\text{KPCA(Poly)}} = 0.76$ and $A_{\text{KPCA(RBF)}} = 0.91$. KPCA with RBF ($\sigma = 1$) is clearly better at distinguishing between two groups than classical PCA and the KPCA using the polynomial kernel.

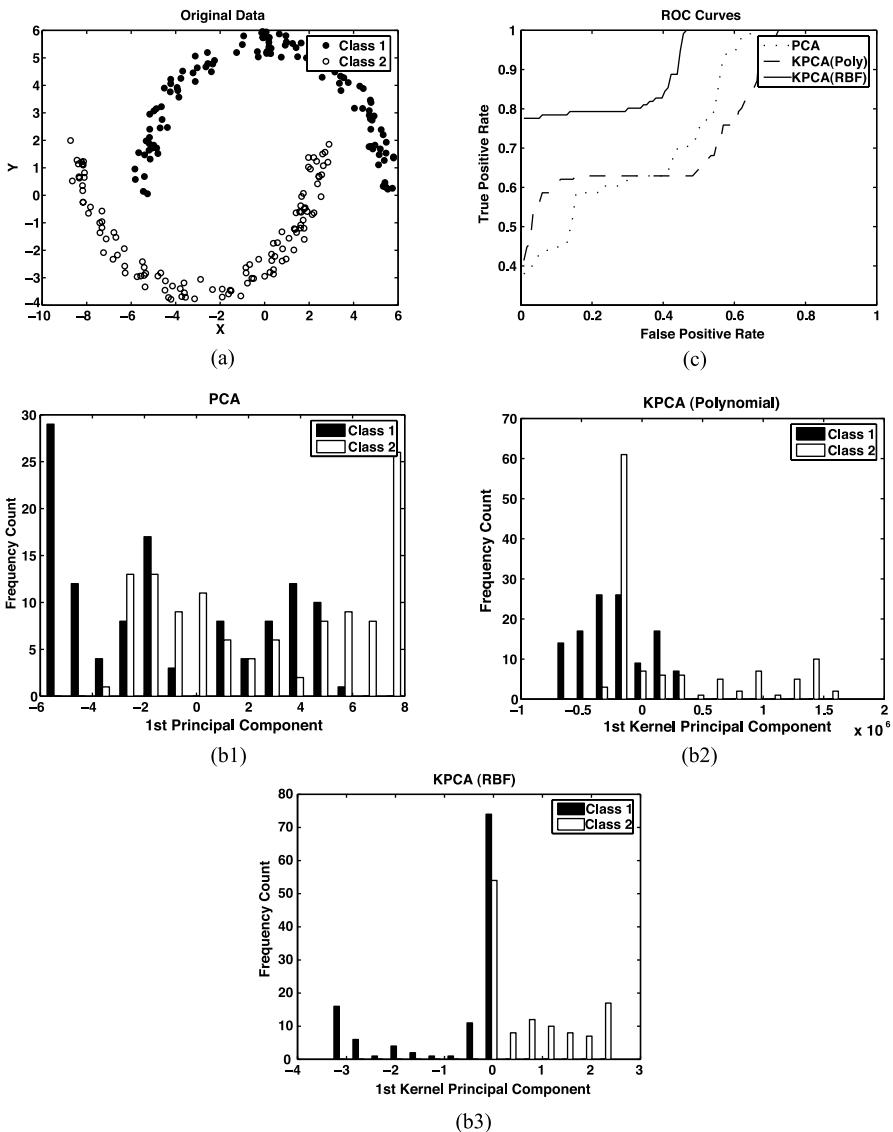


Figure 10.1. PCA and KPCA applied to a synthetic “two moons” data set: (a) original data, (b1) PCA results, (b2) results from KPCA with the polynomial kernel of degree 3, (b3) results from KPCA with the Gaussian kernel, $\sigma = 1$. In each of these histograms (b1–b3), the horizontal axis is the principal coordinate from either the PCA or KPCA and the vertical axis is the frequency. Plot (c) shows the ROC curves for these three procedures, with area under curve reported as $A_{\text{PCA}} = 0.77$, $A_{\text{KPCA}(\text{Poly})} = 0.76$ and $A_{\text{KPCA}(\text{RBF})} = 0.91$

Example 6 In this example we use the “Pima diabetes” data set from the UCI Machine Learning data archives. The reduced kernel method is adopted by randomly sampling 20% of the column vectors from the full kernel matrix. For reduced kernel PCA, we assume $\tilde{\mathbb{K}}$ is the underlying reduced kernel data matrix of size $n \times m$, where n is the data size and m is the reduced set size (i.e., column set size). KPCA using the reduced kernel is a singular value decomposition problem that involves extracting the leading right and left singular vectors $\tilde{\beta}$ and β :

$$\left(I_n - \frac{\mathbf{1}_n \mathbf{1}'_n}{n} \right) \tilde{\mathbb{K}} \tilde{\beta} = \alpha \beta, \text{ normalized to } \alpha \tilde{\beta}' \tilde{\beta} = 1 \text{ and } \alpha \beta' \beta = 1. \quad (10.14)$$

In this data set, there are nine variables, including number of times pregnant, plasma glucose concentration (glucose tolerance test), diastolic blood pressure (mm Hg), triceps skin fold thickness (mm), two-hour serum insulin (mu U/ml), body mass index (weight in kg/(height in m)²), diabetes pedigree function, age (years), and the class variable (test for diabetes), which can be positive or negative. For demonstration purposes, we use the first eight variables as input measurements and the last variable as

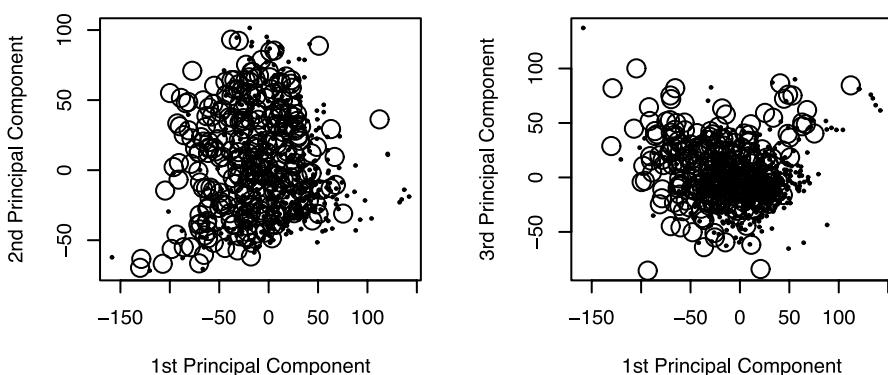


Figure 10.2. Results from PCA based on original input variables

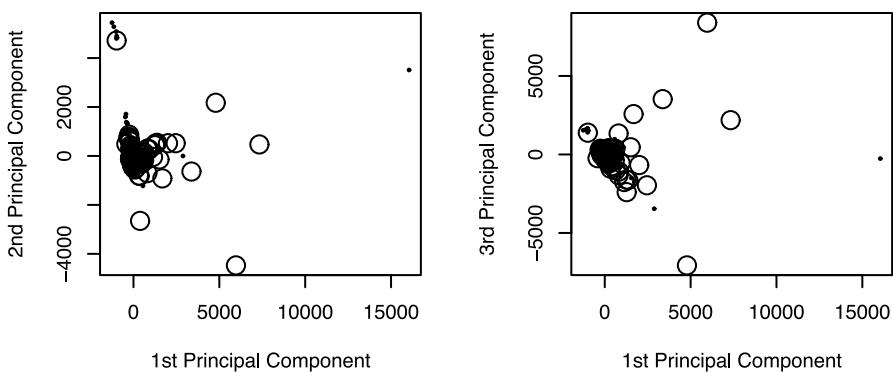


Figure 10.3. Results from KPCA with the polynomial kernel of degree 3 and scale 1

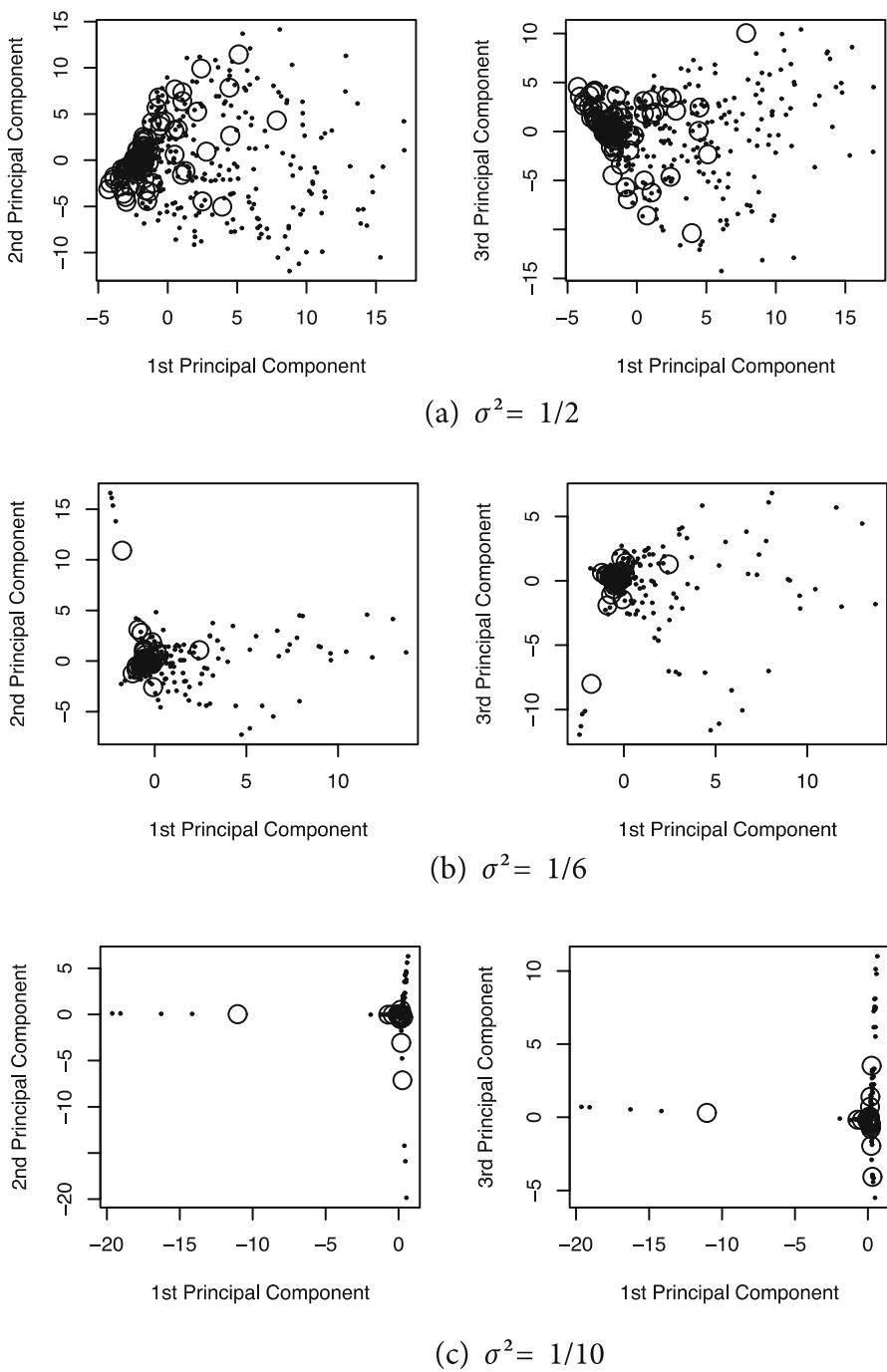


Figure 10.4. Results from KPCA with Gaussian kernels

the class variable. PCA and KPCA using both the polynomial kernel and the Gaussian kernel are carried out on all of the input measurements. In Figs. 10.2–10.4, the circles and dots denote positive and negative samples, respectively. Figure 10.2 shows the data scatter projected onto the subspace spanned by the first three principal components produced by the classical PCA. Similarly, Fig. 10.3 shows plots of data scatter projected onto the subspace spanned by the first three principal components obtained by KPCA using a polynomial kernel of degree 3 and scale parameter 1. Figures 10.4a–c are pictures of projections onto the principal component space produced by Gaussian kernels with $\sigma^2 = 1/2, 1/6$ and $1/10$, respectively. If we compare Fig. 10.2 (obtained using PCA) with the other plots, it is clear that KPCA provides some extra information about the data that cannot be obtained through classical PCA.

7

Example 7 In the third series (Fig. 10.5), we apply PCA KPCA to the “image segmentation” data set (also from the UCI Machine Learning data archives), which consists of 210 data points (each with 19 attributes) that are classified into seven classes. KPCA with the RBF gives better class separation than PCA, as can be seen in (a1) & (b1). When all seven classes are plotted in one graph, it is hard to determine the effect of

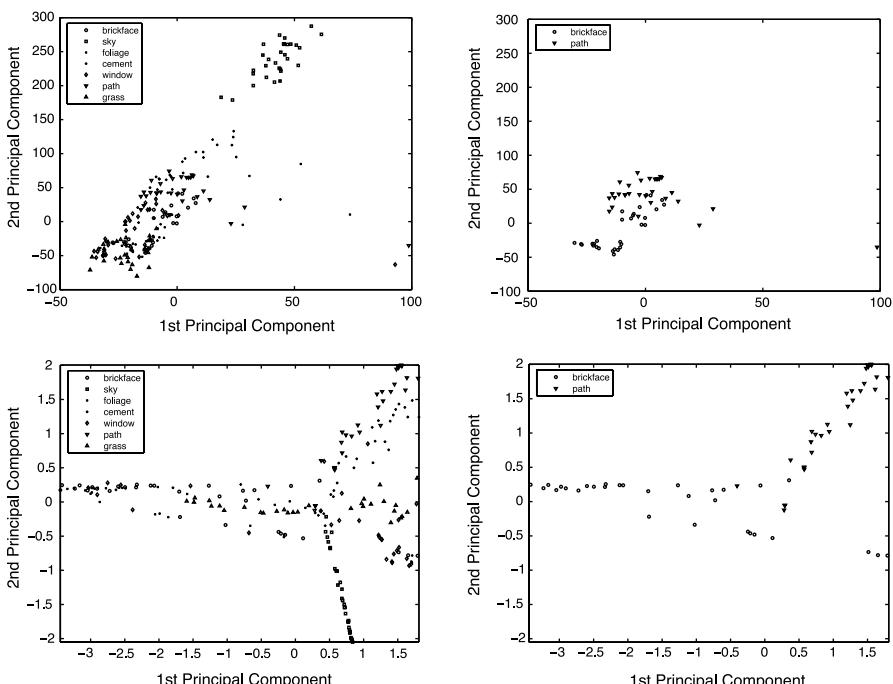


Figure 10.5. Results from applying PCA and KPCA to the “image segmentation” data set. A limited number of outliers have been omitted in the figures

KPCA. Therefore, we have also provided figures retaining only “brickface” and “path.” These clearly show that separation is produced by KPCA but not by the PCA.

Remark 1 The selection of the kernel and its window width is still an issue in general kernel methodology. There are some works that provide guidance on kernel selection for classification and supervised learning problems, but there is still a lack of guidelines in clustering and unsupervised learning problems. In this work, we merely aimed to show that nonlinear information about data can be obtained by applying kernel methods with only minor effort required. The kernel method can help to dig out nonlinear information about the data, which would be difficult or impossible to obtain by applying classical linear PCA in the original input space.

Kernel Canonical Correlation Analysis

10.4

Researchers have long been interested in describing and classifying relations between two sets of variables. Hotelling (1936) introduced canonical correlation analysis to describe the linear relation between two sets of variables that have a joint distribution. This defines a new coordinate system for each of the sets such that the new pair of coordinate systems are the best at maximizing correlations. The new systems of coordinates are simply linear systems of the original ones. Thus, classical CCA can only be used to describe linear relations. Using such linear relations, classical CCA can only find linear dimension reduction subspace and linear discriminant subspace. However, motivated by the active development and the popular and successful usage of various kernel machines, a hybrid approach combining classical CCA with a kernel machine (Akaho, 2001; Bach and Jordan, 2002), named *kernel canonical correlation analysis*, has emerged in recent years. KCCA has also been studied recently by Hardoon et al. (2004) among others.

Suppose a random vector X with p components has a probability distribution P on $\mathcal{X} \subset \mathbb{R}^p$. We partition X into

$$X = \begin{bmatrix} X^{(1)} \\ X^{(2)} \end{bmatrix},$$

with p_1 and p_2 components, respectively. The corresponding partition of \mathcal{X} is denoted by $\mathcal{X}_1 \oplus \mathcal{X}_2$. We are interested in finding relations between $X^{(1)}$ and $X^{(2)}$. Classical CCA is concerned with linear relations. It describes linear relations by reducing the correlation structure between these two sets of variables to the simplest possible form by means of linear transformations of $X^{(1)}$ and $X^{(2)}$. It finds pairs $(\alpha_i, \beta_i) \in \mathbb{R}^{p_1+p_2}$ in the following way. The first pair maximizes the correlation between $\alpha'_1 X^{(1)}$

and $\beta'_1 X^{(2)}$ subject to the unit variance constraints $\mathcal{V}(\alpha'_1 X^{(1)}) = \mathcal{V}(\beta'_1 X^{(2)}) = 1$, and the k th pair (α_k, β_k) , which is uncorrelated with the first $k - 1$ pairs, maximizes the correlation between $\alpha'_k X^{(1)}$ and $\beta'_k X^{(2)}$, and is again subject to the unit variance constraints. The sequence of correlations between $\alpha'_i X^{(1)}$ and $\beta'_i X^{(2)}$ describes only the linear relations between $X^{(1)}$ and $X^{(2)}$. There are cases where linear correlations may not be adequate for describing the “associations” between $X^{(1)}$ and $X^{(2)}$. A natural alternative, therefore, is to look for nonlinear relations. Kernel methods can provide a convenient approach to nonlinear generalization. Let $\kappa_1(\cdot, \cdot)$ and $\kappa_2(\cdot, \cdot)$ be two positive definite kernels defined on $\mathcal{X}_1 \times \mathcal{X}_1$ and $\mathcal{X}_2 \times \mathcal{X}_2$, respectively. Let \mathbb{X} denote the data matrix given by

$$\mathbb{X} = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}_{n \times p} .$$

Each data point (as a row vector) $x_j = (x_j^{(1)}, x_j^{(2)})$ in the data matrix is transformed into a kernel representation:

$$x_j \mapsto \gamma_j = (\gamma_j^{(1)}, \gamma_j^{(2)}), \quad (10.15)$$

where

$$\gamma_j^{(i)} = (\kappa_i(x_j^{(i)}, x_1^{(i)}), \dots, \kappa_i(x_j^{(i)}, x_n^{(i)})), \quad j = 1, \dots, n, \text{ and } i = 1, 2 .$$

Or, in matrix notation, the kernel data matrix is given by

$$\mathbb{K} = [\mathbb{K}_1 \mathbb{K}_2] = \begin{bmatrix} \gamma_1^{(1)} & \gamma_1^{(2)} \\ \vdots & \vdots \\ \gamma_n^{(1)} & \gamma_n^{(2)} \end{bmatrix}_{n \times 2n}, \quad (10.16)$$

where $\mathbb{K}_i = [\kappa_i(x_j^{(i)}, x_{j'}^{(i)})]_{j,j'=1}^n$, $i = 1, 2$, are the full kernel matrices for data $\{x_j^{(i)}\}_{j=1}^n$. The representation of x_j by $\gamma_j = (\gamma_j^{(1)}, \gamma_j^{(2)}) \in \mathbb{R}^{2n}$ can be regarded as an alternative way of recording data measurements with high inputs.

The KCCA procedure consists of two major steps:

- (a) Transform the data points into a kernel representation, as in (10.15) or (10.16), in matrix notation.
- (b) Apply the classical CCA procedure to the kernel data matrix \mathbb{K} . Note that some kind of regularization is needed here to solve the associated spectral problem of extracting leading canonical variates and correlation coefficients. Here we use the reduced kernel concept stated in the RKHS section and in Example 6. Only partial columns are computed to form reduced kernel matrices, denoted by $\tilde{\mathbb{K}}_1$ and $\tilde{\mathbb{K}}_2$. The classical CCA procedure is applied to the reduced kernel matrix $[\tilde{\mathbb{K}}_1 \tilde{\mathbb{K}}_2]$.

As KCCA is simply the application of classical CCA to kernel data, existing code from standard statistical packages can be utilized. In the example below we use the MATLAB m-file “canoncorr” to implement classical CCA on kernel data.

Example 8 We use the data set ‘‘pen-based recognition of hand-written digits’’ from the UCI Machine Learning data archives for a visual demonstration of nonlinear discriminant using KCCA. We use the 7494 training instances for explanatory purposes. For each instance, there are 16 input measurements (i.e., x_j is 16-dimensional) and a corresponding group label y_j from $\{0, 1, 2, \dots, 9\}$. A Gaussian kernel with a window width ($\sqrt{10S_1}, \dots, \sqrt{10S_{16}}$) is used to prepare the kernel data, where S_i ’s are the coordinate-wise sample covariances. A reduced kernel of size 300 equally stratified over ten digit groups is used and serves as the $\tilde{\mathbb{K}}_1$ in step (b) of the KCCA procedure. We use y_j , the group labels, as our \mathbb{K}_2 (no kernel transformation involved). Precisely,

$$\mathbb{K}_2 = \begin{bmatrix} Y'_1 \\ \vdots \\ Y'_n \end{bmatrix}_{n \times 10}, \quad Y'_j = (0, \dots, 1, 0, \dots),$$

where Y_j is a dummy variable for group membership. If $y_j = i$, $i = 0, 1, \dots, 9$, then Y_j has the entry 1 at the $(i+1)$ th position and 0 elsewhere. Now we want to search for relations between the input measurements and their associated group labels using CCA and KCCA. The training data are used to find the leading CCA- and KCCA-derived variates. Next, 20 test samples from each digit group are drawn randomly from the test set¹. Scatter plots of test data projected along the leading CCA-derived variates

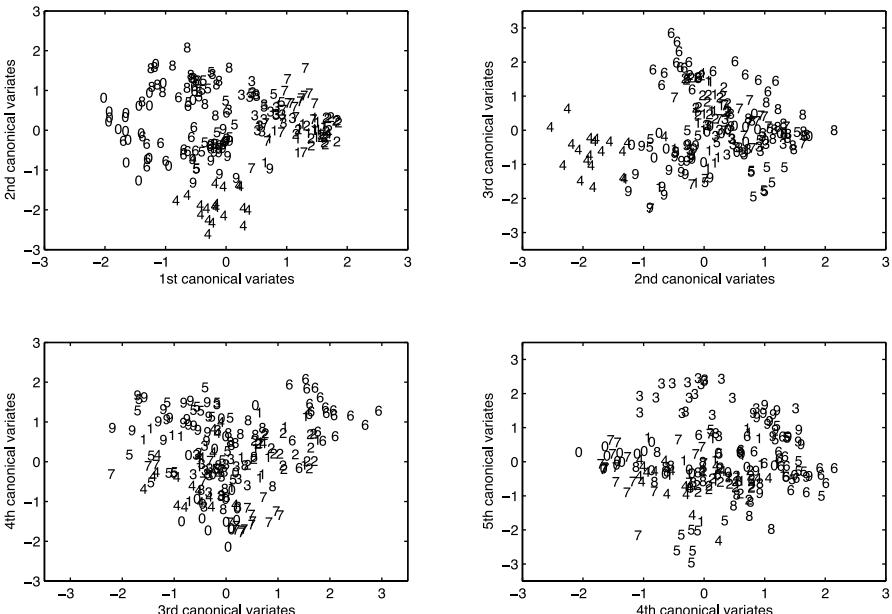


Figure 10.6. Scatter plots of pen digits over CCA-derived variates

¹ The test set has 3498 instances in total, with around 350 instances on average for each digit. For the sake of plot clarity and to avoid excess ink, we use only 20 test points per digit.

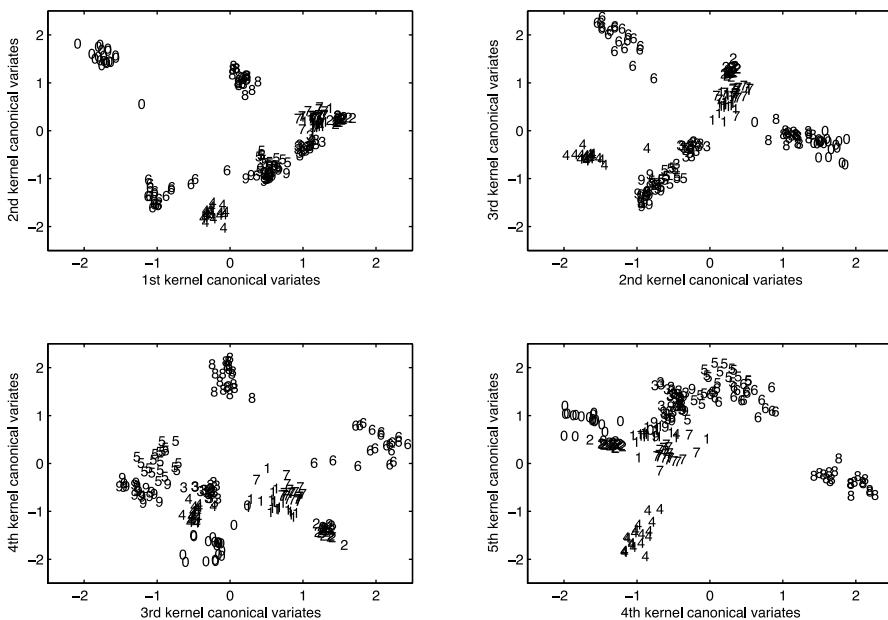


Figure 10.7. Scatter plots of pen digits over KCCA-derived variates

(Fig. 10.6) and the leading KCCA-derived variates (Fig. 10.7) are given below. Different groups are labeled with different digits. It is clear that the CCA-derived variates are not informative regarding group labels, while the KCCA-derived variates are.

10.5 Kernel Cluster Analysis

Cluster analysis is categorized as an unsupervised learning method, which tries to find the group structure in an unlabeled data set. A cluster is a collection of data points which are “similar” to points in the same cluster, according to certain criteria, and are “dissimilar” to points belonging to other clusters. The simplest clustering method is probably the k-means algorithm (which can be used in a hybrid approach with a kernel machine, or as a standalone method). Given a predetermined number of clusters k , the k-means algorithm will proceed to group data points into k clusters by (1) placing k initial centroids in the space, (2) assigning each data point to the cluster of its closest centroid, (3) updating the centroid positions and repeat the steps (1) and (2) until some stopping criterion is reached (see MacQueen, 1967). Despite its simplicity, the k-means algorithm does have some disadvantages. First, a predetermined k is necessary for the algorithm input, and different k 's can lead to dramatically different results. Secondly, suboptimal results can occur for certain

initial choices of centroid seeds. Thirdly, the algorithm may not be appropriate for some data distributions where the metric is not uniformly defined, i.e., the idea of “distance” has different meanings in different regions or for data belonging to different labels.

Here we address these issues by introducing a different clustering approach, namely *support vector clustering*, which allows hierarchical clusters with versatile clustering boundaries. Below we briefly describe the idea behind SVC, as described by Ben-Hur et al. (2001). SVC was inspired by support vector machines and kernel methods. In SVC, data points are mapped from the data space \mathcal{X} to a high-dimensional feature space \mathcal{Z} by a nonlinear transformation (10.2). This nonlinear transformation is defined implicitly by a Gaussian kernel with $\langle \Phi(x), \Phi(u) \rangle_{\mathcal{Z}} = \kappa(x, u)$. The key idea behind SVC is to find the smallest sphere in the feature space which encloses the data images $\{\Phi(x_1), \dots, \Phi(x_n)\}$. That is, we aim to solve the minimization problem

$$\min_{\mathbf{a} \in \mathcal{Z}, R} R^2, \text{ subject to } \|\Phi(x_j) - \mathbf{a}\|_{\mathcal{Z}}^2 \leq R^2, \forall j, \quad (10.17)$$

where R is the radius of an enclosing sphere in \mathcal{Z} . The Lagrangian is introduced to solve the above optimization problem. Let

$$L := R^2 - \sum_{j=1}^n (R^2 - \|\Phi(x_j) - \mathbf{a}\|^2)^2 \beta_j, \quad (10.18)$$

where $\beta_j \geq 0$ are the Lagrange multipliers. By differentiating L with respect to the primal variables R and \mathbf{a} respectively and setting the derivatives equal to zero, we have

$$\sum_j \beta_j = 1 \text{ and } \mathbf{a} = \sum_j \beta_j \Phi(x_j). \quad (10.19)$$

Moreover, the corresponding Karush–Kuhn–Tucker complementarity conditions are

$$(R^2 - \|\Phi(x_j) - \mathbf{a}\|^2)^2 \beta_j = 0, \forall j. \quad (10.20)$$

Combining (10.19) and (10.20), we can eliminate the primal variables R and \mathbf{a} and get the following dual problem:

$$\max_{\beta} W(\beta) := \sum_{j=1}^n \beta_j \|\Phi(x_j) - \bar{\Phi}\|_{\mathcal{Z}}^2 \quad (10.21)$$

$$\text{s.t. } \bar{\Phi} := \sum_j \beta_j \Phi(x_j), \quad \sum_j \beta_j = 1 \text{ and } \beta_j \geq 0.$$

That is, the SVC algorithm aims to find a weighting scheme β so that the weighted data spread $W(\beta)$ is as wide as possible.

As R is the radius of the enclosing sphere, corresponding pre-images of the enclosing sphere consist of points $\mathcal{C} := \{x : \|\Phi(x) - \bar{\Phi}\|_{\mathcal{Z}}^2 = R^2\}$. For $x \in \mathcal{C}$ we have

$$\|\Phi(x) - \bar{\Phi}\|_{\mathcal{Z}}^2 = \kappa(x, x) - 2 \sum_{j=1}^n \beta_j \kappa(x_j, x) + \sum_{j,j'=1}^n \beta_j \beta_{j'} \kappa(x_j, x_{j'}) = R^2.$$

Or equivalently

$$\mathcal{C} := \left\{ x : \sum_{j=1}^n \beta_j \kappa(x_j, x) = \rho \right\}, \quad (10.22)$$

where $\rho = (\kappa(0, 0) + \sum_{j,j'=1}^n \beta_j \beta_{j'} \kappa(x_j, x_{j'}) - R^2)/2$. When the enclosing sphere is mapped back to the data space \mathcal{X} , it forms a set of probability contours. These contours are used as cluster boundaries, and data points inside each contour are assigned to the same cluster. The SVC forms contours via kernel mixture (10.22), with the mixing coefficients β_j being solved from (10.21). Note that $\bar{\Phi}$ is a weighted centroid in the feature space and $\sum_{j=1}^n \beta_j \|\Phi(x_j) - \bar{\Phi}\|_{\mathcal{Z}}^2$ can be regarded as a weighted measure of data dispersion in the feature space. In other words, the SVC algorithm finds mixing coefficients to make the data dispersion as large as possible in the feature space \mathcal{Z} , while it draws the kernel mixture contours in the original data space \mathcal{X} to form clusters. The set \mathcal{C} defines the cluster boundaries. Data points lying on the boundaries are called support vectors. Note that the nonlinear transformation Φ is implicitly defined by a Gaussian kernel, $\kappa(x_j, x_{j'}) = e^{-q\|x_j - x_{j'}\|^2}$, $q > 0$. (The normalizing constant for κ is not relevant to cluster analysis and is dropped for simplicity.) A larger value of q corresponds to a smaller window width and leads to more clusters in the analysis.

Unlike the k-means algorithm, where the number of clusters k must be prescribed by the user, the window width in SVC can vary continuously, resulting in hierarchical clusters. The number of clusters depends on the window width of the Gaussian kernel. Decreasing the width leads to more clusters. Also, in contrast with the k-means procedure, no initial centroids are required as the algorithm input. Therefore, a deterministic result, independent from initial conditions, can be expected. SVC also has the ability to deal with outliers by employing slack variables. It allows some data points to remain outside the enclosing sphere in the feature space. This is same as the “soft margin” idea in support vector machines. With the introduction of slack variables, the optimization problem becomes

$$\min_{\mathbf{a}, R, \xi} R^2 + C \sum_{j=1}^n \xi_j, \text{ subject to } \|\Phi(x_j) - \mathbf{a}\|_{\mathcal{Z}}^2 \leq R^2 + \xi_j, \quad \xi_j \geq 0, \quad \forall j. \quad (10.23)$$

It is straightforward to derive the corresponding dual problem:

$$\max_{\beta} W(\beta) := \sum_{j=1}^n \beta_j \|\Phi(x_j) - \bar{\Phi}\|_{\mathcal{Z}}^2 \quad (10.24)$$

$$\text{s.t. } \bar{\Phi} := \sum_j \beta_j \Phi(x_j), \quad \sum_i \beta_j = 1 \text{ and } 0 \leq \beta_j \leq C.$$

As $\langle \Phi(x_j), \Phi(x_{j'}) \rangle_{\mathcal{Z}} = \kappa(x_j, x_{j'})$, the dual problem can be rewritten as a simple quadratic programming problem:

$$\max_{\beta} W(\beta) := - \sum_j \sum_{j'} \beta_j \beta_{j'} \kappa(x_j, x_{j'}) \quad (10.25)$$

$$\text{s.t. } \sum_j \beta_j = 1 \text{ and } 0 \leq \beta_j \leq C. \quad (10.26)$$

Solutions for β 's are not unique, unless the kernel matrix $\mathbb{K} = [\kappa(x_j, x_{j'})]$ is of full rank. In an optimal solution to problem (10.25), if $0 < \beta_j < C$, then $\xi_j = 0$ and the corresponding data point x_j and its image $\Phi(x_j)$ lie, respectively, on the cluster boundaries and the surface of the sphere in \mathcal{Z} . Such a point is called a *support vector* (SV). The image of a point x_j with $\xi_j > 0$ lies outside the sphere. This implies that the corresponding $\beta_j = C$. Such an x_j is called a *bounded support vector* (BSV). Data points can be classified into three types: SVs lie on the cluster boundaries, BSVs are outside the boundaries, and the other points lie inside clusters. Since $0 \leq \beta_j \leq C$ and $\sum_{j=1}^n \beta_j = 1$, there is no BSV when $C \geq 1$. Moreover, $1/(nC)$ is an upper bound on the fraction of BSVs.

In three-dimensional space, we can easily visualize the clusters once the boundaries are drawn. However, in a data space with more dimensions, it is hard to picture and determine which data points are inside a specific cluster. Thus, we need an algorithm for cluster assignment. Ben-Hur et al. (2001) introduced the *adjacency matrix* for cluster assignment. Let $R(y) := \|\Phi(y) - \bar{\Phi}\|_{\mathcal{Z}}$ be the feature distance of $\Phi(y)$ to the data centroid $\bar{\Phi}$. Denote the adjacency matrix by $A = [A_{jj'}]$, where $A_{jj'} = 1$ indicates that the pair (j, j') is in the same cluster and $A_{jj'} = 0$ otherwise. We need only the upper triangular part of the A matrix. For $j < j'$

$$A_{jj'} = \begin{cases} 1 : & \text{if } R(y) \leq R, \forall y \text{ on the line segment connecting } x_j \text{ and } x_{j'}, \\ 0 : & \text{otherwise.} \end{cases}$$

This definition is based on a geometric observation by Ben-Hur et al. For a given pair of data points, say x_j and $x_{j'}$, belonging to different clusters, any path that connects them must exit from the enclosing sphere. Therefore, such a path contains a segment of points y such that $R(y) > R$. Checking all points on a line segment is impossible. In practice, Ben-Hur et al., suggest that 10–20 uniformly distributed points should be used to check whether $R(y) > R$ or not. Once the adjacency matrix A is formed, the clusters can be defined as the connected components of the graph induced by A . This procedure will leave the BSVs as outliers. One can either assign them to the nearest clusters or leave them alone. We will illustrate an example to show how the SVC works and the effect of the window width of the Gaussian kernel.

Example 9 We synthesize 200 points in R^3 space, among which 80 points are in the upper hemisphere of the ball with radius 1.6 and with its center at the origin. The other 120 points are generated from three areas of the XY plane and then mapped into the lower hemisphere of the ball. We vary the parameter q from 1 to 7 and the number

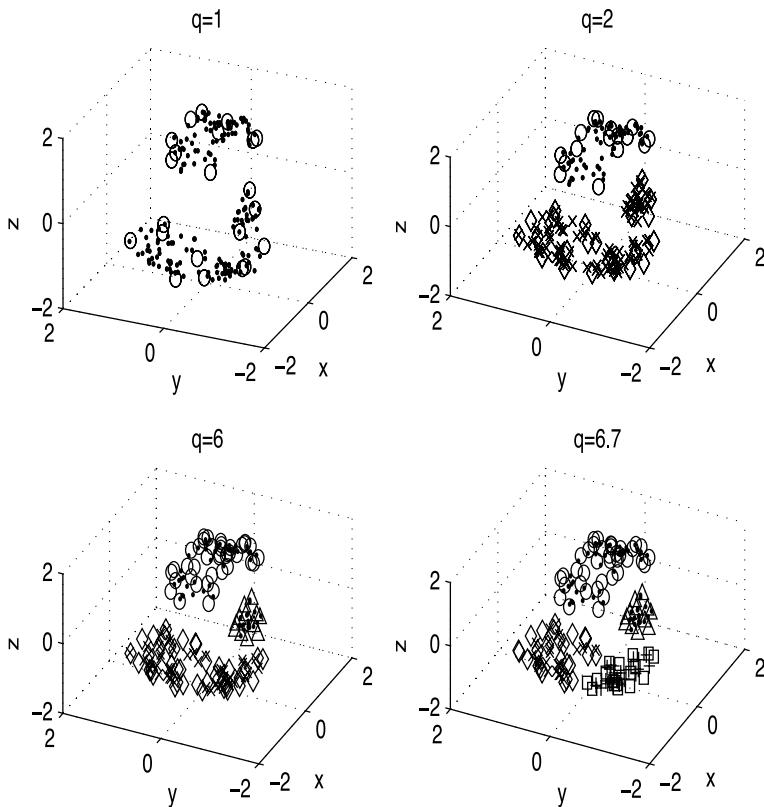


Figure 10.8. 200 points in a three-dimensional space

of resulting clusters changes from one to four. When $q = 1$, all 200 points are in one cluster, and when $q = 6.7$, we have four clusters, which is consistent with the way data is generated. The results are depicted in Fig. 10.8. Cluster membership is represented by different symbols; the circles, diamonds, squares and triangles indicate support vectors.

References

- Aizerman, M.A., Braverman, E.M. and Rozoner, L.I. (1964). Theoretical foundations of the potential function method in pattern recognition learning. *Automation and Remote Control*, 25:821–837.
- Akaho, S. (2001). A kernel method for canonical correlation analysis. In: *International Meeting of Psychometric Society (IMPS2001), 15–19 July 2001, Osaka, Japan*.
- Alpaydin, E. (2004). *Introduction to Machine Learning*. MIT Press, Cambridge.

- Aronszajn, N. (1950). Theory of reproducing kernels. *Transactions of the American Mathematical Society*, 68:337–404.
- Bach, F. and Jordan, M.I. (2002). Kernel independent component analysis. *Journal of Machine Learning Research*, 3:1–48.
- Ben-Hur, A., Horn, D., Siegelmann, H.T. and Vapnik, V. (2001). Support vector clustering. *Journal of Machine Learning Research*, 2:125–137.
- Berlinet, A. and Thomas-Agnan, C. (2004). *Reproducing Kernel Hilbert Spaces in Probability and Statistics*. Kluwer, Boston, MA.
- Boser, B.E., Guyon, I.M. and Vapnik, V.N. (1992). A training algorithm for optimal margin classifiers. In: Valient, L. and Warmuth, M. (eds) *Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory*. ACM Press, Pittsburgh, PA, 5:144–152.
- Haroon, D.R., Szedmak, S. and Shawe-Taylor, J. (2004). Canonical correlation analysis: An overview with application to learning methods. *Neural Computation*, 16(12): 2639–2664.
- Hein, M. and Bousquet, O. (2004). Kernels, associated structures and generalizations. Technical report, Max Planck Institute for Biological Cybernetics, Germany. <http://www.kyb.tuebingen.mpg.de/techreports.html>.
- Hotelling, H. (1936). Relations between two sets of variates. *Biometrika*, 28:321–377.
- Lee, Y.J., Hsieh, W.F. and Huang, C.M. (2005). ϵ -SSVR: A smooth support vector machine for ϵ -insensitive regression. *IEEE Transactions on Knowledge and Data Engineering*, 17(5): 678–685.
- Lee, Y.J. and Huang, S.Y. (2007). Reduced support vector machines: a statistical theory. *IEEE Transactions on Neural Networks*, 18:1–13.
- Lee, Y.J. and Mangasarian, O.L. (2001a). RSVM: Reduced support vector machines. In: Kumar, V. and Grossman, R. (eds) *Proceedings of the First SIAM International Conference on Data Mining*. SIAM, Philadelphia, PA.
- Lee, Y.J. and Mangasarian, O.L. (2001b). SSVM: A smooth support vector machine for classification. *Computational Optimization and Applications*, 20(1): 5–22.
- MacQueen, J.B. (1967). Some methods for classification and analysis of multivariate observations. In: Le Cam, L. and Neyman, J. (eds) *Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability*. University of California Press, Berkeley, CA, 1:281–297.
- Mardia, K.V., Kent, J.T. and Bibby, J.M. (1979). *Multivariate Analysis*. Probability and Mathematical Statistics: A Series of Monographs and Textbooks. Academic, New York.
- Schölkopf, B., Burges, C. and Smola, A. (1999). Kernel principal component analysis. *Advances in Kernel Methods – Support Vector Learning*, 327–352. MIT Press, Cambridge, MA.
- Schölkopf, B., Smola, A. and Müller, K. (1998). Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5): 1299–1319.
- Vapnik, V.N. (1995). *The Nature of Statistical Learning Theory*. Springer-Verlag, New York.

Visualizing Cluster Analysis and Finite Mixture Models

Friedrich Leisch

11.1	<i>Introduction</i>	562
	The Data Sets	562
	Software	564
11.2	<i>Hierarchical Cluster Analysis</i>	564
	Dendograms	565
	Heatmaps.....	567
11.3	<i>Partitioning Cluster Analysis</i>	567
	Convex Cluster Hulls	569
	The Voronoi Partition	570
	Neighborhood Graphs	571
	Cluster Silhouettes.....	572
	Cluster Location and Dispersion	574
	Using Background Variables	576
	Self-Organizing Maps.....	577
11.4	<i>Model-Based Clustering</i>	580
11.5	<i>Summary</i>	586

Introduction

Data visualization can greatly enhance our understanding of multivariate data structures, and so it is no surprise that cluster analysis and data visualization often go hand in hand, and that textbooks like Gordon (1999) or Everitt et al. (2001) are full of figures. In particular, hierarchical cluster analysis is almost always accompanied by a dendrogram. Results from partitioning cluster analysis can be visualized by projecting the data into two-dimensional space or using parallel coordinates. Cluster membership is usually represented by different colors and glyphs, or by dividing clusters into several panels of a trellis display (Becker et al., 1996). In addition, silhouette plots (Rousseeuw, 1987) provide a popular tool for diagnosing the quality of a partition. Some of the popularity of self-organizing feature maps (Kohonen, 1989) with practitioners in various fields can be explained by the fact that the results can be “easily” visualized.

In this chapter we provide an overview of visualization techniques for cluster analysis results. Using two real-world data sets, we explain the most important types of graphs that can be used in combination with hierarchical, partitioning and model-based cluster analysis. Many plots like dendograms, convex cluster hulls or silhouettes are specific to clustering, but we also demonstrate how graphical techniques introduced in other chapters of this handbook can be used as building blocks for cluster visualization.

The Data Sets

Two data sets are used throughout this chapter. The “dentitio” data set is used for hierarchical clustering (e.g., Hartigan, 1975). This data set gives the counts for eight kind of teeth – top-jaw and bottom-jaw counts for incisors, canines, premolars and molars – in 66 different species of animals. A subset of the raw data is listed in Table 11.1.

The second data set, which is used for partitioning and model-based clustering in Sects. 11.3 and 11.4, is related to the German parliamentary elections of September 18, 2005. A subset of the raw data is given in Table 11.2. The data consist of the proportions of the “second votes” obtained by the five parties that got elected to the Bundestag (the first chamber of the German parliament) for each of the 299 electoral districts. The “second votes” are actually more important than the “first votes” because they control the number of seats each party has in parliament. Note that the proportions do not sum to 1 because parties that did not get elected into parliament have been omitted from the table.

Before election day, the German government comprised a coalition of Social Democrats (SPD) and the Green Party (GRUENE); their main opposition consisted of the conservative party (Christian Democrats, CDU/CSU) and the Liberal Party (FDP). The latter two intended to form a coalition after the election if they gained a joint majority, so the two major “sides” during the campaign were SPD+GRUENE versus CDU/CSU+FDP. In addition, a new “party of the left” (LINKE) canvassed for

Table 11.1. The first ten observations from the dentitio data: top-jaw and bottom-jaw counts of incisors, canines, premolars and molars

	t.inc	b.inc	t.can	b.can	t.pre	b.pre	t.mol	b.mol
Opossum	5	4	1	1	3	3	4	4
Hairy tail mole	3	3	1	1	4	4	3	3
Common mole	3	2	1	0	3	3	3	3
Star nose mole	3	3	1	1	4	4	3	3
Brown bat	2	3	1	1	3	3	3	3
Silver hair bat	2	3	1	1	2	3	3	3
Pygmy bat	2	3	1	1	2	2	3	3
House bat	2	3	1	1	1	2	3	3
Red bat	1	3	1	1	2	2	3	3
Hoary bat	1	3	1	1	2	2	3	3
...								

Table 11.2. The first ten observations from the German election data: proportions of votes for the five largest parties in each electoral district

	SPD	CDU/CSU	GRUENE	FDP	LINKE
Flensburg–Schleswig	0.38	0.36	0.08	0.10	0.05
Nordfriesland–	0.36	0.41	0.06	0.10	0.04
Dithmarschen Nord					
Steinburg–	0.36	0.38	0.06	0.11	0.05
Dithmarschen Süd					
Rendsburg–Eckernförde	0.37	0.38	0.08	0.10	0.04
Kiel	0.41	0.28	0.13	0.09	0.06
Plön–Neumünster	0.39	0.36	0.08	0.09	0.04
Pinneberg	0.37	0.36	0.09	0.10	0.04
Segeberg–Stormarn Nord	0.36	0.36	0.09	0.11	0.04
Ostholstein	0.38	0.37	0.07	0.10	0.04
Herzogtum Lauenburg–	0.35	0.37	0.09	0.11	0.04
Stormarn Süd					
...					

the first time; this new party contained the descendants of the Communist Party of the former East Germany and some left-wing separatists from the SPD in the former West Germany.

A projection of the data onto the first two principal components is shown in Fig. 11.1. The point cloud in the lower left corner of Fig. 11.1 mainly correspond to districts in eastern Germany, where support for LINKE was strong, while the upper diagonal cloud mainly to districts in western Germany and contrasts the support for the two major parties: SPD (up) versus CDU/CSU (down). The final outcome of the election was CDU/CSU (226 seats in parliament); SPD (220); FDP (61); LINKE (54); GRUENE (51).

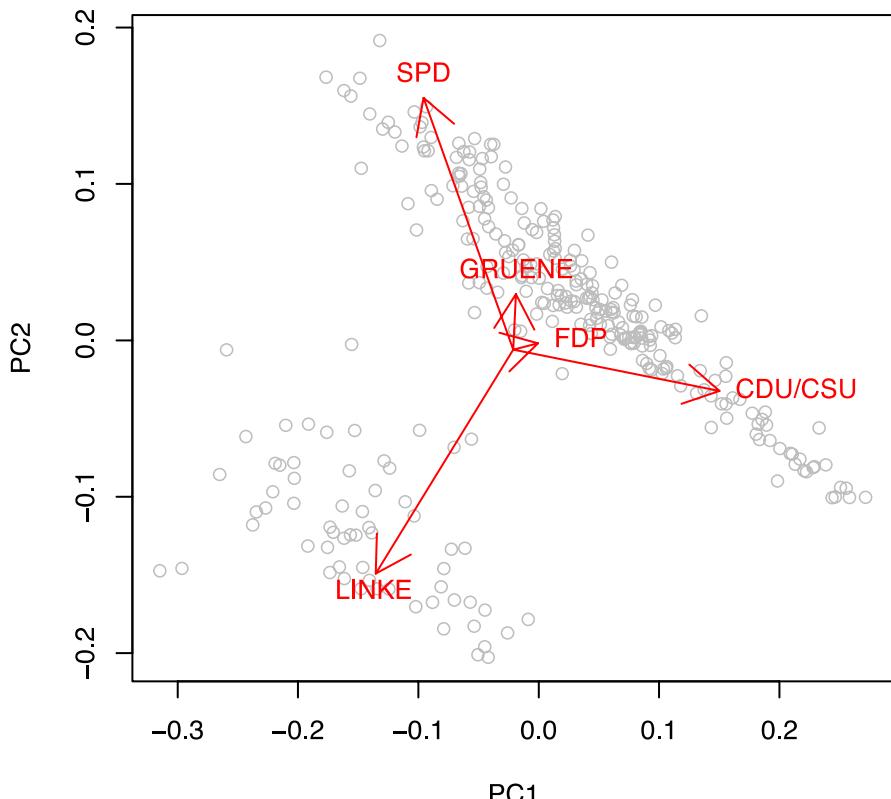


Figure 11.1. Projection of the German election data onto the first two principal components. The arrows give the projected directions of unit vectors corresponding to the five original variables

11.1.2

Software

All of the figures presented in this chapter have been created with R (R Development Core Team, 2007; Murrell, 2005) using the extension packages `cluster`, `colorspace`, `ellipse`, `flexclust`, `flexmix`, `fpc`, `gplots`, `grid`, `kohonen`, `lattice`, `mvtnorm`, and `vcd`. All of those are available from the comprehensive R archive network at <http://cran.R-project.org>. The references associated with the packages are usually given in the text when actually used. The two data sets are contained in package `flexclust`, while the R code used to generate all figures is available from the author's homepage.

11.2

Hierarchical Cluster Analysis

Hierarchical cluster methods are probably the “most intuitive” approach to grouping data, because they approach the problem in a similar way to how a human would

approach the task of dividing a set of N objects into K groups. Trivially, for $K = 1$ the only possible solution is one big cluster consisting of the complete data set X_N . Similarly, for $K = N$ we have N clusters containing only one point each, i.e., each point is its own cluster.

Divisive hierarchical clustering methods start with the complete data set X_N and (usually) split it into two groups. Each of these groups is then recursively divided into two subgroups, and so on until each point forms a cluster of its own. *Agglomerative* hierarchical clustering works the other way round, and thus starts with N singleton clusters. A hierarchy of clusters is created by repeatedly joining the two “closest” clusters until the complete data set forms one cluster.

In both cases we need to be able to measure the distances $d(\mathbf{x}, \mathbf{y})$ between d -dimensional data points \mathbf{x} and \mathbf{y} , and between groups of points. The two most common distance measures are the Euclidean distance

$$d(x, y) = \sqrt{\sum_{i=1}^d (x_i - y_i)^2}, \quad (11.1)$$

and the Manhattan distance

$$d(x, y) = \sum_{i=1}^d |x_i - y_i|. \quad (11.2)$$

Distances between groups of points are used to join (or *link*) clusters, and so these are often referred to as *linkage methods*. Assume that we have two sets A and B of points. The three simplest linkage methods for measuring the distance $l(A, B)$ between these two sets are:

Single linkage: the distance between the two closest points of the clusters

$$l(A, B) = \min_{a \in A, b \in B} d(a, b) \quad (11.3)$$

Complete linkage: the distance between the two points of the clusters that are farthest apart

$$l(A, B) = \max_{a \in A, b \in B} d(a, b) \quad (11.4)$$

Average linkage: the mean distance between the points in the two clusters

$$l(A, B) = \frac{1}{|A||B|} \sum_{a \in A} \sum_{b \in B} d(a, b) \quad (11.5)$$

Numerous other linkage methods have been invented, and many of them can be represented via the unifying framework given by Lance and Williams (1967).

Dendograms

11.2.1

The results from hierarchical clustering are typically presented as a dendrogram, i.e., a tree where the root represents the one-cluster solution (complete data set) and the

leaves of the tree are the single data points. The heights of the branches correspond to the distances between the clusters. There is no “correct” combination of distance and linkage method. Clustering in general, and especially hierarchical clustering, should be seen as exploratory data analysis, and different combinations may reveal different features of the data set.

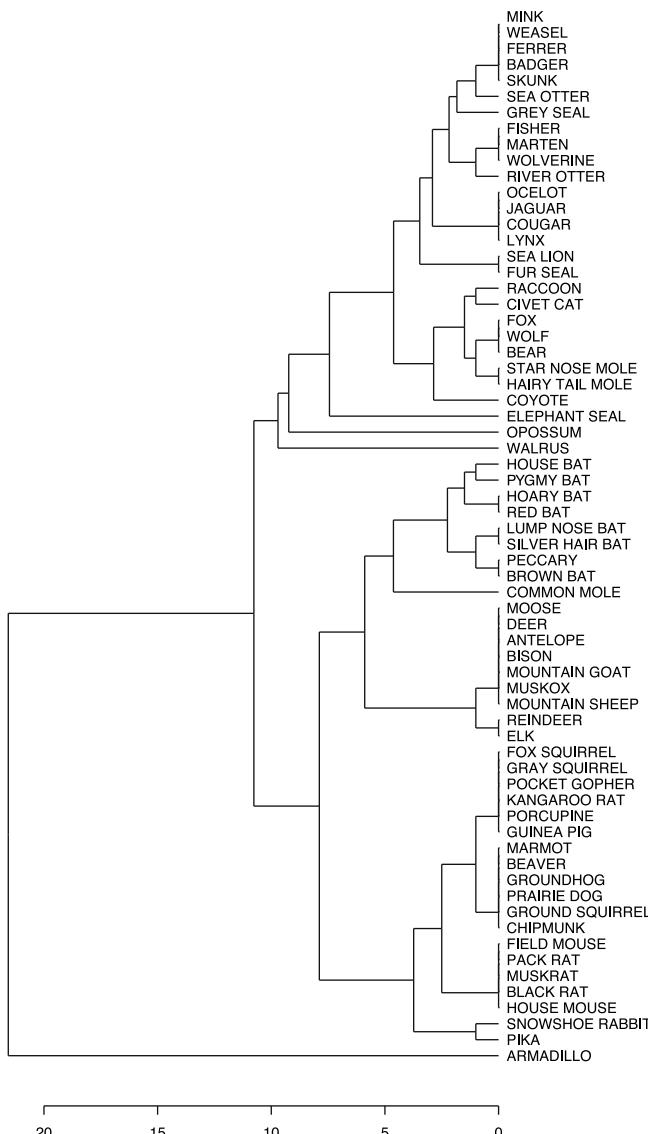


Figure 11.2. Hierarchical clustering of the dentitio data using Manhattan distance and the average linkage method

Figure 11.2 shows a dendrogram for the dentitio data created using an agglomerative algorithm utilizing Manhattan distance and the average linkage method. Manhattan distance was chosen because it has a direct interpretation for these data: it equals the difference between the teeth-group counts for two given species.

First we look for animals separated by the minimal distance. Obviously the minimum possible difference is zero, i.e., animals with identical teeth configurations like the two bats shown in the last two rows of Table 11.1. Animals separated by zero distance are depicted by vertical lines immediately to the left of their names, as shown for mink, weasel, ferret, badger and skunk at the top of the graph. After all of the animals separated by zero distance have been found and connected, those groups that are the next smallest average distance apart are joined. In our case, the next smallest distance possible is a difference of one tooth, as seen for the pygmy bat and the house bat (also shown in Table 11.1).

Note that the actual layout of a dendrogram is not unique, because at each branching point the top and bottom branches could be exchanged. For example, the armadillo, which represents an outlier here because it has eight molars and no other types of teeth, could have equally well been placed at the top of the graph. $N - 1$ branching points are needed to connect all N data points, so the total number of dendograms that could be drawn for exactly the same clustering is 2^{N-1} . This is much smaller than all possible permutations ($N!$), but is still quite a large number. Therefore, many software packages that perform hierarchical clustering allow the user to rearrange the observations, either manually or by specifying an ordering function.

Heatmaps

11.2.2

Of course, we can cluster the variables as well as the observations. For example, we might be interested in whether the animals differ more in terms of type or top/bottom jaw. At the top of Fig. 11.3 is a dendrogram for the variables sorted as they appear in the data set, i.e., with the top and bottom of each type next to each other. The original sorting of the data is compatible with hierarchical clustering of the variables (as depicted by the dendrogram), because there are no crossing lines in the tree. This leads to the (rather obvious) conclusion that the variables for the same type of teeth on the top and bottom jaws are very similar.

Figure 11.3 is a so-called *cluster heatmap*. The main part is an image plot of the original data, where each cell in the matrix corresponds to a value in the original data set. Columns and rows are permuted to conform with the hierarchical clustering of variables and observations; the corresponding dendograms are placed to the left and on top of the matrix, respectively.

Many important features of this data set can be easily picked out using the heatmap representation. The strongest patterns are the four “vertical stripes” for each of the four types of teeth, because many animals have the same (or very similar) counts on the top and bottom jaws. We can also see that the number of canines in general is rather low, while the other three tooth types show “blocks” of animals with either high or low counts. For example, the predators in the upper rows have larger incisor and premolar counts, while the rodents in the bottom rows have more molars.

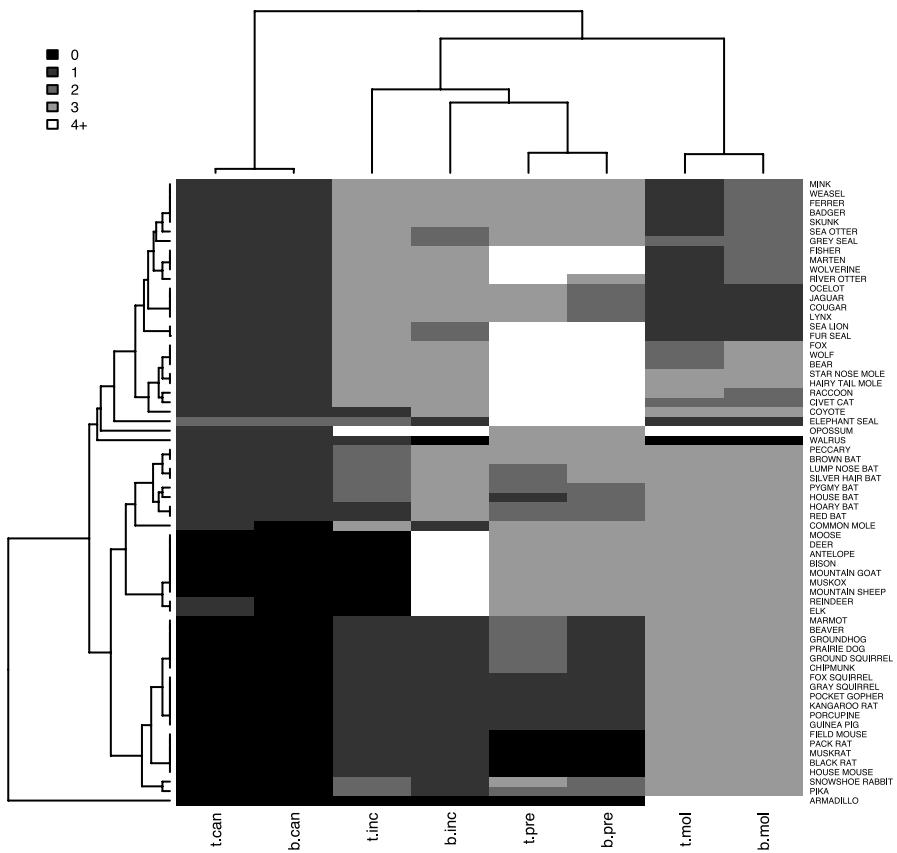


Figure 11.3. Heatmap of the dentitio data using Manhattan distance and the average linkage method for both columns and rows

Therefore, hierarchical clustering can be a valuable tool for rearranging variables and observations in a data set in order to highlight interesting patterns.

11.3

Partitioning Cluster Analysis

Partitioning cluster analysis is a complexity reduction technique. It projects data from a multidimensional (and most often metric) space to a single nominal variable, the cluster membership. The goal is to either find homogeneous subgroups in the data, which are ideally as different as possible from each other, or to impose artificial grouping on the data. In any case, we want to increase our understanding of the data using a “divide and conquer” approach that partitions a large and potentially complex data set into segments that are easier to understand or handle.

The most popular group of partitioning cluster algorithms are probably the centroid-based algorithms, like K -means (MacQueen, 1967; Hartigan and Wong, 1979) or partitioning around medoids (PAM, Kaufman and Rousseeuw, 1990). Assume that we are given a data set $X_N = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$ and a set of centroids $C_K = \{\mathbf{c}_1, \dots, \mathbf{c}_K\}$. Let $d(\mathbf{x}, \mathbf{y})$ again denote the distance between two points \mathbf{x} and \mathbf{y} , let

$$c(\mathbf{x}) = \arg \min_{\mathbf{c} \in C_K} d(\mathbf{x}, \mathbf{c}) \quad (11.6)$$

denote the centroid closest to \mathbf{x} , and let

$$A_k = \{\mathbf{x}_n | c(\mathbf{x}_n) = \mathbf{c}_k\} \quad (11.7)$$

be the set of all points where \mathbf{c}_k is the closest centroid. For simplicity of notation we remove all empty clusters such that $|A_k| > 0$, $\forall k = 1, \dots, K$. Most cluster algorithms will try to find a set of centroids C_K for fixed K such that the average distance

$$D(X_N, C_K) = \frac{1}{N} \sum_{n=1}^N d(\mathbf{x}_n, c(\mathbf{x}_n)) \rightarrow \min_{C_K}, \quad (11.8)$$

of each point to the closest centroid is minimized. However, for the following visualization techniques it is not important whether such an optimum has actually been reached, and the choice of the centroid-based cluster algorithm is also not important.

Convex Cluster Hulls

11.3.1

Once a suitable set of centroids has been found, it is usually of interest to explore how the centroids partition the input space, either in terms of the original data set or in order to enable predictions for new data. For visualization, one usually projects the data into two dimensions, using principal component analysis for example. Pison et al. (1999) use spanning ellipses to visually mark the area each cluster occupies in a PCA plot.

When clustering nonGaussian data and/or using distances other than the Euclidean distance, ellipses can be a misleading representation of cluster regions, because clusters may have arbitrary convex shapes, where the term convex relates to the distance measure used. If clusters are projected into two-dimensional space, then bagplots (Rousseeuw et al., 1999) can be used as a nonparametric alternative to ellipses.

For data partitioned using a centroid-based cluster algorithm, another natural choice is to use the distance $d(\mathbf{x}, c(\mathbf{x}))$ of each point from its respective cluster centroid to define the inner and outer cluster areas. Let

$$m_k = \text{median}\{d(\mathbf{x}_n, \mathbf{c}_k) | \mathbf{x}_n \in A_k\} \quad (11.9)$$

be the median distance of all points in cluster k to \mathbf{c}_k . We defines the *inner area* of a cluster by the convex hull of all data points where $d(\mathbf{x}_n, \mathbf{c}_k) \leq m_k$; this corresponds to the box in a boxplot. The *outer area* of a cluster is defined as the convex hull of

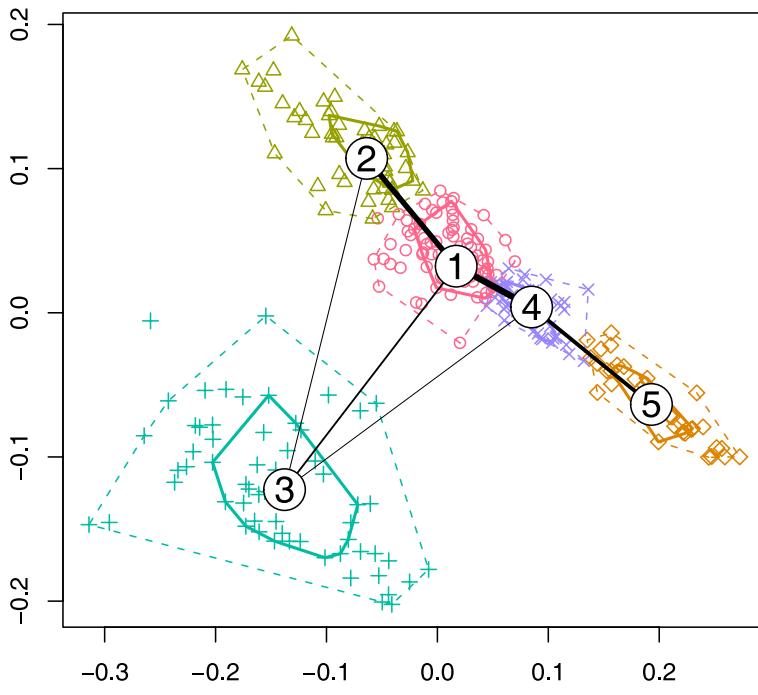


Figure 11.4. Convex hulls of the five clusters

all data points that are no more than $2.5m_k$ away from c_k ; this corresponds to the whiskers in a boxplot. Points outside this area are considered to be outliers.

Figure 11.4 shows the convex hull for a five-cluster PAM solution (Kaufman and Rousseeuw, 1990; Mächler et al., 2005) for the German election data using Euclidean distance. There seems to be no natural number of clusters in the data set in terms of within-cluster variance or cluster indices (Milligan and Cooper, 1985); the two visible groups in the PCA projection do not explain all aspects of the data. Five clusters were chosen because this solution yields good results for interpretation. In addition, model-based clustering selects five clusters based on BIC (see Sect. 11.4), so choosing the same number of clusters for PAM makes the results (and their structural differences) easier to compare.

Although the full five-dimensional data set has been clustered, the partition agrees well with the PCA projection, and there is not much overlap between the projected clusters. Cluster three captures the eastern states, while clusters two, one, four and five divide up the western states, moving from the large number of votes for the SPD (cluster two) to the large number of votes for CDU/CSU (cluster five).

11.3.2 The Voronoi Partition

If interested in both assigning the available data points to clusters and partitioning the whole input space, we can compute the so-called *Voronoi partition* which assigns

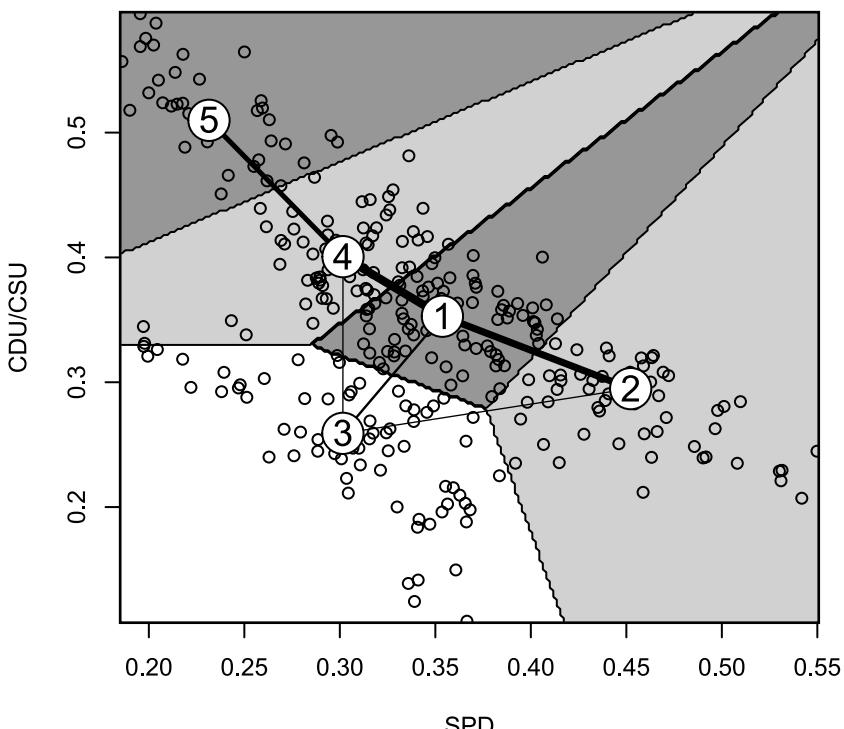


Figure 11.5. Projection of the clusters into the plane spanned by SPD and CDU/CSU, and the corresponding Voronoi partition

each point to the closest centroid. For visualization, we can use a two-dimensional slice through space, with the simplest case being planes spanned by two of the original variables. Figure 11.5 shows the clusters projected into the plane spanned by SPD and CDU/CSU, and the corresponding Voronoi partition. The plots are similar to the PCA plot in Fig. 11.4; the main differences are that the eastern states are less strongly separated from the west, and that the plot is approximately mirrored at the main diagonal: cluster five is now in the upper left corner instead of the lower right (the same idea applies for the remaining three western clusters). Note that the Voronoi partition is only valid for points in the SPD–CDU/CSU plane, so the original data points may belong to a different region in this plot compared to their original cluster membership.

Neighborhood Graphs

11.3.3

Leisch (2006) introduces a neighborhood graph of the centroids, where each centroid forms a node, and two nodes are connected by an edge if there is at least one data point

for which those two are the closest and second-closest nodes; see also Martinet and Schulten (1994). Let

$$\tilde{c}(\mathbf{x}) = \arg \min_{\mathbf{c} \in C_K \setminus \{c(\mathbf{x})\}} d(x, \mathbf{c}) \quad (11.10)$$

denote the second-closest centroid to \mathbf{x} , and let

$$A_{ij} = \{\mathbf{x}_n | c(\mathbf{x}_n) = \mathbf{c}_i, \tilde{c}(\mathbf{x}_n) = \mathbf{c}_j\} \quad (11.11)$$

be the set of all points where \mathbf{c}_i is the closest centroid and \mathbf{c}_j is the second-closest. Now the *shadow value* $s(\mathbf{x})$ for each observation \mathbf{x} is defined as

$$s(\mathbf{x}) = \frac{2d(\mathbf{x}, c(\mathbf{x}))}{d(x, c(\mathbf{x})) + d(x, \tilde{c}(\mathbf{x}))} \quad (11.12)$$

If $s(\mathbf{x})$ is close to 0, then the point is close to its cluster centroid; if $s(\mathbf{x})$ is close to 1, it is almost equidistant from the two centroids. Thus, a cluster that is well separated from all other clusters should have many points with small s values. The average shadow value of all points where cluster i is closest and j is second-closest can be used as a simple measure of cluster proximity:

$$s_{ij} = \begin{cases} |A_i|^{-1} \sum_{x \in A_{ij}} s(\mathbf{x}), & A_{ij} \neq \emptyset \\ 0, & A_{ij} = \emptyset \end{cases} \quad (11.13)$$

If $s_{ij} > 0$, then at least one data point in segment i has \mathbf{c}_j as its second-closest centroid, and segments i and j have a common border. If s_{ij} is close to 1, then most points in segment i are almost equidistant from \mathbf{c}_i and \mathbf{c}_j and the clusters are not separated very well. Finally, if s_{ij} is close to $|A_{ij}|/|A_i|$, then those points that are “between” segments i and j are almost equidistant from the two centroids. A denominator of $|A_i|$ rather than $|A_{ij}|$ is used so that a small set A_{ij} consisting of only badly clustered points with large shadow values does not induce large cluster similarity. The graph with nodes \mathbf{c}_k and edge weights s_{ij} is a directed graph; to simplify matters we use the corresponding adirected graph with average values of s_{ij} and s_{ji} as edge weights in this chapter.

Figures 11.4, 11.5, and 11.9 all contain the same graph using different projections, all of which show the linear structure of the four western clusters. The projection in Fig. 11.9 may give the misleading impression that clusters three and five overlap; the missing connection between the two nodes of the graph indicates correctly that this is an artefact of this particular projection.

11.3.4 Cluster Silhouettes

For high-dimensional data it can be hard (or even impossible) to check from any two-dimensional projection of the data whether clusters of points are well separated. From Fig. 11.4 we know that cluster three is separated from the others, but the remaining four clusters may either split into a wide continuum of electoral districts, or they may be separated from each other in a direction orthogonal to the projection.

One popular approach to partition diagnostics involves cluster silhouettes and plots of them (Rousseeuw, 1987). The basic idea is to compare the distance from each point to the points in its own cluster to the distance to points in the second-closest cluster: the silhouette value of \mathbf{x}

$$\text{sil}(\mathbf{x}) = \frac{b(\mathbf{x}) - a(\mathbf{x})}{\max(a(\mathbf{x}), b(\mathbf{x}))} \quad (11.14)$$

is defined as the scaled difference between the average dissimilarity $a(\mathbf{x})$ of \mathbf{x} to all points in its own cluster and the smallest average dissimilarity $b(\mathbf{x})$ to the points of the second-best cluster. Points with a large positive silhouette value are far from the second-best cluster, and clusters where many points have large silhouette values are

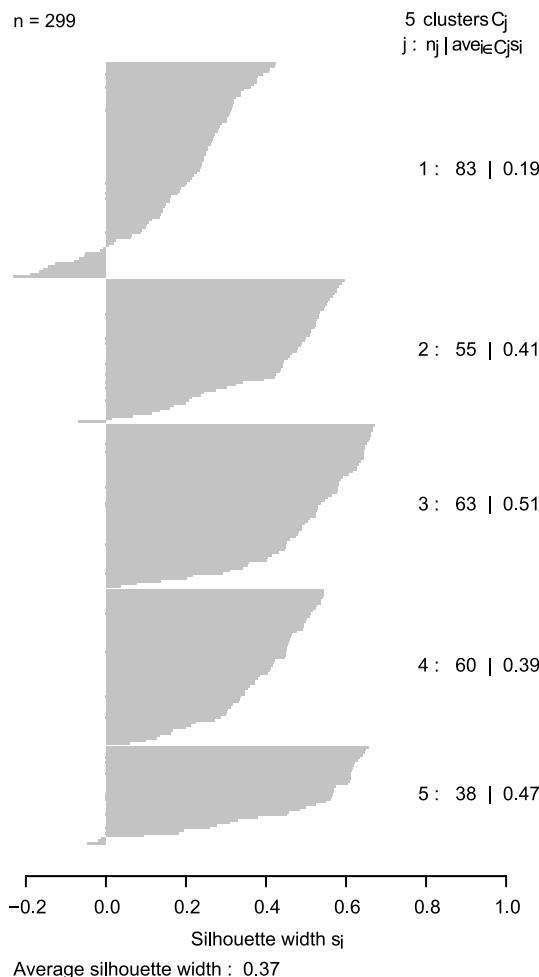


Figure 11.6. Silhouette plot for the five-cluster PAM partition of the German election data

well-separated from the other clusters. Silhouette values that are close to zero indicate points that are almost equidistant between two clusters, and points with negative silhouette values may be considered to be in the “wrong” cluster.

Figure 11.6 shows a silhouette plot for our five clusters. Not surprisingly, cluster three has the largest average silhouette value of 0.51, but cluster five is close with 0.47, which is not obvious from Fig. 11.4. In fact, the only cluster which is not well-separated from the others is cluster one, with an average silhouette of 0.19 and several points with negative values.

11.3.5

Cluster Location and Dispersion

The silhouette plot indicates that the clusters are actually more separated from each other than the projection onto the first two principal components in Fig. 11.4 suggests. Hence, we need more information on the actual location and dispersion of the clusters. We could now start to look at projections onto other principal components than the first two, scatterplot matrices of the original variables, etc. Two alternative approaches are to plot all dimensions at once or to use asymmetric projections that maximize cluster separation.

The simplest solution that uses all variables is to visualize only the cluster centroids and to ignore how the data points scatter around the centroids. Figure 11.7 shows the cluster centroids as bars in a barplot and plots the population centroid (here the mean value) as dots for comparison. This is appropriate for this kind of data because

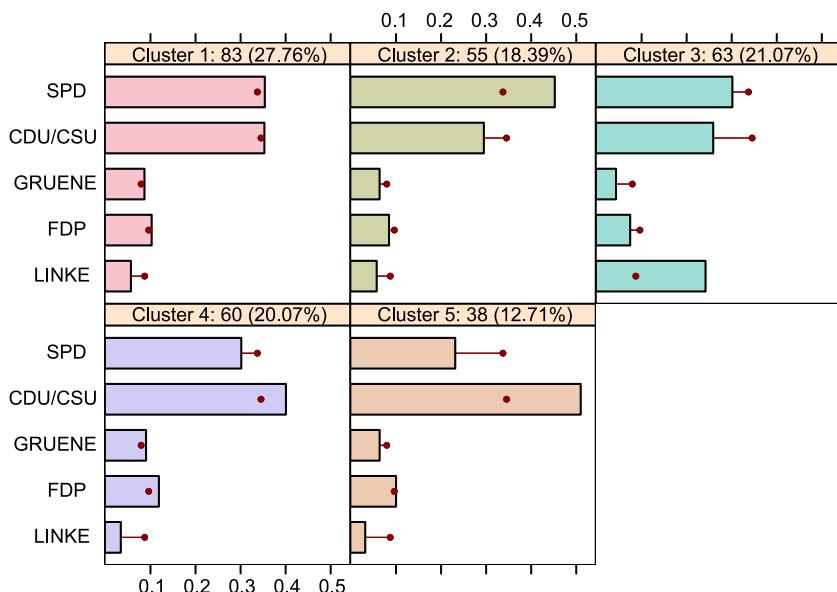


Figure 11.7. Barplot of the five-cluster medoids in comparison to the overall population mean. The numbers in the title strips of the panels give the absolute and relative number of points in each cluster

a large bar means a large number of votes. An alternative would be to use differences between cluster centroids and population centroids instead of absolute values; see also Fig. 11.13. For example, cluster one has only slightly above-average results for all parties, but the LINKE result in cluster three displays a large difference from the total LINKE mean. Note that one cannot easily test whether the mean of a cluster is significantly different from the mean value of another cluster (or the population mean), because the clusters are not independent of each other and were constructed to be as different as possible from each other. If cluster dispersion is also of interest, we could replace the bars in Fig. 11.7 by boxplots of the points in each cluster.

Parallel coordinate plots (see Chapt. III.14, Inselberg) show all variables of a high-dimensional data set in one figure. Cluster membership can be marked using different line types and colors for the clusters. This will work for data sets with relatively few observations and/or well-separated clusters with small within-cluster variation. Figure 11.8 shows a trellis display (Becker et al., 1996) of parallel coordinate plots where each panel corresponds to one cluster. This approach works well for a large number of data points and clusters, because clusters are not plotted over each other. If all clusters are plotted on the same plot, lines drawn later by the software partially mask lines drawn earlier.

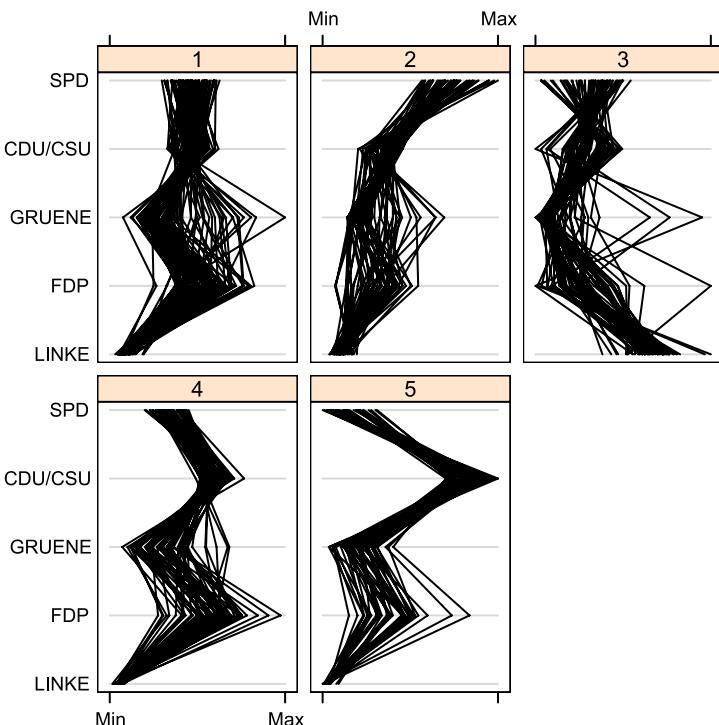


Figure 11.8. Parallel coordinate plot of the five clusters

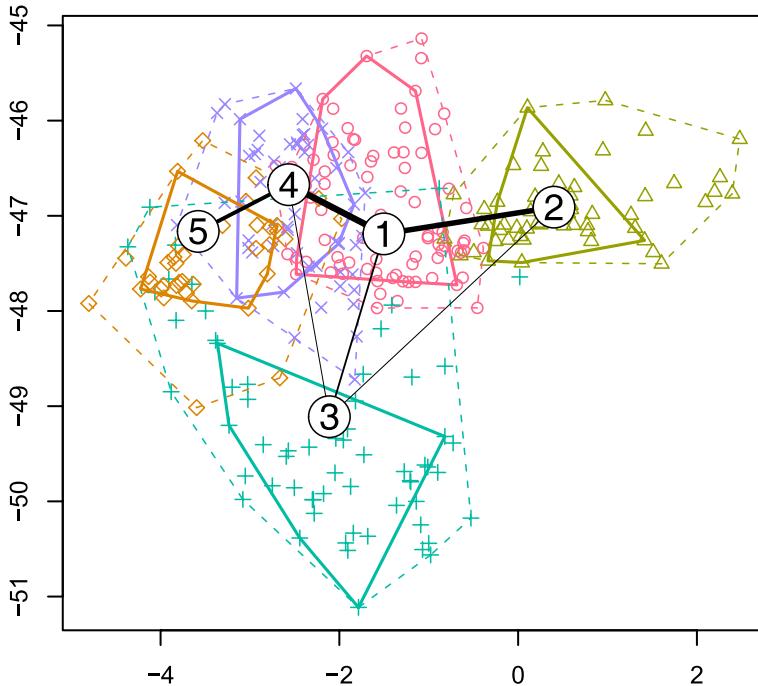


Figure 11.9. Asymmetric projection seeking to horizontally separate cluster two from the rest

One aspect of the data that can be easily seen from Fig. 11.8 is the negative correlation between SPD and CDU/CSU. In clusters two, four and five, one of these is larger than the other, while in cluster one and (much more pronounced) cluster three, the negative correlation is reflected in the X-shape of the lines between the two variables. We also see that for GRUENE there are three outliers with high numbers of votes in cluster three; these correspond to electoral districts in Berlin.

Another way to assess the differences between the clusters is to actively look for projections that maximize cluster separation, for example using the collection of methods in Hennig (2004). Figure 11.9 shows a projection using the “asymmetric neighborhood-based coordinates” method that horizontally separates cluster two from the others.

11.3.6 Using Background Variables

An important task in many clustering applications is to check whether background variables have different distributions in the clusters. By *background variables* we mean all variables in the data set that *have not been used for clustering*. If a variable has not been used for clustering, the usual statistical inference for differences between groups of observations can be used. The partition itself can be seen as a nominal variable, so all visualization techniques that can be used to plot a set of variables versus a single nominal variable are applicable.



Figure 11.10. The 16 federal states of Germany

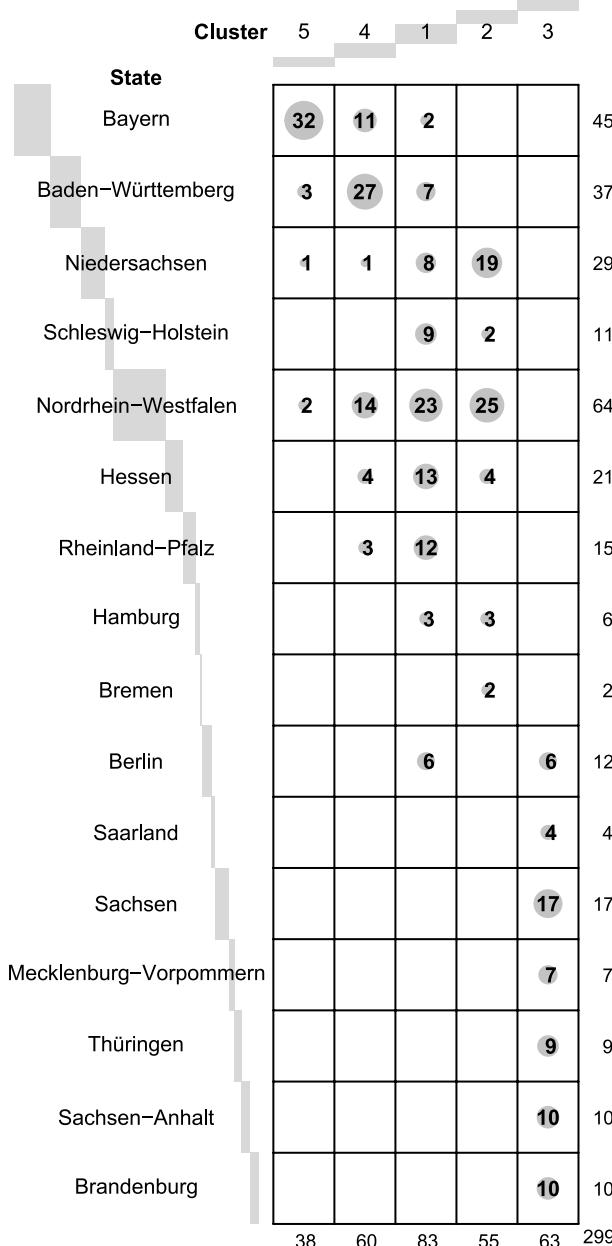
Here we provide only one simple example to demonstrate the basic principle. Each electoral district belongs to one of the 16 German federal states; see Fig. 11.10 for a map. Table 11.3 shows a cross-tabulation of state and cluster membership in a balloon plot (Warnes, 2005). The state information has not been used in the clustering process, so we can treat the table as a usual contingency table of two nominal variables. Figure 11.11 shows a mosaicplot (Hartigan and Kleiner, 1984; Friendly, 2000; Meyer et al., 2005) for the table, see III.12 (Meyer et al.) and III.13 (Hoffmann) for more discussion of mosaicplots. Cells with unusually high or low counts (under the null hypothesis of independence of columns and rows) are shaded. The most striking pattern in the contingency table is that Saarland (located in the southwest of Germany, at the French border) voted in a similar way to the eastern states. This is most likely due to the fact that Oscar Lafontaine, one of the two leaders of LINKE, is a former prime minister of Saarland. Another pattern that can be easily spotted from the mosaicplot is that Nordrhein-Westfalen is not only the largest state, but it also has districts that exhibit very diverse voting behavior and thus spreads over all four western clusters.

Self-Organizing Maps

11.3.7

Self-organizing maps (SOMs, Kohonen, 1989) impose a (typically two-dimensional rectangular) grid on cluster centroids. The grid is specified before the data are clustered, and centroids that are neighbors on the grid are forced to stay “close” to each other during the complete procedure. This has the advantage that the resulting neighborhood graph can always be easily projected onto two dimensions by simply using

Table 11.3. Cross-tabulation of German federal states and cluster membership of electoral districts. Columns and rows are sorted by decreasing percentage of votes for CDU/CSU; the area of each gray circle is proportional to the corresponding table entry. The height of each bar in the top and left margins is proportional to the marginal count for the column or row of the table, respectively



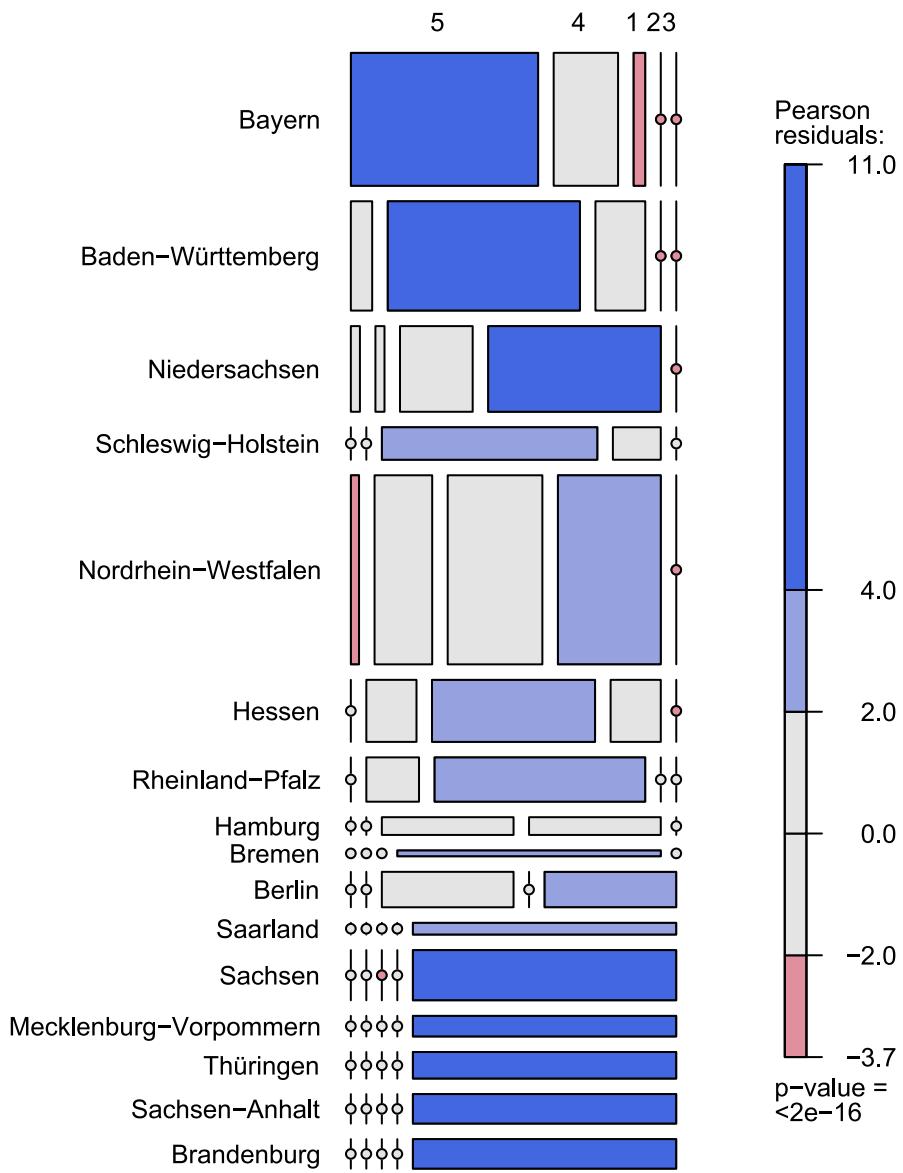


Figure 11.11. [This figure also appears in the color insert.] Mosaic plot for Table 11.3

the location on the grid as coordinates in 2-D. For details on the algorithm, see the reference above.

Figure 11.12 shows the cluster centroids and the connecting grid projected onto the first two principal components of the election data. The cluster centroids are also shown as pie charts in Fig. 11.13. The area of each pie segment corresponds to the

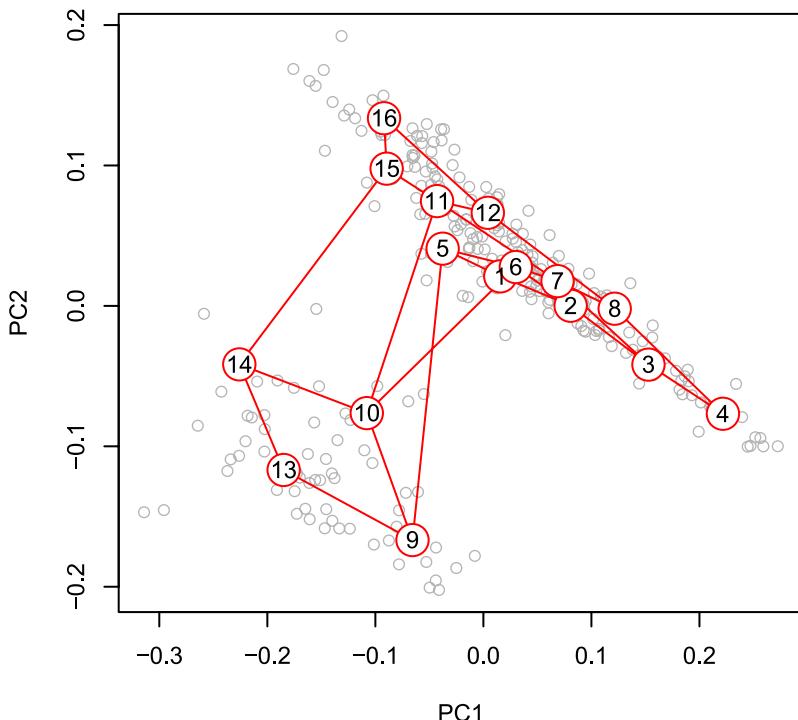


Figure 11.12. 4×4 rectangular SOM grid projected onto the first two principal components of the German election data

relative size of each party in the corresponding segment after rescaling each party separately. Eastern Germany is represented by clusters nine, ten, thirteen and fourteen, while the large black pie segments indicate the strong performance of LINKE in this part of the country.

11.4 Model-Based Clustering

Model-based clustering fits a set of K (usually identical) probabilistic models to the data set. After parameter estimation, each of the K models is interpreted as a cluster, and the likelihood that a data point is observed given one of the models again induces a partition in the input space. Consider a finite mixture model with K components of the following form

$$h(\mathbf{y}|\mathbf{x}, w) = \sum_{k=1}^K \pi_k f(\mathbf{y}|\mathbf{x}, \theta_k), \quad (11.15)$$

$$\pi_k \geq 0, \quad \sum_{k=1}^K \pi_k = 1, \quad (11.16)$$

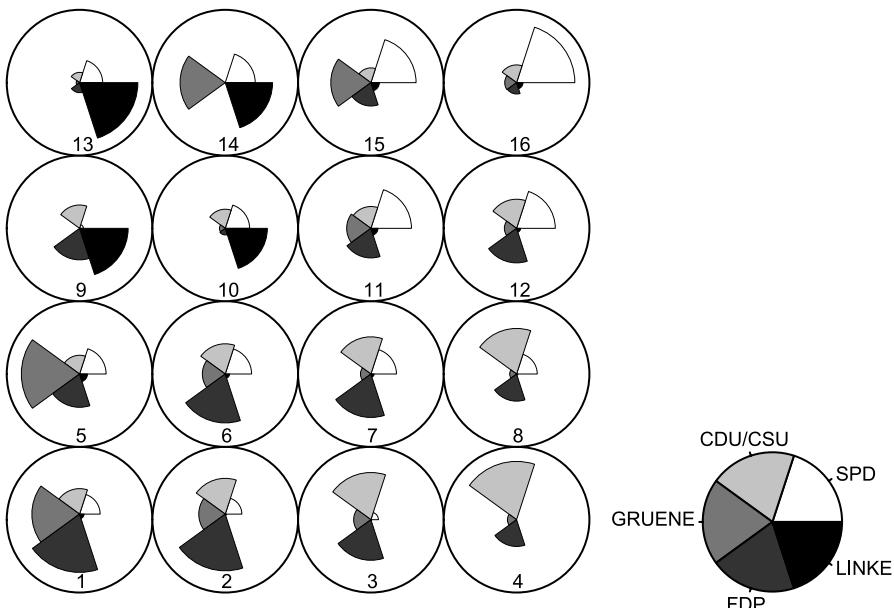


Figure 11.13. Centroids of the SOM. The size of each pie segment corresponds to a cluster centroid after rescaling each variable separately. The parties have large segments in clusters where they performed above average

where \mathbf{y} is a (possibly multivariate) dependent variable with a conditional density h , \mathbf{x} is a vector of independent variables, π_k is the prior probability of component k , and θ_k is the component-specific parameter vector for the density function f .

If f is a normal density with component-specific mean $\mu_k = \beta'_k \mathbf{x}$ and variance σ_k^2 (covariance matrix Σ_k for multivariate \mathbf{y}), we have $\theta_k = (\beta'_k, \sigma_k^2)'$ and Eq. 11.15 describes a mixture of standard linear regression models, also called latent class regression. A special case is $\mathbf{x} \equiv 1$, which gives a mixture of Gaussians without a regression part, and this is also called model-based clustering. If f is a member of the exponential family, we get a mixture of generalized linear models (Wedel and DeSarbo, 1995).

For the German election data, we use five-dimensional Gaussians as mixture components such that the parameters θ_k are the mean μ_k and covariance matrix Σ_k of each component. Using the Bayesian information criterion (BIC) to select the number of components K (Fraley and Raftery, 2002) results in $K = 5$ segments with prior probabilities $\pi_k \in \{0.25, 0.22, 0.31, 0.08, 0.13\}$. The largest component (number three) has a mean vector μ_3 of

SPD	CDU/CSU	GRUENE	FDP	LINKE
0.382	0.307	0.110	0.100	0.055

The covariance matrix Σ_3

	SPD	CDU/CSU	GRUENE	FDP	LINKE
SPD	0.0050	-2.5×10^{-3}	-1.5×10^{-3}	-0.00150	0.00050
CDU/CSU	-0.0025	2.3×10^{-3}	2.3×10^{-5}	0.00083	-0.00048
GRUENE	-0.0015	2.3×10^{-5}	1.4×10^{-3}	0.00032	-0.00006
FDP	-0.0015	8.3×10^{-4}	3.2×10^{-4}	0.00058	-0.00019
LINKE	0.0005	-4.8×10^{-4}	-5.9×10^{-5}	-0.00019	0.00016

is proportional to the correlation matrix

	SPD	CDU/CSU	GRUENE	FDP	LINKE
SPD	1.00	-0.749	-0.579	-0.88	0.57
CDU/CSU	-0.75	1.000	0.013	0.73	-0.81
GRUENE	-0.58	0.013	1.000	0.37	-0.13
FDP	-0.88	0.726	0.367	1.00	-0.63
LINKE	0.57	-0.812	-0.130	-0.63	1.00

Numerical results for the remaining four components are omitted for brevity and simplicity. To visualize the results, we again use a principal component projection of the data into two dimensions. Principal component analysis returns a linear projection. Let \mathcal{A} be the corresponding projection matrix. A linear transformation of a multivariate Gaussian distribution is another multivariate Gaussian, with parameters

$$\tilde{\mu}_k = \mathcal{A}\mu_k \quad (11.17)$$

$$\tilde{\Sigma}_k = \mathcal{A}\Sigma_k\mathcal{A}^T \quad (11.18)$$

Figure 11.14 shows the five cluster components projected onto the first two principal components. The inner solid ellipse for each cluster gives the 50 % confidence region of the corresponding Gaussian distribution, the dashed outer ellipse the 95 % confidence region. At first sight, the main difference between this segmentation and the PAM result from Sect. 11.3 is that the eastern states have been split into two clusters (instead of one), and the western states into three (instead of four). There also seems to be huge overlap within these two groups of clusters, which may make one wonder why the BIC favors a five-cluster solution over a 2-cluster solution (east vs. west). Obviously a more thorough investigation of the result is necessary.

Fitting a mixture model to the data does not directly partition the data into disjoint groups, but it does provide probability values that indicate how likely it is that a given point belongs to a segment – the so-called posterior probability that observation (x, y) belongs to class j :

$$P(j|x, y) = \frac{\pi_j f_j(y|x, \theta_j)}{\sum_k \pi_k f_k(y|x, \theta_k)}. \quad (11.19)$$

Histograms or rootograms of the posterior class probabilities can be used to visually assess the cluster structure (Tantrum et al., 2003). Rootograms are very similar to

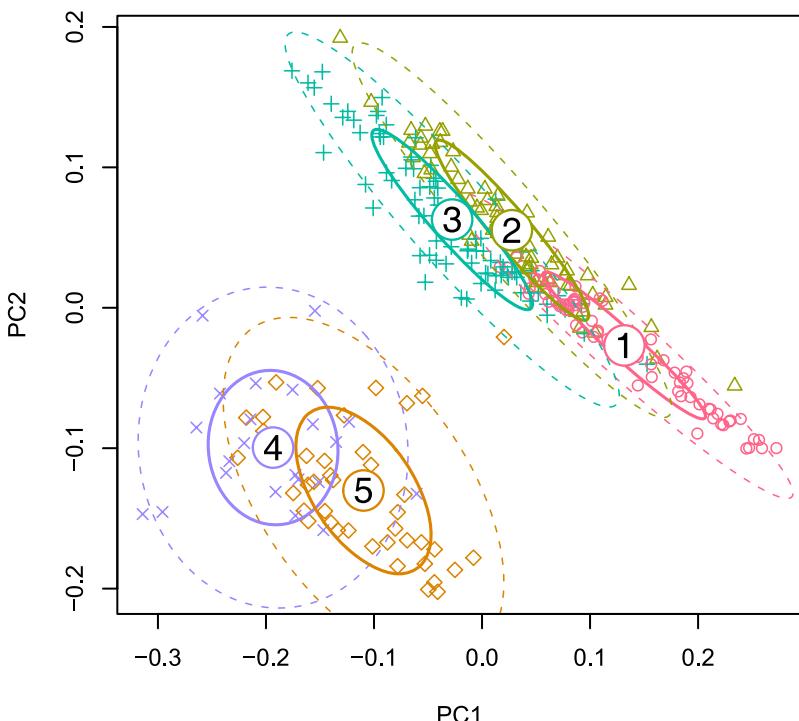


Figure 11.14. A five-cluster finite mixture model for the German election data projected on the first two principal components

histograms; the only difference between them is that the bar height corresponds to the square root of the counts rather than the counts themselves, so low counts become more visible and the emphasis on peaks is reduced.

Usually many of the observations in each component have posteriors close to zero, resulting in a high count for the corresponding bin in the rootogram, which obscures the information in the other bins. To avoid this problem, all probabilities with a posterior below a particular threshold are ignored (we use 10^{-4}). A peak at probability 1 indicates that a mixture component is well separated from the other components, while no peak at 1 and/or a significant mass in the middle of the unit interval indicates overlap with other components.

Figure 11.15 shows that in our example the components are well separated, so the apparent overlap in Fig. 11.14 is probably an artefact of the projection. In addition, we highlight the posteriors of cluster three in the rootograms, which visualizes overlap with other clusters without using any projection. The largest overlap is with clusters one and two, but there is also some overlap (one district each) with four and five. Of course, one would now proceed to highlight posteriors corresponding to the remaining four clusters.

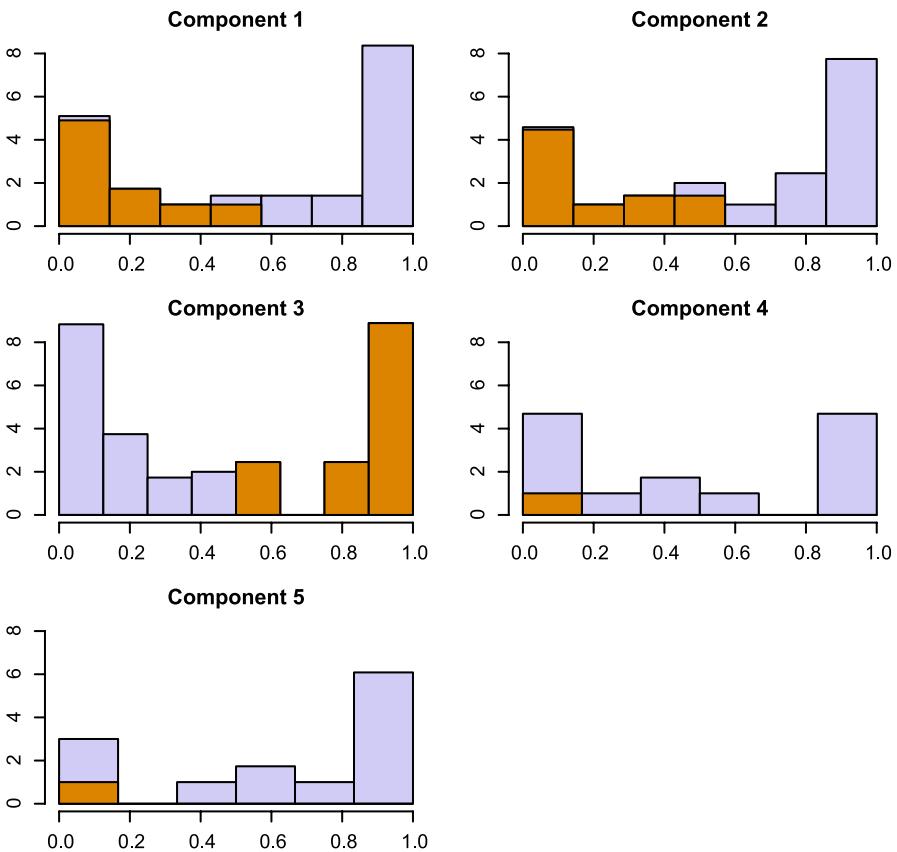


Figure 11.15. Rootograms of the posterior cluster probabilities. Cluster three is highlighted by selecting those cases with posteriors that are larger than 0.5

Projecting the five components onto two combinations of the original variables, as done in Fig. 11.16, provides more detail. In these plots, the original data have been omitted to draw more attention to the fitted model. Both plots have a similar pattern, which is more heavily emphasized in the upper panel (FDP vs. CDU/CSU). These two parties were potential coalition partners before the election and have a positive correlation in four of the five clusters. The only cluster with a negative correlation is cluster number three (mainly Bavaria and Baden-Württemberg), the stronghold of CDU/CSU. In this cluster, every vote for another party seems to be at the expense of CDU/CSU, such that they have a negative correlation with all other parties, including the FDP.

The same is true for the SPD in cluster three, the only cluster where they have a strong negative correlation with their potential coalition partner, GRUENE. From the two plots we also see one of the main differences between the two eastern clusters: in cluster four CDU/CSU and FDP have a strong positive correlation, while in clus-

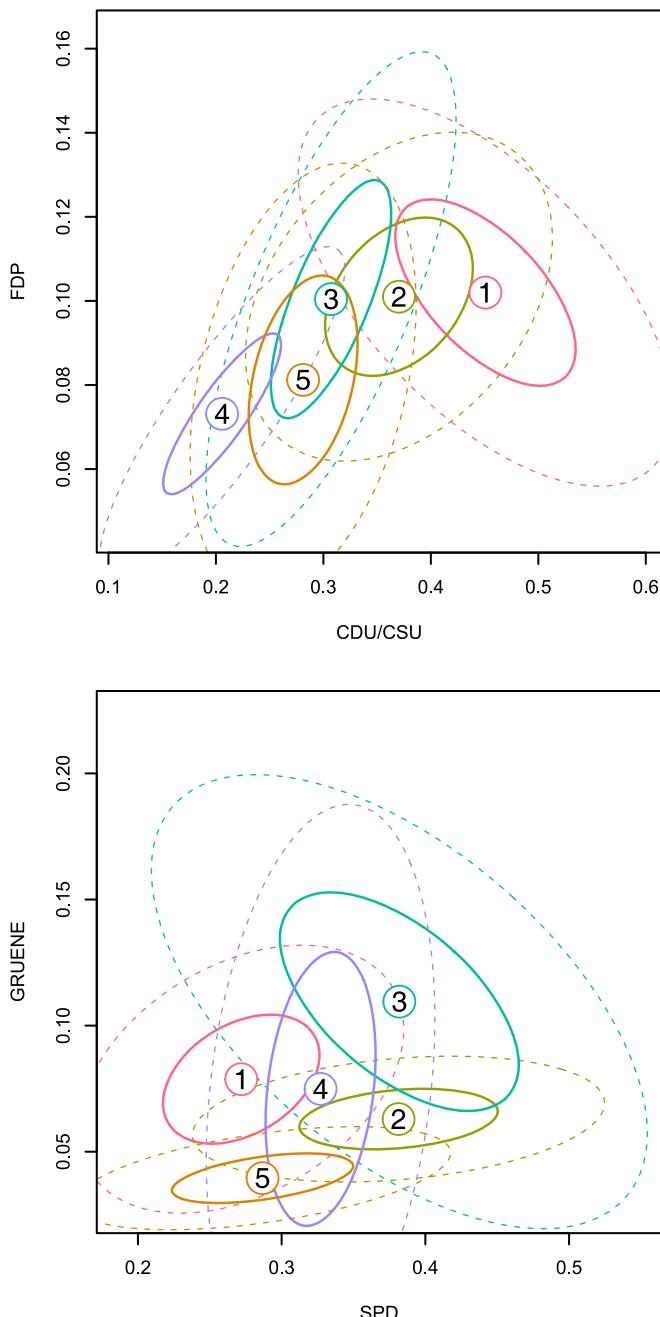


Figure 11.16. Marginal confidence ellipses for the variables FDP vs. CDU/CSU (*top*) and GRUENE vs. SPD (*bottom*)

ter five the correlation is much weaker. SPD and GRUENE are weakly correlated in both clusters; the difference here is mainly the variance: GRUENE has a much higher variance in cluster four than in cluster five.

While all of these patterns can of course also be inferred from numerical printouts of the component parameters, they are made much more obvious by using graphical techniques. Complete scatterplot matrices of all pairwise combinations can be used to find interesting variable combinations. Figures 11.14 and 11.16 could be augmented by a neighborhood graph similar to those used in Sect. 11.3. A natural choice to measure the proximity of the mixture components is the Kullback-Leibler-divergence of the component distributions, see Leisch (2004) for details.

11.5

Summary

The natural visualization method for hierarchical clustering is the cluster dendrogram, because it directly reflects the construction principle of the underlying algorithm. Similarly, the visualization of SOMS is tightly bundled with the algorithm, and feature maps on three-dimensional grids are a de facto standard.

Two larger groups of plots are available for partitioning and model-based clustering: the first group are diagnostic plots like silhouettes and posterior rootograms, which try to visualize the quality of the clustering. The second group of plots simply treats cluster membership as a categorical variable, and uses standard techniques like glyphs, colors or trellis displays to highlight cluster membership in visualizations of the original data (or projections thereof). Virtually all of the methods of plotting a group of variables against a single categorical variable proposed in this handbook can be used for this purpose. The examples shown in this chapter are popular choices, but are primarily intended as a starting point and source of inspiration.

References

- Becker, R., Cleveland, W. and Shyu, M.-J. (1996). The visual design and control of trellis display, *Journal of Computational and Graphical Statistics* 5:123–155.
- Everitt, B.S., Landau, S. and Leese, M. (2001). *Cluster Analysis*, 4th edn, Arnold, London, UK.
- Fraley, C. and Raftery, A.E. (2002). Model-based clustering, discriminant analysis and density estimation, *Journal of the American Statistical Association* 97:611–631.
- Friendly, M. (2000). *Visualizing Categorical Data*, SAS Press, Cary, NC. ISBN 1-58025-660-0.
- Gordon, A.D. (1999). *Classification*, 2nd edn, Chapman & Hall / CRC, Boca Raton, FL, USA.
- Hartigan, J.A. (1975). *Clustering Algorithms*, Wiley, New York.
- Hartigan, J.A. and Kleiner, B. (1984). A mosaic of television ratings, *The American Statistician* 38(1):32–35.
- Hartigan, J.A. and Wong, M.A. (1979). Algorithm AS136: A k -means clustering algorithm, *Applied Statistics* 28(1):100–108.

- Hennig, C. (2004). Asymmetric linear dimension reduction for classification, *Journal of Computational and Graphical Statistics* 13(4):1–17.
- Kaufman, L. and Rousseeuw, P.J. (1990). *Finding Groups in Data*, Wiley, New York.
- Kohonen, T. (1989). *Self-organization and Associative Memory*, 3rd edn, Springer, New York.
- Lance, G.N. and Williams, W.T. (1967). A general theory of classification sorting strategies I. hierarchical systems, *Computer Journal* 9:373–380.
- Leisch, F. (2004). Exploring the structure of mixture model components, in J. Antoch (ed), *Compstat 2004 – Proceedings in Computational Statistics*, Physica Verlag, Heidelberg, pp. 1405–1412. ISBN 3-7908-1554-3.
- Leisch, F. (2006). A toolbox for k-centroids cluster analysis, *Computational Statistics and Data Analysis* 51(2):526–544.
- Mächler, M., Rousseeuw, P., Struyf, A. and Hubert, M. (2005). *cluster: Cluster Analysis*. R package version 1.10.0.
- MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations., in Cam, L.M.L. and Neyman, J. (eds), *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, University of California Press, Berkeley, CA, pp. 281–297.
- Martinetz, T. and Schulten, K. (1994). Topology representing networks, *Neural Networks* 7(3):507–522.
- Meyer, D., Zeileis, A. and Hornik, K. (2005). *vcd: Visualizing Categorical Data*. R package version 0.9-5.
- Milligan, G.W. and Cooper, M.C. (1985). An examination of procedures for determining the number of clusters in a data set, *Psychometrika* 50(2):159–179.
- Murrell, P. (2005). *R Graphics*, Chapman & Hall / CRC, Boca Raton, FL.
- Pison, G., Struyf, A. and Rousseeuw, P.J. (1999). Displaying a clustering with CLUS-PLOT, *Computational Statistics and Data Analysis* 30:381–392.
- R Development Core Team (2007). *R: A language and environment for statistical computing*, R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0. <http://www.R-project.org>
- Rousseeuw, P.J. (1987). Silhouettes: A graphical aid to the interpretation and validation of cluster analysis, *Journal of Computational and Applied Mathematics* 20:53–65.
- Rousseeuw, P.J., Ruts, I. and Tukey, J.W. (1999). The bagplot: A bivariate boxplot, *The American Statistician* 53(4):382–387.
- Tantrum, J., Murua, A. and Stuetzle, W. (2003). Assessment and pruning of hierarchical model based clustering, *Proceedings of the ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ACM, New York, pp. 197–205. ISBN:1-58113-737-0.
- Warnes, G.R. (2005). *gpplots: Various R programming tools for plotting data*. R package version 2.0.8.
- Wedel, M. and DeSarbo, W.S. (1995). A mixture likelihood approach for generalized linear models, *Journal of Classification* 12:21–55.

Visualizing Contingency Tables III.12

David Meyer, Achim Zeileis, Kurt Hornik

12.1	<i>Introduction</i>	590
12.2	<i>Two-Way Tables</i>	591
	Mosaic Displays	592
	Sieve Plots.....	595
	Association Plots	596
	Summary	598
12.3	<i>Using Colors for Residual-Based Shadings</i>	598
	A Note on Colors and Color Palettes.....	598
	Highlighting and Color-Based Shadings.....	601
	Visualizing Test Statistics	603
	Summary	605
12.4	<i>Selected Methods for Multiway Tables</i>	606
	Exploratory Visualization Techniques	607
	Model-Based Displays for Conditional Independence Models	608
	A Four-Way Example	611
	Summary	614
12.5	<i>Conclusion</i>	614

Introduction

Categorical data analysis is typically based on two- or higher dimensional contingency tables, cross-tabulating the co-occurrences of levels of nominal and/or ordinal data. In order to explain these, statisticians typically look for (conditional) independence structures using common methods such as independence tests and log-linear models. One idea behind the use of visualization techniques is to use the human visual system to detect structures in the data that may not be obvious from solely numeric output (e.g., test statistics). Whether the task is purely exploratory or model-based, techniques such as mosaic, sieve, and association plots offer good support for visualization. Mosaic and sieve plots in particular have been extended over the last two decades, and implementations exist in many statistical environments.

All three graphical methods visualize aspects of (possibly high-dimensional) contingency tables. A *mosaicplot* (Hartigan and Kleiner, 1984) is basically an area-proportional visualization of (typically observed) frequencies, consisting of tiles (corresponding to the cells) created by vertically and horizontally splitting a rectangle recursively. Thus, the area of each tile is proportional to the corresponding cell entry *given* the dimensions of previous splits. *Sieve plots* (Riedwyl and Schüpbach, 1994) are similar to mosaicplots, but the area of each tile is proportional to the *expected* cell entry, and each tile is filled with a number of rectangles corresponding to the observed value. An *association plot* (Cohen, 1980) visualizes the standardized deviations of observed frequencies from those expected under a certain independence hypothesis. Each cell is represented by a rectangle that has a (signed) height that is proportional to the residual and width that is proportional to the square root of the expected counts, so that the area of the box is proportional to the difference in observed and expected frequencies.

Over the years, extensions to these techniques have mainly focused on the following aspects:

- Varying the shapes of mosaicplots (as well as bar plots) to yield, e.g., double-decker plots (Hofmann, 2001) or spine plots (Riedwyl and Schüpbach, 1994).
- Using residual-based shadings to visualize log-linear models (Friendly 1994, 2000) and the significance of statistical tests (Meyer et al., 2003).
- Using pairs plots and trellis-like layouts for marginal, conditional and partial views (Friendly, 1999).
- Adding direct user interaction, allowing quick exploration and modification of the visualized models (Unwin et al., 1996; Theus, 2003).
- Providing a modular and flexible implementation to easily allow user extensions (Meyer et al., 2006).

Current implementations of mosaic displays can be found for, e.g., SAS, (SAS Institute Inc., 2005), ViSta (Young, 1996), MANET (Unwin et al., 1996), Mondrian (Theus, 2003), R, (R Development Core Team, 2006), and S-PLUS (Insightful Inc., 2005). Implementations of association and sieve plots can only be found in R and SAS (in the latter, these plots are available for two-way tables only). Table 12.1 gives an overview of

Table 12.1. Comparison of current software environments

	SAS	S-PLUS	R	ViSta	MANET	Mondrian
Basic functionality	×	×	×	×	×	×
Shape			×		×	×
Residual-based shadings	×		×	×	(×)	(×)
Conditional views	×		×		×	×
Interaction				×	×	×
Extensible design			×			

the functionality available in these systems. The figures in this chapter have all been produced using the R system, using the extension packages vcd (Meyer et al., 2006) and scatterplot3d (Ligges and Mächler, 2003) (Fig. 12.2 only), all freely available from the Comprehensive R Archive Network (<http://CRAN.R-project.org/>). The R code used for the figures is available from <http://statmath.wu-wien.ac.at/projects/vcd/>.

This chapter will provide an overview of the state of the art for mosaic and association plots, from both exploratory visualization and model-based analysis perspectives. Exploratory techniques will include specialized displays for the bivariate case, as well as pairs plot-like displays for high-dimensional tables. As for the model-based tools, particular emphasis will be given to methods suitable for the visualization of conditional independence tests (including permutation tests), as well as for the visualization of particular GLMs (such as log-linear models). In Sect. 12.2, we start with the simple bivariate case. Sect. 12.3 explains how the use of color in residual-based shadings can support data exploration, and even promotes the methods to diagnostic and model-based tools by visualizing test statistics and residuals of independence models. In Sect. 12.4, we show how the basically bivariate methods straightforwardly extend to the multivariate case by using “flat” representations of the multiway tables. In this section, we also introduce specialized displays for conditional independence structures. The techniques are illustrated using three- and four-way tables. Section 12.5 concludes the chapter.

Two-Way Tables

12.2

Throughout this section, our examples will be based on the hospital data (Wing, 1962) given in Table 12.2.

The table relates the length of stay (in years) of 132 long-term schizophrenic patients in two London mental hospitals with the frequency of visits (from relatives or friends). The length of stay (LOS) has been categorized into 2–9 years, 10–19 years, and more than 19 years. There are also three categories for the visit frequency: regular (including patients who were allowed to go home), less than monthly, and never. Wing (1962) concludes from this data that the longer the patients stayed in hospital, the less frequently they are visited, which can be seen from the column-standardized table (see Table 12.3).

Table 12.2. The hospital data

Visit frequency	Length of stay (in years)			Σ
	2–9	10–19	20+	
Regular	43	16	3	62
Less than monthly	6	11	10	27
Never	9	18	16	43
Σ	58	45	29	132

Table 12.3. The hospital data, corrected for the column margin

Visit frequency	Length of stay (in years)		
	2–9	10–19	20+
Regular	0.74	0.36	0.10
Less than monthly	0.10	0.24	0.35
Never	0.16	0.40	0.55
Σ	1.00	1.00	1.00

Table 12.4. The hospital data, corrected for the row margin

Visit frequency	Length of stay (in years)			Σ
	2–9	10–19	20+	
Regular	0.69	0.26	0.05	1.00
Less than monthly	0.22	0.41	0.37	1.00
Never	0.21	0.42	0.37	1.00

In addition, Haberman (1974) notes that this pattern is not significantly different in the “less than monthly” and “never” strata. From the row-standardized table (see Table 12.4), it seems indeed that LOS is homogeneous with respect to these two visit frequency strata.

Although far from optimal, contingency tables are frequently visualized using grouped bar plots (see Fig. 12.1) or even by means of 3-D bar charts (see Fig. 12.2). It seems hard to detect the aforementioned pattern in these, especially in the 3-D plot, where the perspective view tends to distort the true proportions of the bars. In the following, we will introduce three graphical methods that are better suited to contingency tables.

12.2.1 Mosaic Displays

Mosaic displays were introduced by Hartigan and Kleiner (1981, 1984) and extended by, among others, Friendly (1994, 1999, 2000). They visualize the observed values of a contingency table by area-proportional tiles, arranged in a rectangular mosaic. The tiles are obtained by recursively partitioning and splitting a rectangle. In the following, we describe the main concepts of mosaicplots; Chapter III-13 in this book

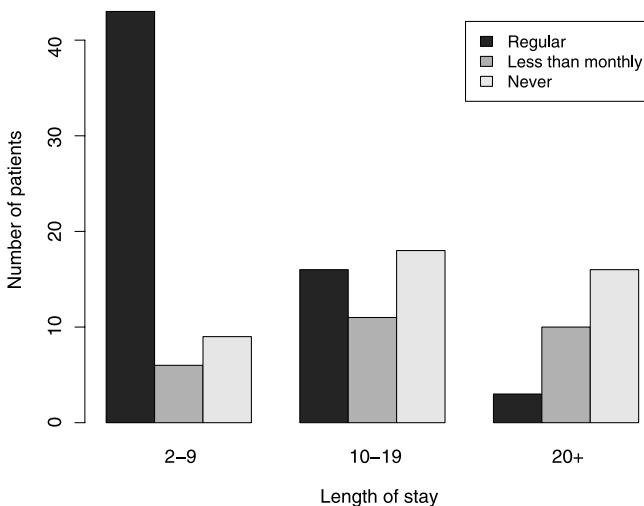


Figure 12.1. Bar plot for the hospital data

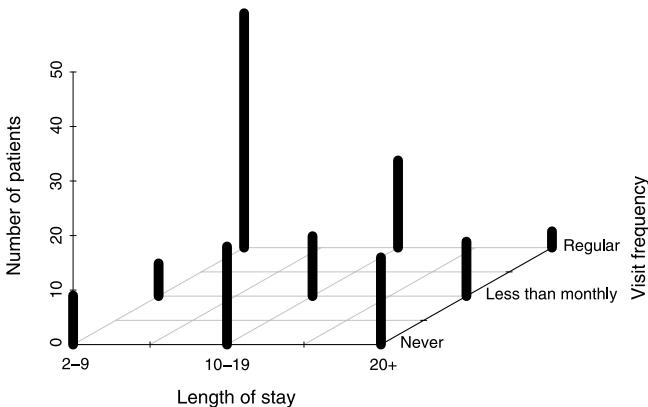


Figure 12.2. 3-D-bar chart for the hospital data

provides more detailed information on this topic. Consider our example of the hospital data from above. Step 1 consists of splitting a square according to the marginals of one of the variables. To be consistent with the textual representation, we choose LOS with vertical splits (see Fig. 12.3). The result is similar to a bar plot, but in this case the width rather than the height is adapted to visualize the counts for each level. Such a plot is also called a *spine plot* (Hummel, 1996). From this plot, we see that the number of patients decreases with the length of stay. Step 2 involves performing further splitting in the other direction (resulting in horizontal splits) for the second variable. This means that each vertical bar is split according to the marginals of the second variable, *given* the first variable (see Fig. 12.4). The resulting plot visualizes the contingency table such that each cell has a size proportional to the corresponding ta-

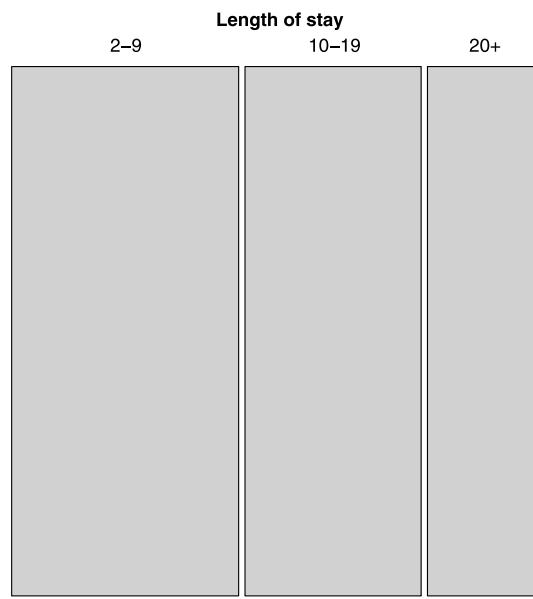


Figure 12.3. Construction of a mosaicplot for a two-way table: step 1

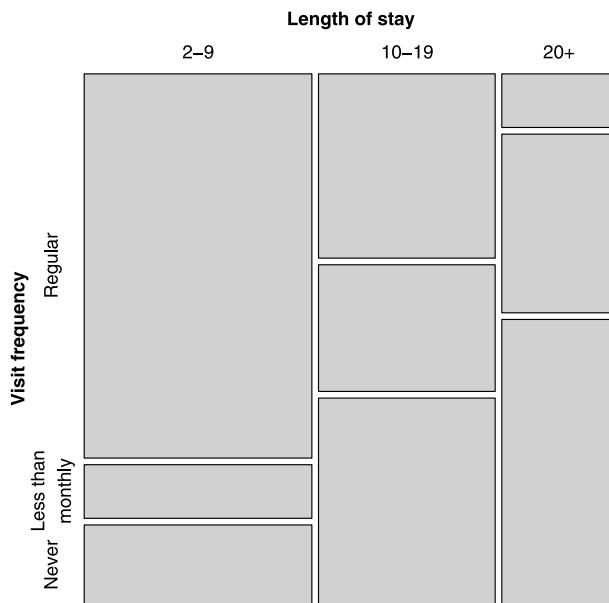


Figure 12.4. Construction of a mosaicplot for a two-way table: step 2

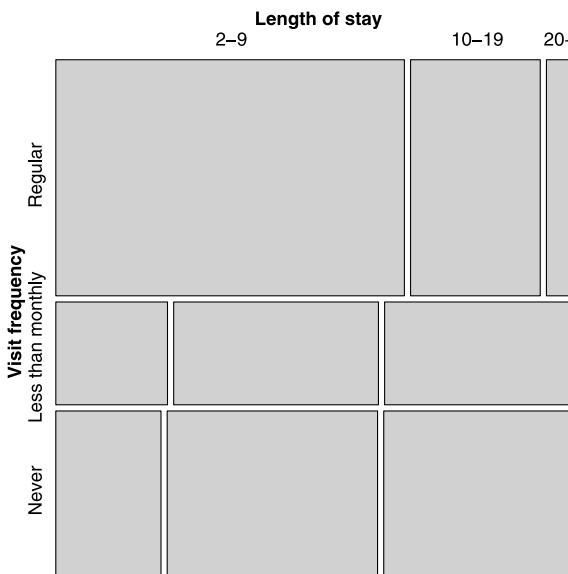


Figure 12.5. Mosaic plot for the hospital data, using “visit frequency” as first splitting variable

ble entry. We can still see the marginal distribution of LOS and additionally the visit frequency *given* the category of LOS. If the two variables were independent, the grid would be regular. Clearly, compared to a length of stay of 10–19 years, more patients get regular visits for stays from 2–9 years, and conversely, fewer patients get regular visits for stays of for more than 19 years. For patients that get no visits, the pattern is reversed.

Since mosaicplots are asymmetric by construction, the choice of the variable order matters, as the first splitting variable dominates the plot. In our example, if we use “visit frequency” as the first splitting variable, the impression obtained is very different compared to that of the previous mosaic (see Fig. 12.5). In this alternative display (see Fig. 12.6), we see the marginal distribution of “visit frequency” in the rows: about half of the patients get visited regularly. This group is dominated by patients staying 2–9 years. It seems apparent that the distribution of LOS is similar for monthly and never visited patients, so these two categories actually represent one homogeneous group (patients visited only casually). Since the first splitting variable dominates the plot, it should be chosen to be the explanatory variable.

Sieve Plots

12.2.2

When we try to explain data, we assume the validity of a certain model for the generating process. In the case of two-way contingency tables, the two most common and well-known hypotheses (Agresti, 2002) are

1. independence of the two variables,
2. homogeneity of one variable among the strata defined by the second.

Table 12.5. The hospital data – expected values

Visit frequency	Length of stay (in years)			Σ
	2–9	10–19	20+	
Regular	27.24	21.14	13.62	62
Less than monthly	11.86	9.20	5.93	27
Never	18.89	14.66	9.45	43
Σ	58.00	45.00	29.00	132

It is easy to compute the *expected* table under either of these hypotheses. To fix notations, in the following we consider a two-way contingency table with I rows and J columns, cell frequencies $\{n_{ij}\}$ for $i = 1, \dots, I$ and $j = 1, \dots, J$, and row and column sums $n_{i+} = \sum_j n_{ij}$ and $n_{+j} = \sum_i n_{ij}$, respectively. For convenience, the number of observations is denoted $n = n_{++}$. Given an underlying distribution with theoretical cell probabilities π_{ij} , the null hypothesis of independence of the two categorical variables can be formulated as

$$H_0 : \pi_{ij} = \pi_{i+} \pi_{+j}. \quad (12.1)$$

Now, the expected cell frequencies in this model are simply $\hat{n}_{ij} = n_{i+} n_{+j} / n$. The expected table for our sample data is given in Table 12.5. It could again be visualized using a mosaicplot, this time applied to the table of expected frequencies. If we cross-tabulate each tile to fill it with a number of squares equal to the corresponding number of *observed* frequencies, we get a *sieve plot* (see Fig. 12.6). This implicitly compares expected and observed values, since the density of the grid will increase with the deviation of the observed from the expected values. This allows the detection of general association patterns (for nominal variables) and of linear association (for ordinal variables), the latter producing tiles of either very high or very low density along one of the diagonals. For our data, the density of the rectangles is marked along the secondary diagonal, indicating a negative association of the two variables. This provides evidence that visit frequency decreases with the length of stay for these patients.

12.2.3 Association Plots

In the last section, we described how to compare observed and expected values of a contingency table using sieve plots. We can do this more straightforwardly by using a plot that directly visualizes the residuals. The most widely used residuals are the Pearson residuals

$$r_{ij} = \frac{n_{ij} - \hat{n}_{ij}}{\sqrt{\hat{n}_{ij}}}. \quad (12.2)$$

which are standardized raw residuals. In an association plot (Cohen, 1980), each cell is represented by a rectangle that has a (signed) height that is proportional to the

		Length of stay		
		2–9	10–19	20+
Visit frequency	Regular	43	16	3
	Less than monthly	6	11	10
	Never	9	18	16

Figure 12.6. Sieve plot for the hospital data

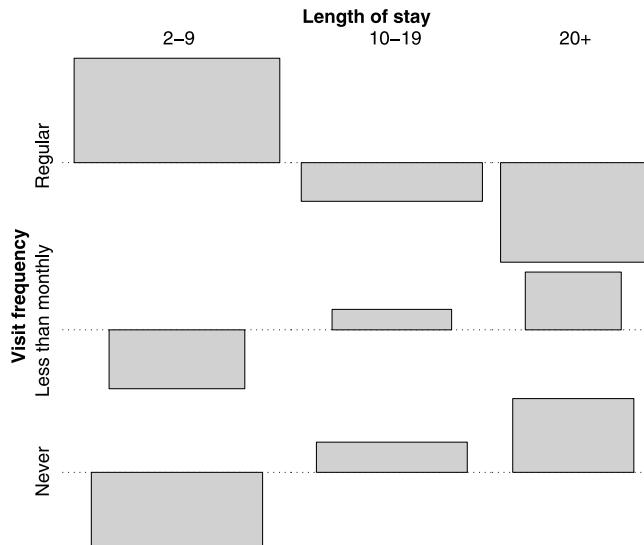


Figure 12.7. Association plot for the hospital data

corresponding Pearson residual r_{ij} and a width that is proportional to the square root of the expected counts $\sqrt{\hat{n}_{ij}}$. Thus, the area is proportional to the raw residuals $n_{ij} - \hat{n}_{ij}$. The sign is visualized by its position relative to the baseline (upward tiles for positive, and downward tiles for negative residuals). Figure 12.7 shows the association plot for the hospital data. Consistent with the corresponding mosaic and sieve plots,

we clearly see that too many (too few) patients that stay from two to nine (more than 19) years get visited regularly than would be expected under the null hypothesis of independence, and that this pattern is reversed for patients visited less than monthly or that are never visited.

12.2.4

Summary

Mosaic plots are a tool for visualizing the *observed* frequencies of a contingency table based on recursive conditional splits. If one variable is explanatory, it should be used first for splitting; the display then shows the conditional distribution of the dependent variable given the explanatory one. Sieve plots basically visualize the table of *expected* frequencies, and in addition the deviations from the observed frequencies by the density of the grid added to each tile. They complement mosaicplots by detecting dependency patterns for ordinal variables. An alternative way of enhancing mosaicplots to display deviations from expected frequencies is to use residual-based shadings (see the next section), which are typically more intelligible than sieve plots, in particular for nominal variables. Association plots directly visualize Pearson and raw residuals, i.e., standardized and nonstandardized deviations of observed from expected frequencies, respectively. These plots should be used if the diagnostics of independence models are of primary interest.

12.3

Using Colors for Residual-Based Shadings

As introduced in the previous section for association plots, the investigation of residuals from a posited independence model is of major interest when analyzing contingency tables. In the following, we will demonstrate how the use of colors can greatly facilitate the detection of interesting patterns. We start with some general remarks on colors and color palettes.

12.3.1

A Note on Colors and Color Palettes¹

The plots introduced in the previous section are basically composed of tiles whose areas represent characteristics derived from the contingency tables – observed and expected frequencies in the case of mosaic and sieve plots, or residuals as visualized by association plots. When using color for these tiles, it is imperative to choose the right color palettes, derived from suitable color spaces. Apart from aesthetic considerations, wrongly chosen colors might seriously affect the analysis. For example,

- Using high-chroma colors for large areas tends to produce after-image effects which can be distracting (Ihaka, 2003).

¹ The printed version of the article is monochrome, but the electronic version uses colors. The colored versions of the plots are available from the Web page <http://statmath.wu-wien.ac.at/projects/vcd/>.

- Lighter colors tend to make areas look larger than darker colors, so using colors with unequal luminance values makes it difficult to compare area sizes (Cleveland and McGill, 1983).
- Color palettes derived from nonuniform color spaces may contain unbalanced colors with respect to their colorfulness or brightness. When tiles are shaded using such a palette, some of them might appear to be more important than others in an uncontrolled way.

Due to the three-dimensional nature of human color perception (Mollon, 1995), it has been common to specify colors using the three primaries red, green, and blue (RGB colors), especially for computer devices. The appearance of the on-screen color is affected by the characteristics of the device used. For example, the intensity I of a primary on a particular device follows the rule $I = L^\gamma$, where L is the value for the primary color and γ is device-dependent (but typically close to 2.2). Therefore, a first caveat is that if colors are to appear identical on different devices, their gamma characteristics must be taken into account.

Choosing colors and color palettes in RGB space can be a bit inconvenient and hence software implementations are often based on hue–saturation–value (HSV) colors (or the comparable hue–luminance–saturation color scheme). Both spaces are rather similar transformations of the RGB space (Brewer, 1999; Poynton, 2000) and are very common implementations of colors in many computer packages (Moretti and Lyons, 2002). Each color in HSV space is represented by three dimensions (H , S , V): the hue H (the dominant wavelength in the spectrum, in $[0, 360]$), the saturation S (the “colorfulness” or “purity,” in $[0, 100]$), and the value V (the “brightness,” the amount of gray, in $[0, 100]$).² These intuitive dimensions make HSV colors easier to specify than RGB colors. However, HSV colors have several disadvantages. Most importantly, HSV colors are not perceptually uniform because the three HSV dimensions map only poorly to the three perceptual dimensions of the human visual system (Brewer, 1999; Ihaka, 2003). One important issue here is that HSV dimensions are confounded, e.g., saturation is not uniform across different hues. As an example, see Fig. 12.8 (left), which shows a qualitative color palette (colors $(H, 100, 100)$) for varying hues H) in the HSV space: although saturation and value are fixed, the fully saturated blue is perceived to be much darker than the fully saturated red or green, making it difficult to judge the size of shaded areas. Furthermore, the flashy fully saturated HSV colors are hard to look at for a long time. For similar reasons, it is equally difficult to derive acceptable diverging palettes from the HSV space, i.e., bipolar scales containing colors ranging between two very distinct colors. The upper part of Fig. 12.9 shows a diverging palette in the HSV space with colors ranging from a saturated red $(0, 100, 100)$ to a neutral white $(H, 0, 100)$ to a saturated blue $(240, 100, 100)$. Although the palette should be balanced with respect to colorfulness and brightness, the red colors are perceived to be more intense and flashy than the corresponding blue colors.

² In many implementations, all three dimensions are scaled to the unit interval instead of the coordinates used here.

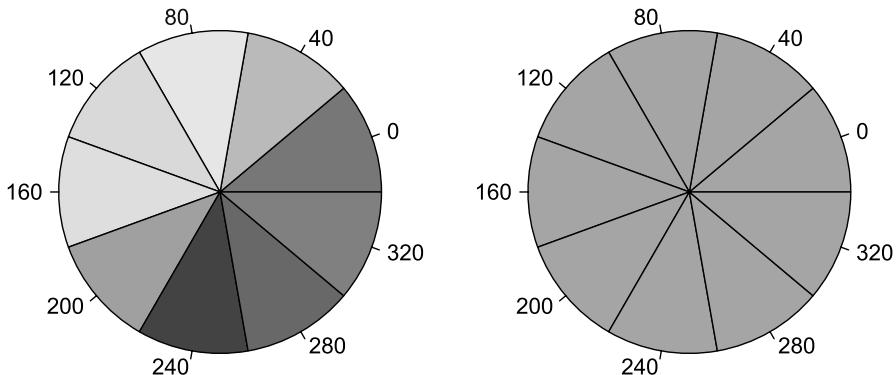
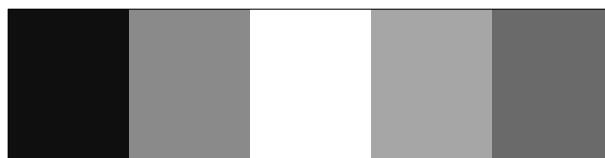


Figure 12.8. [This figure also appears in the color insert.] Qualitative color palettes for the HSV (left) and HCL (right) spaces. The HSV colors are $(H, 100, 100)$ and the HCL colors $(H, 50, 70)$ for the same hues H . Note that in a monochrome version of this paper, all pies in the right wheel will be shaded with the same gray, i.e., they will appear virtually identical

$(240, 100, 100)$ $(240, 50, 100)$ $(H, 0, 100)$ $(0, 50, 100)$ $(0, 100, 100)$



$(260, 100, 50)$ $(260, 50, 70)$ $(H, 0, 90)$ $(0, 50, 70)$ $(0, 100, 50)$

Figure 12.9. [This figure also appears in the color insert.] Diverging color palettes for the HSV space (upper part) and the HCL space (lower part), ranging from blue to a neutral color to red. The triples indicate the settings for the three dimensions: (hue, saturation, value) in the upper part, and (hue, chroma, luminance) in the lower part. In a monochrome version of the paper, the right- and left-hand sides of the HCL color palette will appear to be identical, unlike the HSV color palette

The use of colors that are more “in harmony” goes back to Munsell (1905), who introduced a notation for balanced colors. Based on those, tools producing better palettes for specific tasks have been developed (Harrouer and Brewer, 2003). Other perceptually based color spaces, especially suited for computer displays, are the CIELAB and CIELUV spaces (Commission Internationale de l’Éclairage, 2004) from which qualitative palettes for statistical graphics have been derived (Ihaka, 2003). A transformation of the CIELUV space leads to the HCL (hue–chroma–luminance) space. These colors are again specified by triplets (H, C, L) : chroma C loosely corre-

sponds to colorfulness and luminance L to brightness, but in contrast to HSV colors, chroma is an *absolute* measure that is valid for all hues, and luminance can be varied independent of the other two dimensions. Qualitative color palettes can easily be obtained by holding chroma and luminance constant, and using varying hues. HCL colors with fixed luminance are always balanced towards the same gray and thus do not have the problem of varying saturation like HSV colors (see Fig. 12.8, right). Similarly, diverging HCL color palettes can be derived (Zeileis et al., 2007) by interpolating again between a neutral color such as $(H, 0, 90)$ and two colors with full chroma such as blue $(260, 100, 50)$ and red $(0, 100, 50)$. The resulting palette is shown in the lower part of Fig. 12.9. In contrast to the HSV counterpart, matching colors (light red/blue and dark red/blue) are balanced to the same gray, and thus receive the same perceptual “weight.” Note that chroma and luminance are varied simultaneously, i.e., the full chroma colors are also darker than the neutral color. Of course, it would also be possible to choose a darker neutral color (or lighter full chroma colors); however, by varying both chroma and luminance better contrasts can be achieved.

Highlighting and Color-Based Shadings

12.3.2

The mosaicplots introduced in Sect. 12.2 are composed of empty tiles. It seems intuitive to use filled tiles to highlight information of interest. Consider again the hospital example: in Fig. 12.10, we mark tiles for patients never or seldom visited in order to visualize their proportions in the LOS strata. For optical clarity, we set the spacing

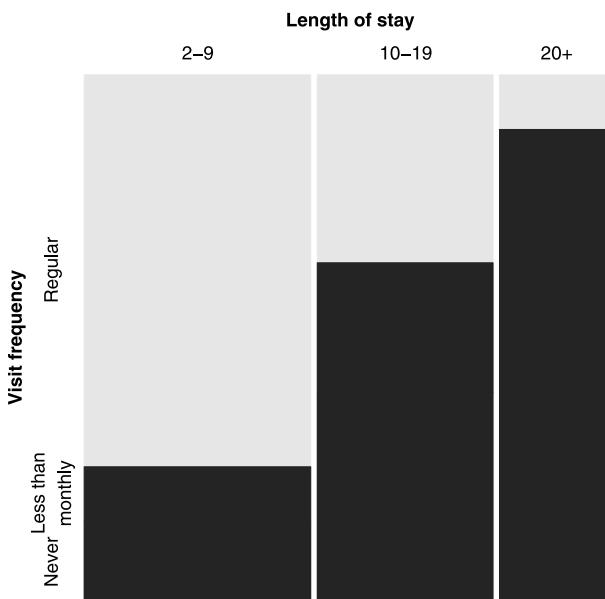


Figure 12.10. Spine plot with highlighting for the hospital data

between the visit frequency tiles to 0. Clearly, the proportion of these patients increases with LOS. In fact, such a “stacked” plot can also be interpreted as a spine plot with highlighting (Hummel, 1996), which is particularly useful when analyzing relationships among categorical data with a binary dependent variable. More variations on this theme, such as doubledecker plots, are treated in Chapter III-13.

Using colors, even more complementary information can be visualized, either by adding additional information, or by redundantly coding information already visualized by the “raw” plot to support our perceptual system. First, we consider the sieve plots. The density of the grid in the raw version implicitly gives us an idea of the sizes of the residuals, but since the plot does not include the density corresponding to zero residuals (the null model) for comparison, we cannot easily assess whether there are more or fewer counts in a cell than expected under the null hypothesis. Using color, we can add the sign information; for example, we can use blue for positive, red for negative, and gray for zero residuals.

The simplest mosaicplots are monochrome displays. Friendly (1994) introduced a residual-based shading of the tiles to additionally visualize the residuals from a given independence model fitted to the table. The idea is to use a color coding for the mosaic tiles that visualizes the sign and absolute size of each residual r_{ij} . Cells corresponding to small residuals ($|r_{ij}| < 2$) have no color. Cells with medium-sized residuals ($2 \leq |r_{ij}| < 4$) are shaded light blue and light red for positive and negative residuals, respectively. Cells with large residuals ($|r_{ij}| \geq 4$) are shaded with fully saturated blue and red, respectively. The heuristic for choosing the cut-offs 2 and 4 is that the Pearson residuals are asymptotically standard normal, which implies that the highlighted cells are those with residuals that are *individually* significant at approximately the

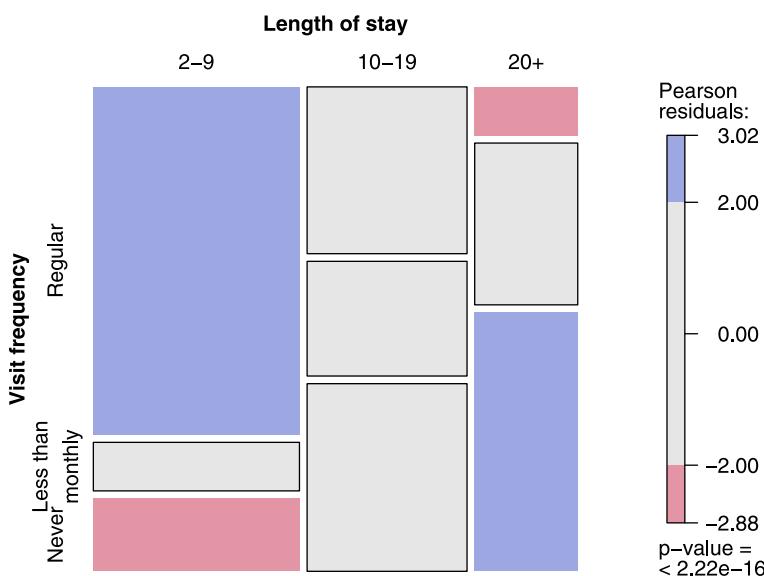


Figure 12.11. Mosaic display with Friendly-like color coding of the residuals

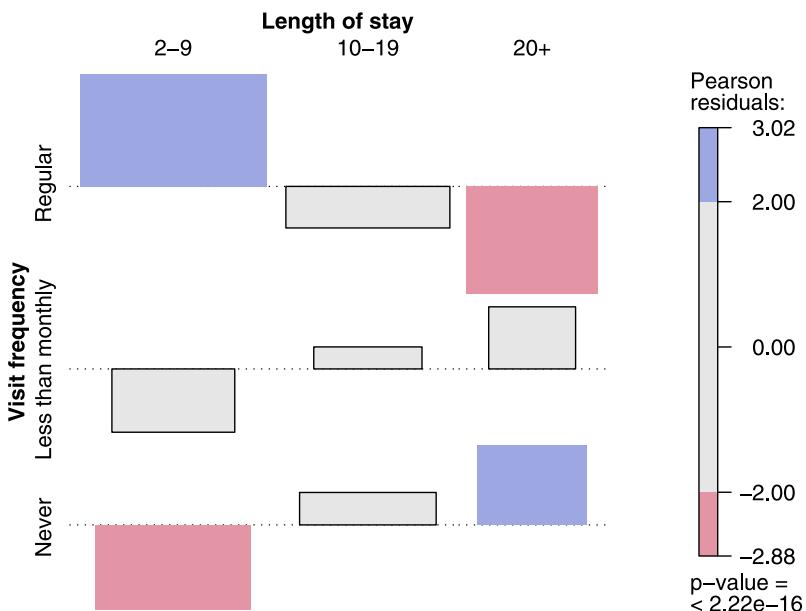


Figure 12.12. Association plot with Friendly-like color coding of the residuals

$\alpha = 0.05$ and $\alpha = 0.0001$ levels. However, the main purpose of this shading is not to visualize the significance but the *pattern* of deviation from independence (Friendly, 2000). In addition to the shading of the rectangles themselves, the Friendly shading also encompasses a choice of line type and line color for the rectangle borders with similar ideas to those described above.

In Fig. 12.11, we again show the mosaic for the hospital data, this time using a Friendly-like color shading (with HCL instead of HSV colors, and no line type coding). Clearly, the asymmetry for regular and never visited patients as well as the pattern inversion for lengths of stay of 2–9 and more than 19 years are emphasized using the color shading.

For association plots, residual-based shadings are redundant since all relevant information is already contained in the plot by construction. Nevertheless, the use of one of the shadings discussed above will support the analysis process and is therefore recommended. For example, applying the Friendly shading in Fig. 12.12 on the one hand facilitates discrimination between positive and negative residuals, and on the other hand, more importantly, makes it easier to compare tile sizes. Thus, the shading supports the detection of the pattern.

Visualizing Test Statistics

12.3.3

Figures 12.11 and 12.12 include the (same) p value for the χ^2 test of independence, which is frequently used to assess the significance of the hypothesis of independence

(or homogeneity for stratified data) in two-way tables. The test statistic is just the sum of the squared Pearson residuals

$$X^2 = \sum_{i,j} r_{ij}^2, \quad (12.3)$$

which is known to have a limiting χ^2 distribution with $(I - 1)(J - 1)$ degrees of freedom under the null hypothesis. An important reason for using the unconditional limiting distribution for the X^2 statistic from (12.3) was the closed form result for the distribution. With the recent improvements in computing power, conditional inference (or permutation tests) – carried out either by simulation or by computation of the (asymptotic) permutation distribution – have been receiving increasing attention (Ernst, 2004; Pesarin, 2001; Strasser and Weber, 1999).

The use of a permutation test is a particularly intuitive way of testing the independence hypothesis from (12.1), due to the permutation invariance (given row and column sums) of this problem. Consequently, all results in this paper are based on conditional inference performed by simulating the permutation distribution of test statistics of type $\lambda([r_{ij}])$.

Since the HCL space is three-dimensional and we have only used two ‘degrees of freedom’ so far to code information (hue for the sign and a linear combination of chroma and luminance for residual size), we can add a third piece of information to the plot. For example, we can visualize the significance of some specified test statistic (e.g., the χ^2 test statistic) using less colorful (“uninteresting”) colors for non-significant results. These can again be derived using the same procedure described in Sect. 12.3.1 but using a smaller amount of color, i.e., a smaller maximal chroma (e.g., 20 instead of 100).

The heuristic for choosing the cut-off points in the Friendly shading may lead to wrong conclusions: especially in large tables, the test of independence may not be significant, even though some of the residuals are “large.” On the other hand, the test might be significant even though the residuals are “small.” In fact, the cut-off points are really data-dependent. Consider the case of the arthritis data (Koch and Edwards, 1988), resulting from a double-blind clinical trial investigating a new treatment for rheumatoid arthritis, stratified by gender (see Table 12.6 for the female patients). Figure 12.13 visualizes the results for the female patients, again using a mosaic display. Clearly, the hypothesis of independence is rejected by the χ^2 test, even at a 1% level ($p = 0.0032$), but since all residuals are in the interval $[-1.7173, 1.8696]$ the tiles remain uncolored. One solution to this issue is to use a different test statistic, for exam-

Table 12.6. The arthritis data (female patients)

Treatment	Improvement			Σ
	None	Some	Marked	
Placebo	19	7	6	32
Treatment	6	5	16	27
Σ	25	12	22	59

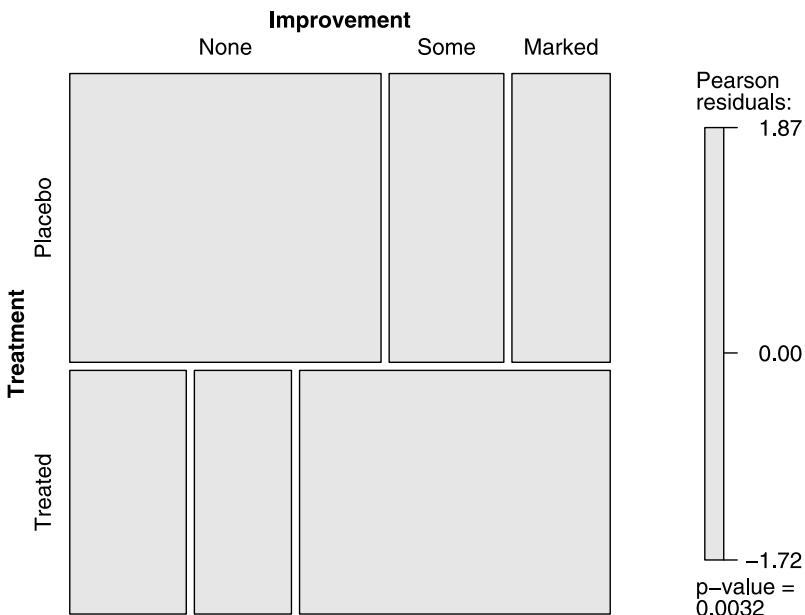


Figure 12.13. Mosaic plot for the arthritis data, using the χ^2 test and fixed cut-off points for the shading

ple the maximum of the absolute values of the Pearson residuals (Meyer et al., 2003), instead of the sum of squares:

$$M = \max_{i,j} |r_{ij}|. \quad (12.4)$$

Given a critical value c_α for this test statistic, all residuals whose absolute values exceed c_α violate the hypothesis of independence at level α (Mazanec and Strasser, 2000, Chap. 7). Thus, the interesting cells that provide evidence for the rejection of the independence hypothesis can easily be identified. As explained above, the conditional distribution of this test statistic under the null hypothesis can be obtained by simulation, by sampling tables with the same row and column sums n_{i+} and n_{+j} using, e.g., the Patefield algorithm (Patefield, 1981) and computing the maximum statistic for each of these tables. In Fig. 12.14, we again visualize the arthritis data, this time using the maximum test statistic and its 10 % and 1 % critical values as cut-off points. Now the tile shading clearly shows that the treatment is effective: significantly more patients in the treatment group exhibit marked improvements than would be expected for independence.

Summary

12.3.4

Uniform colors and color palettes should always be used to visualize areas, and the HCL color space provides a convenient way to do this. Shadings can be used to add information to the basic plots and to support the analysis. Tile highlighting can sup-

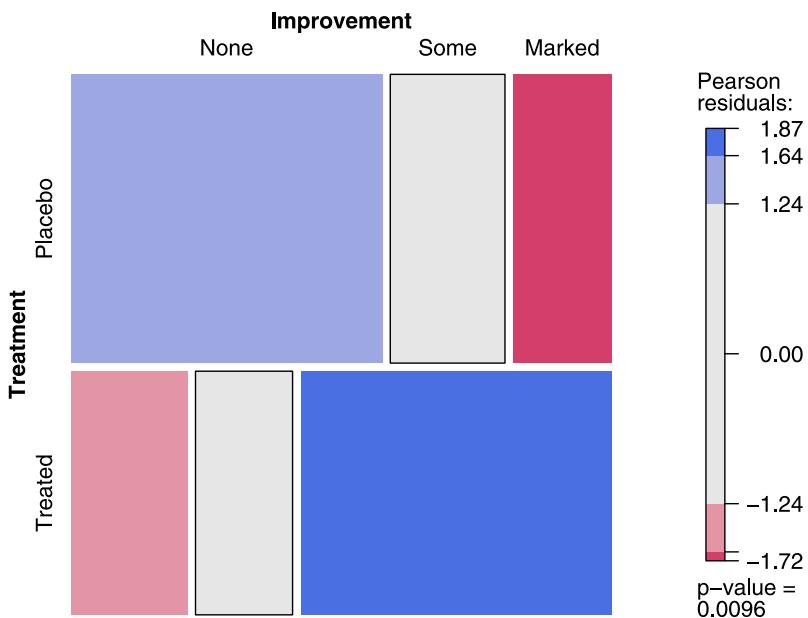


Figure 12.14. Mosaic plot for the arthritis data, using the maximum test and data-driven cut-off points for the residuals

port the analysis of relationships with a single dependent variable. Residual-based shadings can be used for model diagnostics, e.g., by using diverging color palettes to code the sign and size of a residual. In addition, the non-significance of test statistics can be visualized by using less colorful palettes. Using the maximum instead of the χ^2 (or other) test statistic(s) allows for data-driven cut-offs in the diverging palettes and precise diagnostic identification of the cells in conflict with the null hypothesis of independence.

12.4

Selected Methods for Multiway Tables

In Sect. 12.2 we presented basic displays of two-way tables, based on the visualization of information in table cells that are arranged in rectangular form. For multiway tables, mosaicplots can be used directly by simply adding further splits for each additional variable. For sieve and association plots, we apply the basic idea of mosaicplots to the table itself, i.e., we simply nest the variables into rows and columns using recursive conditional splits, given the margins. The result is a “flat” representation of the multiway table that can be visualized in a way similar to a two-dimensional table.

In the following, we will first treat specialized displays for exploratory purposes, followed by model-based methods for conditional independence models.

Exploratory Visualization Techniques

12.4.1

As an example, consider the well-known UCB admissions data (Bickel et al., 1975) on applicants, classified by admission and gender, to graduate school at the University of California in Berkeley for the six largest departments in 1973. The “flattened” contingency table, which associates departments with columns and admissions – nested by gender – with rows, is shown as Table 12.7. Aggregated over all departments, the table gives a false impression of gender bias, i.e., higher admission rates for male students.

A first step in the exploratory analysis of more complex tables is to get a quick overview of the data. For this, all basic plots can be combined in pairwise displays, arranged in a matrix similar to scatterplots in a pairs plot. The diagonal cells contain the variable names, optionally with univariate statistics, whereas the off-diagonal cells feature plots whose variables are implicitly specified by the cells’ positions in the matrix. In Fig. 12.15, the diagonal cells show bar plots for the distributions of the variables, and the off-diagonal cells mosaicplots for the corresponding pairs of variables. The plots suggest that admission differs between male and female students, and between the departments, and that the proportion of male and female students varies across the departments. In particular, departments A and B have higher proportions of male students and lower rejection rates than the other departments.

The next step is to investigate three-way interactions among the variables. In this example, we have a binary variable of interest (admission) that needs to be “explained” by the others. A natural way of representing such a three-way table is to use a mosaic display, which first involves splitting by the explanatory variables department and gender and then highlighting the resulting mosaic with respect to the dependent variable admissions. Here, we use vertical splits for both explanatory variables, resulting in the doubledecker plot in Fig. 12.16. From the widths of the tiles, it is clear that students apply to the six departments in unequal numbers (with a particularly small number of females in departments A and B), and from the highlighting we can also see that the admission rates differ among the departments (roughly speaking, the admission rate is high for A and B, low for F, and in-between for C to E). The rates are equal for male and female students, except for department A where *more* female than male students are admitted.

Table 12.7. The UCB admissions data, in flat representation

Gender	Admission	Department					
		A	B	C	D	E	F
Male	Admitted	512	353	120	138	53	22
	Rejected	313	207	205	279	138	351
Female	Admitted	89	17	202	131	94	24
	Rejected	19	8	391	244	299	317

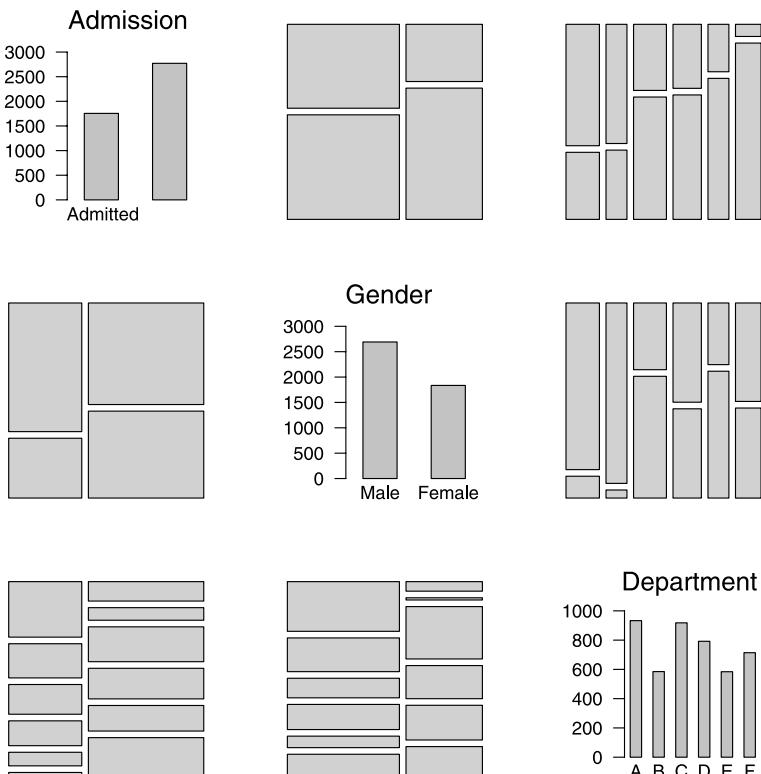


Figure 12.15. Pairs plot for the UCB admissions data

Model-Based Displays for Conditional Independence Models

In addition to the exploratory approaches to the visualization of multiway tables outlined in the previous section, there are specialized displays that are designed for the visualization of conditional independence structures.

One approach is to use pairs plots (such as in Fig. 12.15) to search for the model; we can use the positions of the cells in the pairs matrix to select an independence model and add corresponding shading to the tiles to visualize the residuals. More precisely, each cell a_{ij} in such a matrix defines two variables i and j that can be used to specify the model visualized in that cell. Typical hypotheses are: variables i and j are marginally independent; variables i and j are conditionally independent, given all others; variables i and j are jointly independent from all others.

Another approach is to visualize (the deviations from) a particular fitted model using ideas similar to those described in Sect. 12.3. Thus, mosaic displays can be extended from purely explorative views to model-based views of the data by residual-based shadings, and association plots can directly visualize deviations from a given

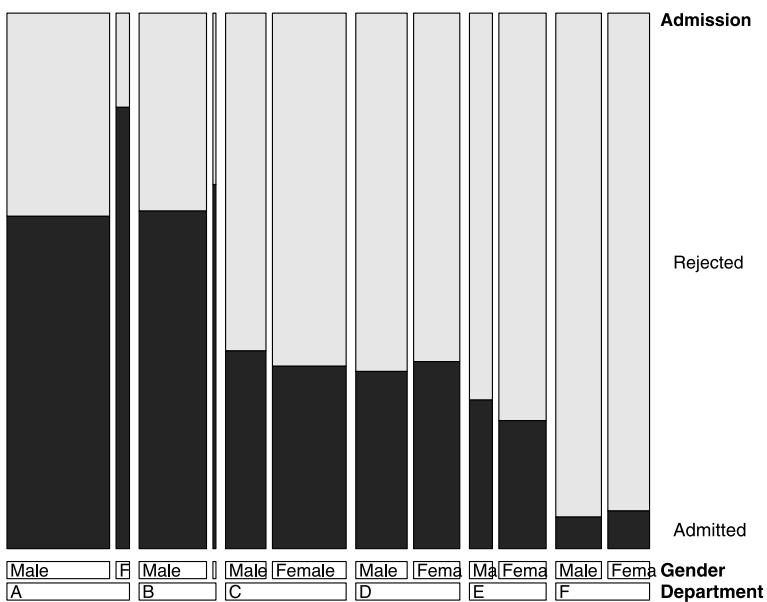


Figure 12.16. Doublededecker plot for the UCB admissions data

model. The models considered are log-linear models, but residual-based shadings are also conceivable for other types of models, such as logistic regression. In our example, from exploratory analysis, we already suspect that admission and gender are independent for a given department, except in the case of department A. Therefore, a sensible way of representing the table is to nest admission into department instead of gender to obtain a stratified view. The corresponding mosaic and association plots are shown in Figs. 12.17 and 12.18. The shading visualizes a model where admission and gender are conditionally independent, given the department.

Clearly, there is no gender bias in the departments except for department A: here, significantly *more* female students are admitted than expected for independence. Mosaic displays are already considered to be an excellent visualization tool for log-linear models (Friendly, 1999; Theus and Lauer, 1999; Hofmann, 2001). Using the flat table representation, the association plot similarly extends from the simple two-way case to a structured residual plot applicable to the diagnostics of log-linear models. In particular, if a mosaic display contains very small or empty cells, the corresponding association plot could provide an easier way to detect deviation patterns than adding residual-based shading to the mosaicplot. Yet another approach is to plot a mosaicplot of the *expected* frequencies and to highlight the (relative) residuals in the cells, using different colors for positive and negative residuals (Theus and Lauer, 1999).

As we have seen, we can use *all of the plots* to analyze stratified data by first splitting based on the conditioning variables. However, this also displays the marginal distributions of the conditioning variables, which might make it more difficult to in-

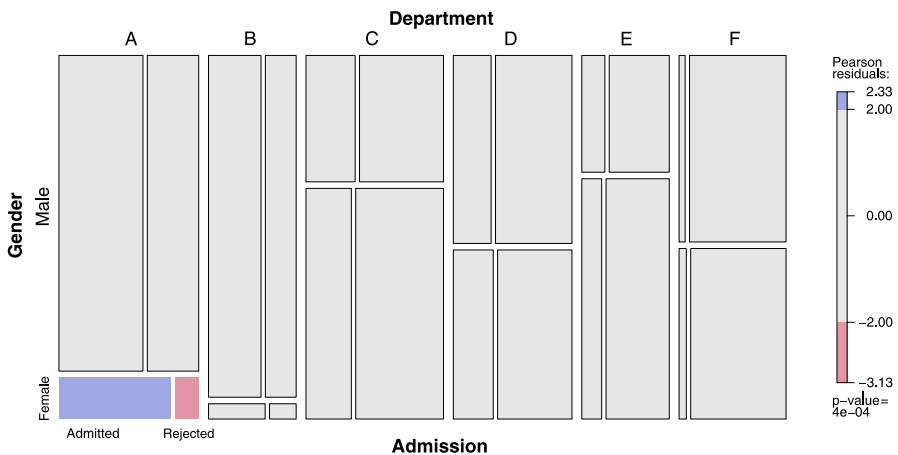


Figure 12.17. Mosaic plot for the UCB admissions data.

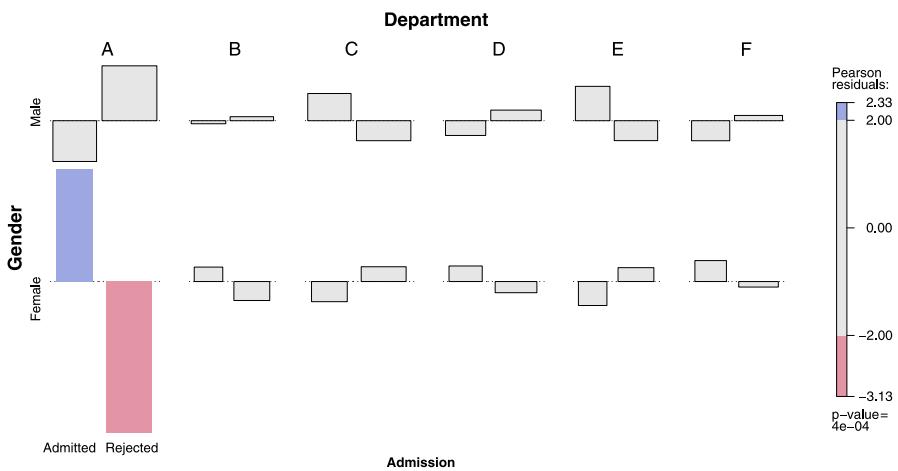


Figure 12.18. Association plot for the UCB admissions data.

terpret the dependent variables of interest. Particularly for very unevenly distributed marginals, the strata can become very distorted, and the tiles, along with their shadings, can become inscrutable. One alternative is to complement the conditioning in the model with conditioning in the plot; i.e., to use trellis-like layouts to visualize *partial* tables as defined by the conditioning variables. All subtables are corrected for marginals, and thus all corresponding plots in the panels are of the same size. In addition, trellis layouts help to reduce the complexity of bigger tables. Figure 12.19 visualizes the data by means of a conditional association plot. Each panel corresponds to a department, and contains an association plot corresponding to the partial (four-fold) table of admission and gender. Accordingly, the shading visualizes the residuals from the corresponding conditional independence model (independence of gender

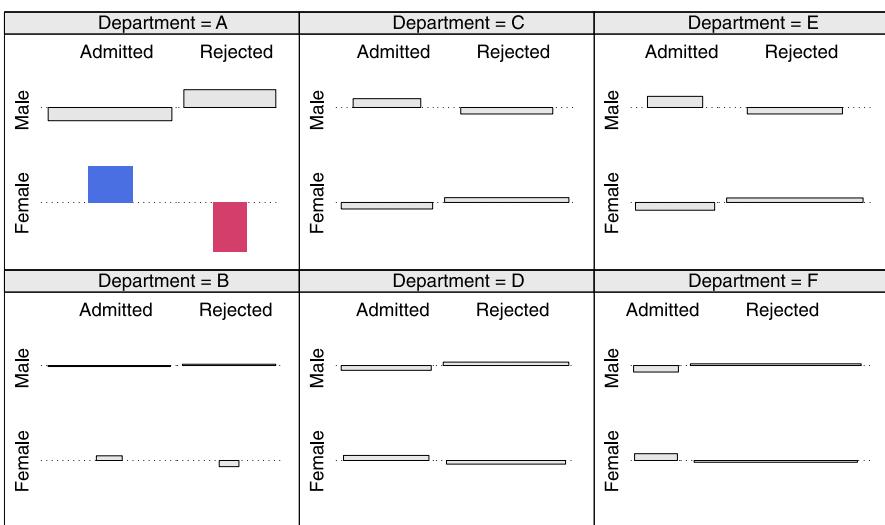


Figure 12.19. Conditional association plot for the UCB admissions data. For each individual association plot, rows correspond to gender and columns to admission

and admission, given department), stratified by department. Clearly, the situation in department A (more women/less men admitted than expected under the null hypothesis) results in the rejection of the hypothesis of conditional independence. In Fig. 12.19, data-driven cut-offs (derived from the maximum test for conditional independence) are used, resulting in more colorful shading than the fixed cut-offs in Figs. 12.17 and 12.18.

A Four-Way Example

12.4.3

As a four-way example, we use the punishment data (Andersen, 1991) from a study of the Gallup Institute in Denmark in 1979 regarding the attitude of a random sample of 783 people towards corporal punishment of children (see Table 12.8). The data consist of four variables. *Attitude* is a binary variable indicating whether a person approves of moderate punishment of children (“moderate”), or disapproves of any punishment of children (“no”). *Memory* indicates whether the person recalls having been punished as a child. *Education* indicates the highest level of education (elementary, secondary, high). Finally, age indicates the age group in years (15–24, 25–39, 40+).

In a first step, we create a mosaicplot for an exploratory view of the data: attitude towards punishment is clearly the response variable here and should be used for the last split via highlighting. Furthermore, we expect age and education to have an influence on both memory and attitude, so they should be used first for splitting as stratifying variables. In fact, the question is whether memory has some influence on attitude, given age and education. We choose to create a three-way mosaicplot of the hypothesized explanatory variables (first splitting horizontally by age, then vertically by education, and finally horizontally again by memory), and to have attitude,

Table 12.8. The punishment data

Education	Attitude	Age Memory	15–24		25–39		40+	
			Yes	No	Yes	No	Yes	No
Elementary	No		1	26	3	46	20	109
	Moderate		21	93	41	119	143	324
Secondary	No		2	23	8	52	4	44
	Moderate		5	45	20	84	20	56
High	No		2	26	6	24	1	13
	Moderate		1	19	4	26	8	17

the response variable, highlighted in the tiles (see Fig. 12.20). We can see that half of the people are more than 40 years of age, most of whom of only completed elementary school (much more than in the other age groups). Furthermore, it can be seen that the proportion of people who recall being punished increases with age, and that the approval rate decreases with education (this, however, is better illustrated in Fig. 12.21, as described below). For the question of whether attitude depends on memory, the plot quite clearly (and somewhat surprisingly) suggests that a higher proportion of those that had an elementary school education and that recall being punished tend to accept moderate punishment of children than those that do not recall being punished – those that have experienced punishment seem to support the use of punishment. For the other education groups, the picture is less clear: some cells indicate the same association, while others do not.

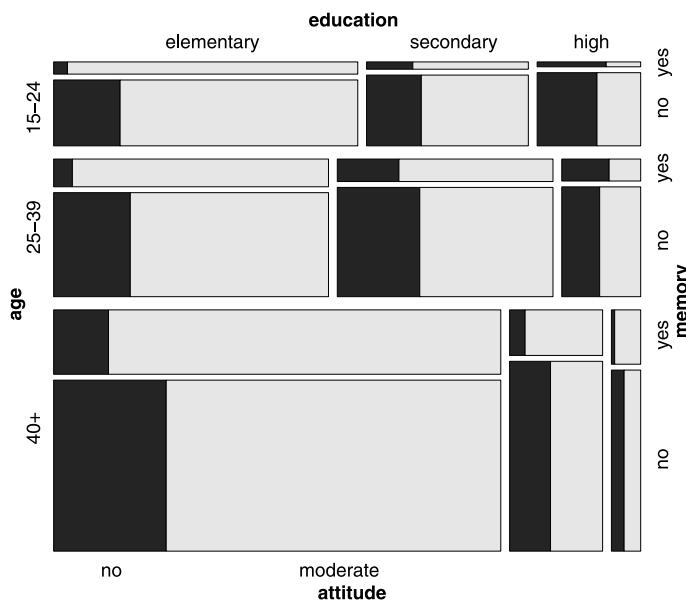


Figure 12.20. Mosaic plot with highlighting for the punishment data

In summary, both age and education clearly matter here. But is there really any significant influence of memory on attitude towards punishment apart from this? Or, rephrased as a hypothesis: are memory and attitude conditionally independent, given age and education? To focus on this question, we take another model-based view of the data which avoids the distracting influence of the marginal variables (age and education). We employ partial mosaicplots in a trellis layout (conditioning on age and education) with residual-based shading and data-driven cut-offs (see Fig. 12.21). This visualization illustrates much more clearly that the increased approval rate among people that recall being punished is present mainly in the elementary education column and, to a lesser degree, in the age 40+ row. Using a permutation test similar to the one discussed in Sect. 12.3.3, model statistics reveal that the hypothesis is rejected at a significance level of 1%, but the shading clearly shows that this association is only significant in two cells (the two older age groups with elementary education), where



Figure 12.21. Conditional mosaicplot for the punishment data. For each mosaicplot, rows correspond to memory (first split) and columns to attitude (second split)

fewer of the people that do not recall being punished do not approve of punishment than expected for (conditional) independence. For the group of people aged between 25 and 39, the lighter shading indicates that the association is only significant at the 10 % level.

The advantage of the exploratory view is its ability to visualize the joint distributions of all variables. On the other hand, this visualization may be strongly influenced by the marginal distribution of education over age (in particular the large proportion of 40+ people with elementary education), which is not relevant to the conditional independence problem. The partial mosaicplot suppresses this effect by complementing the conditioning in the model with conditioning in the visualization.

12.4.4

Summary

Mosaic, association, and sieve plots can be used to visualize multiway tables by converting them into flat representations. Mosaic plots are particularly useful for exploratory analysis, whereas the other two require that a particular model of independence be specified from which deviations can be examined. In addition, specialized plot “flavours” can leverage exploratory analyses (pairs plots, highlighting, doubledecker plots) or model-based analyses (residual-based shadings, conditional plots).

12.5

Conclusion

This chapter reviews several alternatives for the visualization of multiway contingency tables. For two-way tables, mosaic, sieve, and association plots are suitable for the visualization of observed and expected values and Pearson residuals, respectively. These basic methods can be enhanced by using residual-based shadings, preferably based on perceptual color palettes such as those derived from HCL space. Residual-based shadings can be used to visualize the signs and sizes of the residuals, as well as the significance of test statistics such as χ^2 or the maximum test statistic. The latter has the advantage that it detects residuals that cause the hypothesis of independence to be rejected. The methods extend directly to the multiway case by using “flat” representations of the multiway tables, and specialized displays for conditional independence such as trellis layouts of partial tables and pairs plots.

References

- Agresti, A. (2002). *Categorical Data Analysis*. Wiley, Hoboken, NJ, 2nd edn.
- Andersen, E.B. (1991). *The Statistical Analysis of Categorical Data*. Springer, Berlin, 2nd edn.
- Bickel, P.J., Hammel, E.A. and O’Connell, J.W. (1975). Sex bias in graduate admissions: Data from Berkeley. *Science*, 187:398–403.

- Brewer, C.A. (1999). Color use guidelines for data representation. In *Proceedings of the Section on Statistical Graphics, American Statistical Association*, Alexandria, VA, pp 55–60.
- Cleveland, W.S. and McGill, R. (1983). A color-caused optical illusion on a statistical graph. *The American Statistician*, 37:101–105.
- Cohen, A. (1980). On the graphical display of the significant components in a two-way contingency table. *Communications in Statistics – Theory and Methods*, A9:1025–1041.
- Commission Internationale de l’Éclairage (2004). *Colorimetry*. Publication CIE 15:2004, Vienna, Austria, 3rd edn. ISBN 3-901-90633-9.
- Ernst, M.D. (2004). Permutation methods: A basis for exact inference. *Statistical Science*, 19:676–685.
- Friendly, M. (1994). Mosaic displays for multi-way contingency tables. *Journal of the American Statistical Association*, 89:190–200.
- Friendly, M. (1999). Extending mosaic displays: Marginal, conditional, and partial views of categorical data. *Journal of Computational and Graphical Statistics*, 8(3):373–395.
- Friendly, M. (2000). *Visualizing Categorical Data*. SAS Institute, Carey, NC.
- Haberman, S.J. (1974). Log-linear models for frequency tables with ordered classifications. *Biometrics*, 30:689–700.
- Harrower, M.A. and Brewer, C.A. (2003). ColorBrewer.org: An online tool for selecting color schemes for maps. *The Cartographic Journal*, 40:27–37.
- Hartigan, J.A. and Kleiner, B. (1981). Mosaics for contingency tables. In W.F. Eddy (ed) *Computer Science and Statistics: Proceedings of the 13th Symposium on the Interface*, Springer, New York, pp 268–273.
- Hartigan, J.A. and Kleiner, B. (1984). A mosaic of television ratings. *The American Statistician*, 38:32–35.
- Hofmann, H. (2001). Generalized odds ratios for visual modelling. *Journal of Computational and Graphical Statistics*, 10:1–13.
- Hummel, J. (1996). Linked bar charts: Analysing categorical data graphically. *Computational Statistics*, 11:23–33.
- Ihaka, R. (2003). Colour for presentation graphics. In K. Hornik, F. Leisch and A. Zeileis (eds), *Proceedings of the 3rd International Workshop on Distributed Statistical Computing*, Vienna, Austria. ISSN 1609-395X.
- Insightful Inc. (2005). *S-PLUS 7*. Seattle, WA.
- Koch, G. and Edwards, S. (1988). Clinical efficiency trials with categorical data. In K.E. Peace (ed), *Biopharmaceutical Statistics for Drug Development*, pp 403–451. Marcel Dekker, New York.
- Ligges, U. and Mächler, M. (2003). scatterplot3d – an R package for visualizing multivariate data. *Journal of Statistical Software*, 8(11):1–20.
- Mazanec, J.A. and Strasser, H. (2000). *A Nonparametric Approach to Perceptions-based Market Segmentation: Foundations*. Springer, Berlin.

- Meyer, D., Zeileis, A. and Hornik, K. (2003). Visualizing independence using extended association plots. In K. Hornik, F. Leisch and A. Zeileis (eds), *Proceedings of the 3rd International Workshop on Distributed Statistical Computing, Vienna, Austria*. ISSN 1609-395X.
- Meyer, D., Zeileis, A. and Hornik, K. (2006). The strucplot framework: Visualizing multi-way contingency tables with vcd. *Journal of Statistical Software*, 17(3):1–48.
- Meyer, D., Zeileis, A. and Hornik, K. (2006). *vcd: Visualizing Categorical Data*. R package version 1.0-6.
- Mollon, J. (1995). Seeing color. In T. Lamb and J. Bourriaud (eds), *Colour: Art and Science*. Cambridge University Press, Cambridge.
- Moretti, G. and Lyons, P. (2002). Tools for the selection of colour palettes. In *Proceedings of the New Zealand Symposium On Computer-Human Interaction (SIGCHI 2002)*, University of Waikato, New Zealand.
- Munsell, A.H. (1905). *A Color Notation*. Munsell Color Company, Boston, MA.
- Patefield, W.M. (1981). An efficient method of generating $r \times c$ tables with given row and column totals. *Applied Statistics*, 30:91–97. Algorithm AS 159.
- Pesarin, F. (2001). *Multivariate Permutation Tests*. Wiley, Chichester, UK.
- Poynton, C. (2000). Frequently-asked questions about color.
<http://www.poynton.com/ColorFAQ.html>. Accessed 2006-09-14.
- SAS Institute Inc. (2005). *SAS/STAT Version 9*. Cary, NC.
- R Development Core Team (2006). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-00-3.
- Riedwyl, H. and Schüpbach, M. (1994). Parquet diagram to plot contingency tables. In F. Faulbaum (ed), *Softstat '93: Advances in Statistical Software*, pp 293–299, Gustav Fischer, New York.
- Strasser, H. and Weber, C. (1999). On the asymptotic theory of permutation statistics. *Mathematical Methods of Statistics*, 8:220–250.
- Theus, M. (2003). Interactive data visualization using Mondrian. *Journal of Statistical Software*, 7(11):1–9.
- Theus, M. and Lauer, S.R.W. (1999). Visualizing loglinear models. *Journal of Computational and Graphical Statistics*, 8(3):396–412.
- Unwin, A.R., Hawkins, G., Hofmann, H. and Siegl, B. (1996). Interactive graphics for data sets with missing values – MANET. *Journal of Computational and Graphical Statistics*, 4(6):113–122.
- Wing, J.K. (1962). Institutionalism in mental hospitals. *British Journal of Social Clinical Psychology*, 1:38–51.
- Young, F.W. (1996). ViSta: The visual statistics system. Technical Report 94-1(c), UNC L.L. Thurstone Psychometric Laboratory Research Memorandum.
- Zeileis, A., Meyer, D. and Hornik, K. (2007). Residual-based shadings for visualizing (conditional) independence. *Journal of Computational and Graphical Statistics*, 16(3):507–525.

Mosaic Plots and Their Variants III.13

Heike Hofmann

13.1	<i>Definition and Construction</i>	619
13.2	<i>Interpreting Mosaic Plots</i>	622
	Probabilities in Mosaic Plots	622
	Visualizing Interaction Effects	624
13.3	<i>Variants</i>	627
	Doubledecker Plots	627
	Fluctuation Diagrams	628
	Others	632
13.4	<i>Related Work and Generalization</i>	635
	Treemaps	635
	Trellis Plots	636
	Pivot Tables	638
13.5	<i>Implementations</i>	640

In this chapter we consider mosaicplots, which were introduced by Hartigan and Kleiner (1981) as a way of visualizing contingency tables. Named “mosaicplots” due to their resemblance to the art form, they consist of groups of rectangles that represent the cells in a contingency table. Both the sizes and the positions of the rectangles are relevant to mosaicplot interpretation, making them one of the more advanced plots around. With a little practice they can become an invaluable tool in the representation and exploration of multivariate categorical data.

In this chapter we will be discussing ways of constructing and interpreting mosaicplots, including their connection to loglinear models (Hofmann, 2001; Theus and Lauer, 1999; Friendly, 1992). In Sect. 13.1 we will be discussing ways of constructing mosaicplots (Hofmann, 2003). Mosaic plots have the huge advantage of preserving all of the information in multivariate contingency tables while simultaneously providing an overview of it. As the mosaicplot follows the hierarchy its corresponding contingency table exactly, the order of the variables in the table is important. Selecting the “right,” or at least “good,” ordering is commonly found to be one of the main difficulties first-time users experience with mosaicplots. We will discuss the effects of changes in the order and provide recommendations about how to obtain “good plots.”

Multivariate categorical modeling is usually done with loglinear models. It can be shown (Hofmann, 2001; Theus and Lauer, 1999; Friendly, 1992) that mosaicplots have excellent mathematical properties which enable the strengths of interaction effects to be assessed visually and provide tools for checking residuals and modeling assumptions. We will discuss the relationship between mosaicplots and loglinear models in Sect. 13.2.

Close relatives of the mosaicplot, such as fluctuation diagrams and doubledecker plots (Hofmann et al., 2000), have also been found to be very useful in practice. We are therefore going to have a look at those and other important variants of mosaicplots too in Sect. 13.3. All of these variants are essentially simplifications of the default mosaic construction. While some information is lost in the process, these plots place additional emphasis on a specific aspect of the data.

Shneiderman (1992) and trellis plots (Becker et al., 1994) are generalizations of two different aspects of mosaicplots. While trellis plots and mosaicplots share the same structure, trellis plots are more flexible since numbers do not necessarily have to be displayed as rectangles. Treemaps, on the other hand, always use rectangles, but are able to deal with more general partitions than mosaicplots. These generalizations do not come without losses, though. We will compare mosaicplots to these other displays in Sect. 13.4, and comment on the strengths and weaknesses of each. Software implementations of mosaicplots are becoming more frequent. An implementation in R (Gentleman and Ihaka, 1995) was created by Emerson (1998). Mosaic plots in JMP (John Sall, 1989) have some limited interactive features. Fully interactive mosaicplots (Hofmann, 2000) are implemented, e.g., in MANET (Unwin, Hawkins, Hofmann and Siegl, 1997), Mondrian (Theus, 2002) and KLIMT (Urbanek, 2002).

Definition and Construction

Mosaic plots were introduced by Hartigan and Kleiner (1981) as a means of visualizing contingency tables. More recent works on them include those of Hartigan and Kleiner (1984), Friendly (1994), Friendly (1995), Friendly (1999), and Hofmann (2003). As the name suggests, mosaicplots consist of groups of rectangular tiles. Each tile corresponds to one cell from a contingency table. Its area is proportional to the size of the cell, and its shape and location are determined during the construction process.

Table 13.1 provides a breakdown of people on board the Titanic according to their class and gender, as reported in Dawson (1995). Figure 13.1 shows the corresponding mosaicplot.

Mosaic plots are not restricted to two dimensions; in principle they can be extended to an arbitrary number of dimensions. In practice, however, space is a limiting factor.

Construction of a p -dimensional mosaicplot:

Let us assume that we want to construct a mosaicplot for p categorical variables X_1, \dots, X_p . Let c_i be the number of categories of variable X_i , $i = 1, \dots, p$.

1. Start with one single rectangle r_0 (of width w_0 and height h_0), and let $i = 1$.
2. Cut rectangle r_{i-1} into c_i pieces: find all observations corresponding to rectangle r_{i-1} , and find the breakdown for each variable X_i (i.e., count the number of observations that fall into each of the categories). Split the width (height) of

Table 13.1. Breakdown of Gender versus Class for those aboard the Titanic (Dawson, 1995)

		Class				Totals
		1st	2nd	3rd	Crew	
Sex	f	145	106	196	23	470
	m	180	179	510	862	1731
Totals		325	285	706	885	2201

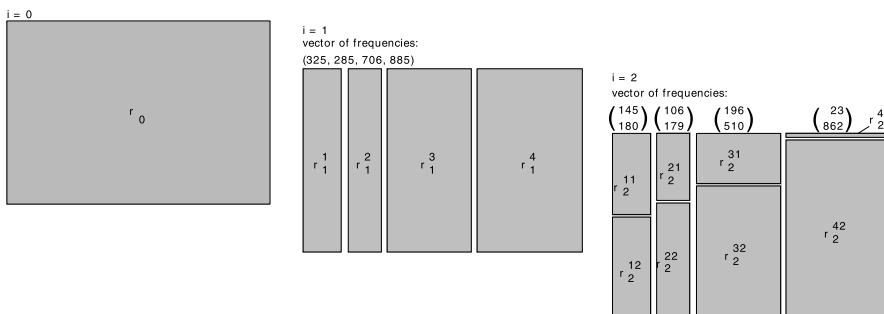


Figure 13.1. Stepwise construction of a two-dimensional mosaicplot of Gender distribution across different classes based on the data given in Table 13.1

rectangle r_{i-1} into c_i pieces, where the widths (heights) are proportional to the breakdown, and keep the height (width) of each the same as r_{i-1} . Call these new rectangles r_i^j , with $j = 1, \dots, c_i$.

3. Increase i by 1.

4. While $i \leq p$, repeat steps 2 and 3 for all r_{i-1}^j with $j = 1, \dots, c_{i-1}$.

Obviously, the hierarchical construction of mosaicplots places a lot of emphasis on the order of the variables in the plot. As is the case for multiway contingency tables, different variable orders in mosaics emphasize different aspects of the data; in contrast with two-way tables, the order in which the variables should occur in a multiway table is not immediately apparent. Tables 13.2 and 13.3 show two different arrangements for the same variables of the Titanic data. Each table contains the same data for Class, Gender and Survival. Table 13.2 emphasizes the different survival rates of men and women in all classes, whereas the arrangement in Table 13.3 emphasizes the notion that the lower the passenger class, the lower the survival rate, to for both women and men. These different tables arise because the order of the variables *Class* and *Gender* can be changed.

Figures 13.2 and 13.3 each show a three-dimensional mosaicplot of **Class**, **Gender** and **Survival**. Changing the order of the variables emphasizes different aspects of the data. The survivors are highlighted in both mosaicplots. In Fig. 13.2, differences in the survival rates of women and men are highlighted for all classes. The mosaicplot in Fig. 13.3 shows the survival rates within the different classes (first, second, third and crew) for men and women.

Table 13.2. Number of Survivors from the Titanic by class affiliation and Gender. There are large differences between the survival rates of men and women across all classes

Class:	First		Second		Third		Crew	
Gender:	Female	Male	Female	Male	Female	Male	Female	Male
Survived:								
yes	141	62	93	25	90	88	20	192
no	4	118	13	154	106	422	3	670
Survival Rate (in %)	97	34	88	14	46	17	87	22

Table 13.3. Number of Survivors from the Titanic by Gender and class. The survival rates decline as the passenger class decreases

Gender:	Female				Male			
Class:	First	Second	Third	Crew	First	Second	Third	Crew
Survived:								
yes	141	93	90	20	62	25	88	192
no	4	13	106	3	118	154	422	670
Survival Rate (in %)	97	88	46	87	34	14	17	22

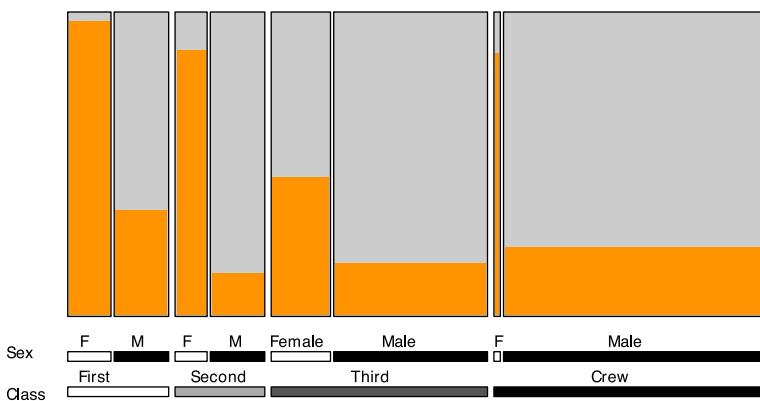


Figure 13.2. Three-dimensional mosaicplot of the Titanic data. Survivors are highlighted. Differences between the survival rates of men and women are emphasized. In all classes, the women had higher survival rates than the men

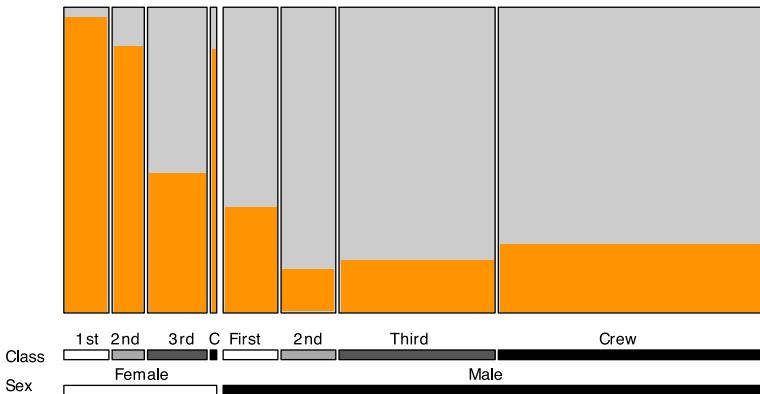


Figure 13.3. Three-dimensional mosaicplot of the Titanic data. Survivors are highlighted. Survival rates for men (right) and women (left) in all of the classes are plotted. The survival rate for women increases with class. The survival rate for men shows a peculiarity: it is extremely low for men in the second class

Figures 13.2 and 13.3 were constructed using two successive splits in the horizontal direction and then a final split in the vertical direction. For the third dimension, we colored some of the rectangles to show the splits and did not physically split them. Variations like these are not specifically fixed in the algorithm. The original mosaicplot, as defined by Hartigan and Kleiner (1981), has splits in alternating directions, whereas doubledecker plots (Hofmann et al., 2000) only have horizontal splits, except for the last dimension, which is split in the vertical direction (and usually shows highlighting).

To distinguish between mosaicplots that show the same data but refer to structurally different contingency tables, we must consider extra information about the plot: we will call this additional information the *structure or frame* (Wilkinson, 1999)

of a mosaicplot. This structure contains the meta-information of a set of categorical variables:

- the values that each variable can adopt and the order in which these values occur (i.e., a tuple of the variable's categories)
- the order and direction in which the variables appear in the hierarchy.

Throughout this chapter, we will assume that the default split direction in a mosaicplot is horizontal. If a vertical split is intended, we will use the superscript t behind the variable name (denoting the “transpose” of the variable). The mosaicplots in Figs. 13.2 and 13.3 therefore have structures *Class*, *Gender*, *Survival* t and *Gender*, *Class*, *Survival* t , respectively. The concept of mosaicplot structure is discussed in more depth in Hofmann (2003).

The strictly hierarchical construction of mosaicplots enables us to interpret various properties of mosaicplots, as we will discuss in the next section.

13.2

Interpreting Mosaic Plots

Mosaics are area-based graphics: the area of each tile is proportional to the cell size of the corresponding contingency table. However, the overall number of observations is not displayed in a mosaicplot; multiplying each cell by ten does not change the plot because the ratios between the cell counts are illustrated, not the cell counts themselves. It is therefore more relevant to discuss percentages or probabilities in relation to mosaicplots rather than actual cell sizes. Tile sizes should always be interpreted in relation to the sizes of other tiles. After mosaic construction, we will describe the mathematical properties that we can retrieve from the plot, i.e., what we can reliably “see from a mosaicplot.”

13.2.1

Probabilities in Mosaic Plots

Figure 13.4 shows a two-dimensional mosaicplot of the Titanic data. Class is plotted versus Survival. The first split is in the horizontal direction; the width of a tile is based on the number of persons in each class. The width is therefore an estimate of the probability of $P(\text{Class} = i)$, with $i \in \{1\text{st}, 2\text{nd}, 3\text{rd}, \text{Crew}\}$. The number of survivors in each of these tiles is then found, which determines the height of the final tiles. This makes the height of the bottom left tile an estimate of the survival probability of the first class passengers: $P(\text{Survival} = \text{yes} \mid \text{Class} = 1\text{st})$, whereas the upper left tile's height gives $P(\text{Survival} = \text{no} \mid \text{Class} = 1\text{st})$. Consequently, the area of a tile gives the joint probability for Survival and Class, since

$$\begin{aligned} P(\text{Survival}=\text{yes} \cap \text{Class}=1\text{st}) &= \underbrace{P(\text{Survival}=\text{yes} \cap \text{Class}=1\text{st})}_{\text{area}} \\ &= \underbrace{P(\text{Survival}=\text{yes} \mid \text{Class}=1\text{st})}_{\text{height}} \cdot \underbrace{P(\text{Class}=1\text{st})}_{\text{width}} \end{aligned}$$

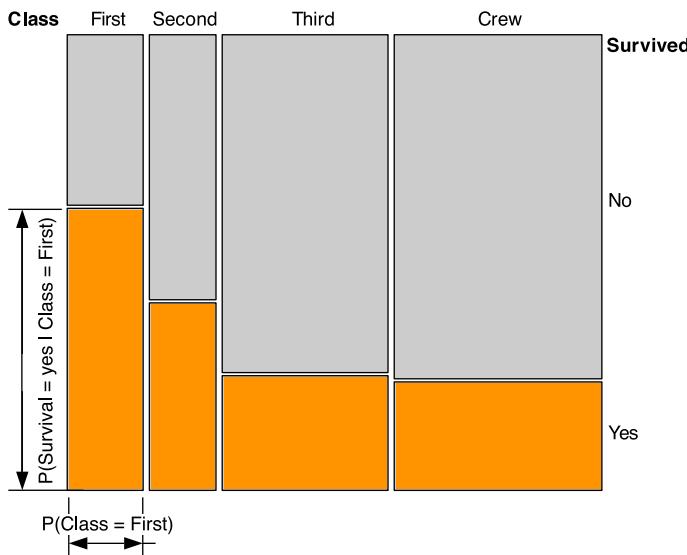


Figure 13.4. Two-dimensional mosaicplot of Class versus Survival. Each tile's width corresponds to the probability of the associated class, while its height is the conditional probability of (Non-)Survival

For high-dimensional mosaicplots, the structure of the plot governs which of the probabilities correspond to the heights and the widths of the tiles. However, the area of a particular tile always corresponds to the joint probability of X_1, \dots, X_p .

Let $v(i)$ be the set of variables among the first i variables that trigger splits in the vertical direction. Similarly, let $h(i)$ be the set of variables among the first i variables that split horizontally.

Obviously, $\emptyset = v(0) \subset v(1) \subset v(2) \subset \dots \subset v(p)$ and $v(p) \cup h(p) = \{X_1, \dots, X_p\}$. Let $X_{(i)}$ be the i th variable in a mosaicplot. In the default construction by Hartigan and Kleiner (1981), $v(i)$ consists of all variables with even indices that are less than i , and $h(i)$ contains all of the variables with odd indices:

$$v(i) = \{X_{(2k)} \mid 0 \leq 2k \leq i\} \text{ and } h(i) = \{X_{(2k+1)} \mid 1 \leq 2k + 1 \leq i\}$$

For a doubledecker plot, the sets $v(i)$ and $h(i)$ can be written as:

$$v(i) = \emptyset \text{ for all } i = 1, \dots, p-1 \text{ and}$$

$$v(p) = \{X_{(p)}\}$$

$$h(i) = \{X_{(1)}, \dots, X_{(i)}\} \text{ for all } i = 1, \dots, p-1 \text{ and}$$

$$h(p) = h(p-1).$$

The width of each tile can then be interpreted as $P(h(p) \mid v(p))$ and its height is $P(v(p) \mid h(p))$. After the i th split, the width of a tile is $P(h(i) \mid v(i))$ and its height is $P(v(i) \mid h(i))$.

Assuming that the order of the variables in the mosaicplot is X_1, \dots, X_p , we can make the above equations a bit more specific: if variable X_i is presented in a horizontal split of the plot, the width of each tile represents

$$P(h(i) | v(i)) = P(X_i | v(i)) \cdot P(h(i-1) | v(i)),$$

i.e., the height is given by the conditional probability of X_i given previous variables with splits in the vertical direction, while the overall width of the tile is given by previous splits in the horizontal direction. If we consider what can actually be seen in a mosaicplot, the way to specify the order in which variables should be addressed in a mosaic becomes clear. Since we are working in this highly conditioned framework, stratum variables should be addressed in a mosaicplot first, while the most important variables should be addressed last.

13.2.2

Visualizing Interaction Effects

One way to measure the strength of association between two binary variables X and Y is to compute the odds ratio: denote the cell probabilities of a 2×2 table with variables X and Y by $\pi_{11}, \pi_{12}, \pi_{21}$ and π_{22} , such that $\pi_{ij} = P(X = X_i \cap Y = Y_j)$, where X_i is the i th category of X and Y_j is the j th category of Y .

The odds ratio (cross-product ratio) θ between two binary variables X and Y provides a measure of the association between them. It is defined as

$$\theta = \frac{\pi_{11}\pi_{22}}{\pi_{12}\pi_{21}}.$$

Values for the odds ratio range between 0 and $+\infty$; values of θ close to 0 mean a strong negative association of X and Y , and large positive values indicate a strong positive association. When $\theta = 1$ the variables X and Y are assumed to be independent. This asymmetry in values causes many people to work with the logarithm of the odds ratio instead, which ensures symmetric behavior, i.e., a value of $\log \theta = q$ is just as strong an association as $\log \theta = -q$, with $\log \theta = 0$ indicating independence.

Figure 13.5 shows several examples of mosaicplots of 2×2 contingency tables. The amount of interaction between the variables increases from left to right. Visually, this is indicated by an increase in the measure d , where d is the difference in conditional probabilities $d = m_{12}/(m_{11} + m_{12}) - m_{22}/(m_{21} + m_{22})$. It can be shown (see Hofmann, 2001) that this difference d is approximately linear in $\log \theta$, the logarithm of the odds ratio for the four cells, or more specifically:

$$d \approx -0.25 \cdot \log \theta$$

The approximation holds so long as either $m_{11}/m_{12} \approx m_{22}/m_{21}$ or $m_{11}/m_{21} \approx m_{22}/m_{12}$.

Statements about whether or when d indicates a significant interaction between the variables cannot be drawn directly from a diagram, since only the odds ratio is visualized, which is independent of the underlying sample size. Comparisons of odds ratios, however, can be made directly. From the two mosaicplots on the right of Fig. 13.5, we can see that the odds ratio for the plot on the right is about twice the size

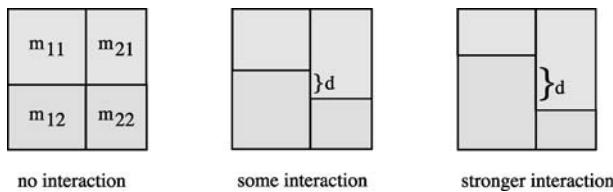


Figure 13.5. Three 2×2 mosaicplots; the interaction between the variables increases from left to right

as for the middle mosaicplot, since d on the right hand side is about twice as big as d in the middle diagram. The advantage of being able to read off the odds ratio from a mosaicplot is that it can then be used in modeling: for two binary variables X and Y , the model of independence can be written as

$$\log m_{ij} = \mu + \lambda_i^X + \lambda_j^Y,$$

where m_{ij} is the cell count of cell (i, j) , μ is the grand mean, λ_i^X is the effect of the i th level of variable X , and λ_j^Y is the effect of the j th level of variable Y (with $1 \leq i, j \leq 2$). This model holds if the interaction term λ_{ij}^{XY} is not significantly different from zero. Using control group constraints, i.e., all “first effects” are set to zero for identifiability, the interaction term between X and Y is equal to the logarithm of the odds ratio between the variables:

$$\lambda_{22}^{XY} = \log \theta.$$

A three-way interaction between variables X , Y , and Z is present if the (conditional) two-way interaction between X and Y is different for different levels of Z . The trick therefore is to compare odds ratios. For three variables X , Y and Z , the corresponding (conditional) odds ratio between X and Y for a fixed level k of Z is defined as:

$$\theta_{ij(k)} = \frac{m_{ijk} m_{i+1j+1k}}{m_{i+1jk} m_{ij+1k}}.$$

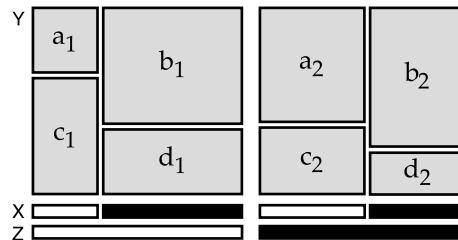


Figure 13.6. Mosaic plot of binary variables X , Y , and Z with structure Z, X, Y' . By comparing the two conditional odds ratios $\theta_1 = a_1 d_1 / (b_1 c_1)$ and $\theta_2 = a_2 d_2 / (b_2 c_2)$, conclusions can be made about the three-way interaction between the variables visually

Using the same argument as above, we can read off approximate values for the (conditional) logarithms of the odds ratios $\log \theta_1$ and $\log \theta_2$. Finding the difference between these logarithms of odds ratios then leads us to a generalization of odds ratios to dimension three (Bhapkar and Koch, 1968):

$$\log \theta_1 - \log \theta_2 = \log \theta_1/\theta_2 =: \log \theta_{XYZ}.$$

Using group constraints, the three-way interaction effect between X , Y and Z is equal to the logarithm of the odds ratio:

$$\lambda_{222}^{XYZ} = \log \theta_{XYZ}.$$

The absence of a three-way interaction therefore corresponds to $\theta_{XYZ} = 1$, which in turn corresponds to equal conditional odds ratios θ_1 and θ_2 . High-dimensional interaction effects can be found visually in a similar fashion. This is explained in more detail in Hofmann (2001).

For variables with multiple categories, we will suppose that at least one of the variables of interest is binary. For a visual assessment of the interaction, we will use the same approach as before. An example is shown in Fig. 13.7, based on the College Plans data set (Sewell and Shah, 1968). Highlighted (marked in orange) are pupils whose parents strongly encourage them to go to college. Highlighted heights show the conditional probability of encouragement given a pupil's IQ. The percentage of parental encouragement increases as the IQ increases. The increase in high parental encouragement with each successive IQ level (denoted by d_1 , d_2 and d_3) stays approximately equal.

For multivariate contingency tables like the example, there are multiple ways of obtaining odds ratios. A sufficient set of odds ratios, though, is the set of odds ratios gained by computing the odds ratios of all quadruples of four adjacent cells (see, e.g., Agresti, 1990).

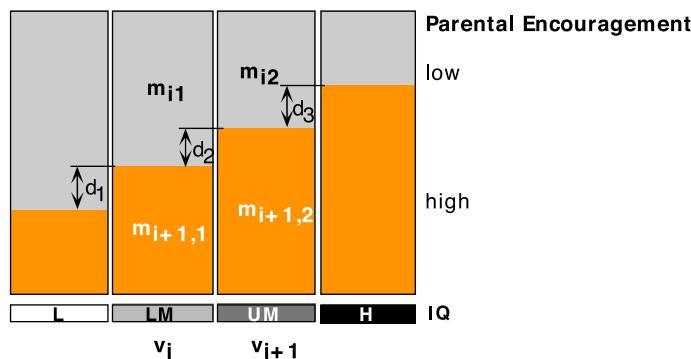


Figure 13.7. Doublededecker plot of parental encouragement vs. IQ. Highlighting reveals an approximately linear trend of encouragement vs. IQ

We will denote the odds ratio corresponding to the set of cells $\{m_{ij}, m_{i+1,j}, m_{i,j+1}, m_{i+1,j+1}\}$ as θ_{ij} , i.e.,

$$\theta_{ij} = \frac{m_{ij}m_{i+1,j+1}}{m_{i+1,1}m_{i,j+1}} \text{ for all } 1 \leq i \leq I-1, 1 \leq j \leq J-1.$$

The differences in the highlighted heights, d_1, d_2 and d_3 , are approximately linear in the log odds ratio of the corresponding cells:

$$-4 \cdot d_i \approx \log \theta_i = \log \frac{m_{i1}m_{(i+1)2}}{m_{(i+1)1}m_{i2}} \text{ for } i = 1, 2, 3.$$

The fact that these differences do not change much indicates a linear relationship (on a log scale) between the percentage of high parental encouragement and the IQ level.

Obviously the two variables X and Y are independent if $\log \theta_{ij} = 0$ for all i and j . By visualizing low-dimensional relationships between variables, mosaicplots provide us with a way to find models graphically and to check the fits of existing models. Earlier approaches by Friendly (1992) and Theus and Lauer (1999) established the link between loglinear models and mosaicplots by displaying fitted values of loglinear models together with their residuals.

Variants

13.3

By default, the sizes, shapes and locations of tiles in a mosaicplot are determined by the mosaic's hierarchical construction process. However, it is of course possible to vary these features, and this is a very useful technique for exploring the data, as each of them tends to emphasize different aspects of the data. We are going to discuss two different groups of variations: doubledecker plots are, as mentioned earlier, a variant of the default structure of a mosaicplot. All other variations discussed here refer to the layout of the plot.

Doubledecker Plots

13.3.1

Doubledecker plots were introduced in Hofmann et al. (2000) as a way of visualizing an association rule within the framework of the whole contingency table. Doubledecker plots are a special case of standard mosaicplots. Instead of splitting the bins alternately in horizontal and vertical directions as in a default mosaicplot, all of the bins are split horizontally in a doubledecker plot. All of the bins in a doubledecker plot are the same height and are drawn side by side. Doubledecker plots of p variables X_1, \dots, X_p therefore have the structure $X_1, X_2, \dots, X_{p-1}, X'_p$, where X_p is usually the highlighting variable. Thus, highlighting the heights in a p -dimensional mosaicplot illustrates the conditional probabilities $P(X_p | X_1, \dots, X_{p-1})$. One thing that distinguishes doubledecker plots from standard mosaicplots is that they are easy to label. This is done by drawing a striped bar below the graphic for each of the variables. The stripes show different shades of gray, with each shade representing one category of

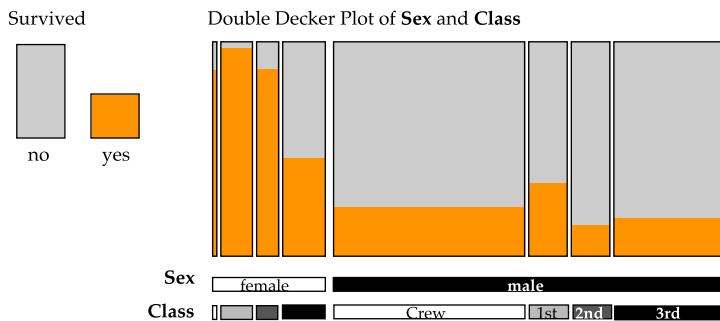


Figure 13.8. Doubledecker plot of the Titanic data. Survivors are highlighted. The highest survival rates appear for the women in the crew and in the first and second classes

the corresponding variable. The stripe width is determined by the width of the tile in the doubledecker plot. The exact combination that a specific bin represents can be read from these labels by drawing a (virtual) vertical line from the bin through the bars below. The shades that this line passes through give the exact combination of the bin. The first bin of the doubledecker plot in figure 13.8 therefore is the combination of Sex = female \cap Class = Crew, while the second bin shows all female passengers in the first class, and so on.

13.3.2 Fluctuation Diagrams

In a *fluctuation diagram* the area of each rectangle represents the number of cases, but the position of the bottom right corner of a bin is fixed on a grid. This display emphasizes large bins, while small or empty bins simply vanish.

Fluctuation diagrams are constructed as follows. All of the tiles are laid out on a rectangular grid, and each tile is initially the same size. The information on the cell sizes is included by shrinking both the height and the width of each tile equally, such that the remaining area is proportional to the cell size. If the original display is quadratic, the height and the width of the bin end up being proportional to the square root of the cell size. This makes comparisons between the widths and the heights of the bins possible at the same time.

Figure 13.9 shows a mosaic of the College Plans data (Sewell and Shah, 1968). The intelligence quotient (IQ) of a pupil is plotted versus the socioeconomic status (SES) of his/her parents. The mosaic has the structure IQ, SES'. All of the pupils who intended to go to college are highlighted. In this representation, we are able to detect three trends graphically without the need for any calculations. Let us first consider the highlighted values. The highlighted areas grow from left to right and from top to bottom. This type of highlighting is required in order to pick out both of the following trends simultaneously: first, pupils with high parental SES are more likely to go to college, and second, the percentage of college-goers also increases with IQ.

As well as these two trends, a connection between the social status of the parents and the IQ of the pupil is visible – as the social status of the parents increases the IQ increases, and vice versa.

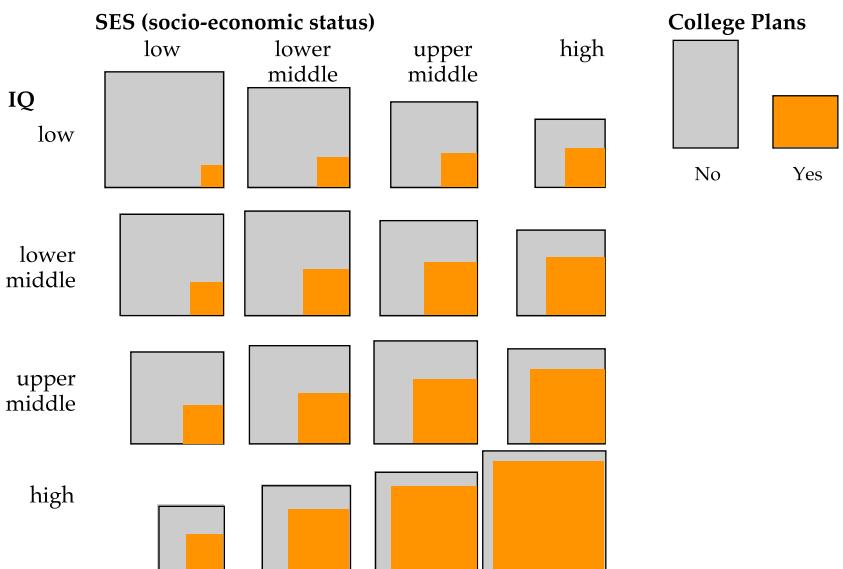


Figure 13.9. Two-dimensional mosaicplot in fluctuation mode. Highlighting shows pupils who intend to go to college. Three trends are recognizable visually

Another area in which fluctuation diagrams are applied in matrix visualization. Figure 13.10 shows an example of a 36×36 correlation matrix for three years of monthly temperature averages from a set of locations. Large cells indicate highly correlated variables, while small cells correspond to correlation values that are close to zero. Negative correlations don't occur in this example, but would be displayed as their absolute values. The dominant feature in Fig. 13.10 is provided by the strong seasonal trends; variables in summer months are highly correlated with each other and between years, and are close to being independent of variables for winter month temperatures. Spring and fall months are weakly correlated with all temperature variables.

For a matrix of cases by variable (the standard spreadsheet format), we might be tempted to use a fluctuation diagram for an initial overview; since mosaicplots take categorical data as input, and fit each tile according to the number of cases within each combination, we can only visualise whole numbers. Instead of counting the number of cases within each combination of variables, we now use weights for each case and sum those. The size of a tile then is given by the total sum of weights for the cases of each combination. Plots like these are sometimes called *permutation matrices* (Bertin, 1967) or *Bertin plots* (Falguerolles et al., 1997).

For case-by-variables matrices, we therefore use column labels and row labels as two categorical variables and the entries of the matrix give us the weights. For the Mammals' Milk data (Hartigan, 1975), we have a variable "mammal", consisting of a list of the mammals' names: HORSE, ORANGUTAN, MONKEY, DONKEY, HIPPO, CAMEL, BISON, ..., and a second variable "ingredients", with categories:

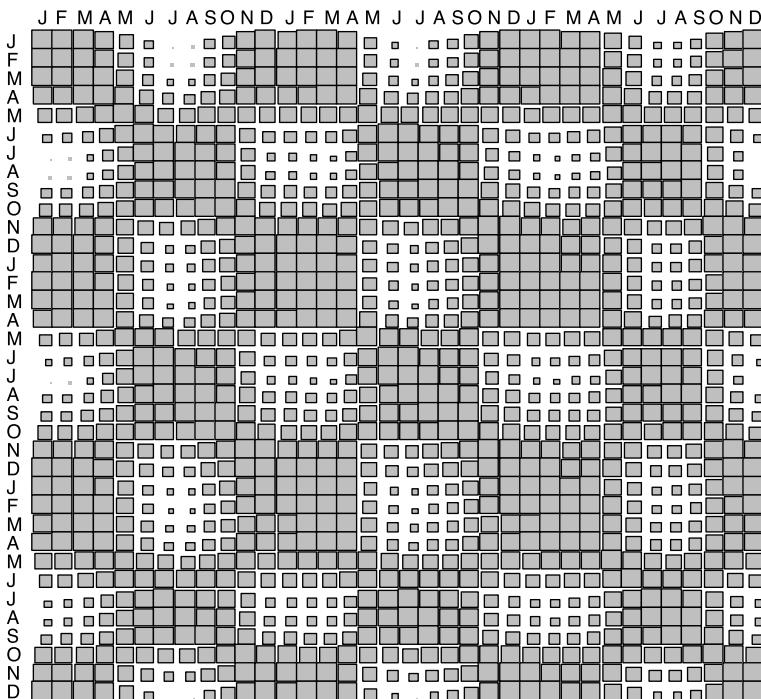


Figure 13.10. Fluctuation diagram for a correlation matrix. The correlations between monthly average temperatures at different locations are plotted for three years. Seasonal trends yield the dominant structure in the picture

Ash, Fat, Lactose, Protein, and Water. We use the percentages of the ingredients of the milks for each animal as weights in the mosaicplot. Figure 13.11 shows two weighted mosaics given in fluctuation mode for the Mammals Milk data. The cases are listed in alphabetic order. The left side shows the raw data; we can see that all of the milks mostly consist of water. DOLPHIN and SEAL show up as outliers, since they have the lowest percentages of Water, and highest percentages of Fat.

The different ingredients are allocated rather unevenly, so the use of standardized values here helps to detect more detailed patterns between each pair of mammals. The right side of Fig. 13.11 shows standardized values ($X \rightarrow X/\sigma_X$) used as weights. Again, it is obvious that DOLPHIN and SEAL behave similarly, but now we can also see that DEER and REINDEER also are very similar. Surprisingly, BISON and BUFFALO seem to be very different.

The use of clustering techniques like Complete Linkage yields results like those shown in Fig. 13.12. The rows of the data table are sorted according to the hierarchical clustering scheme. Now we can check the results from the clustering algorithm graphically by comparing the rows of the mosaicplot. This enables us to choose the number of clusters by graphical means, which, of course, is a soft and very subjective criterion, but one that enables us to respond quickly to different situations. In this

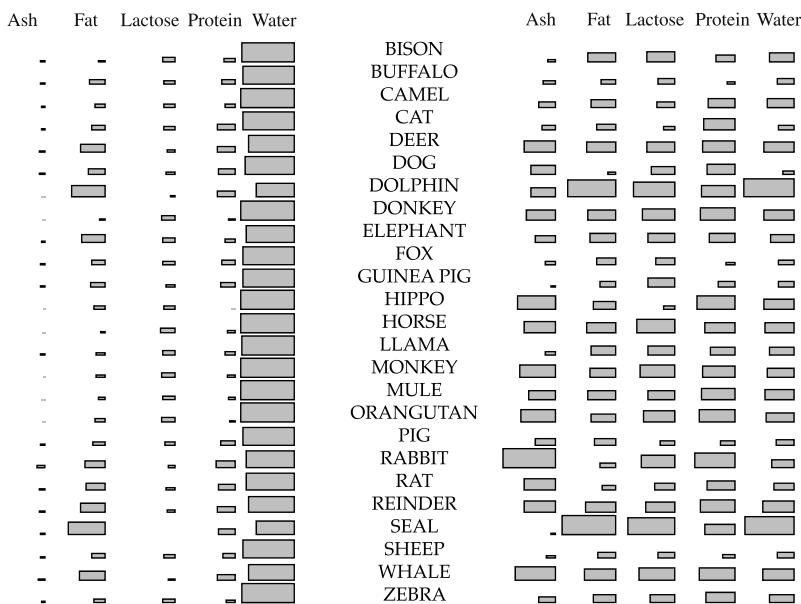


Figure 13.11. (Weighted) mosaicplots of the Mammals data. Raw data is shown on the *left*, while the values on the *right* are standardized

case, we'd probably decide to split the clusters a bit further, take the HIPPO out of cluster 1, split cluster 2 into CAMEL–LLAMA and BUFFALO–SHEEP, take the ELEPHANT out of cluster 2, and remove the RABBIT from cluster 4. The most questionable cluster is the remaining one, cluster 3, which contains the largest differences according to the graphic. The new clusters found graphically basically correspond to a vertical cut in the dendrogram with 11 clusters (see Fig. 13.12).

Of course, it is a lot easier to compare different rows if they are spatially close to each other. In the clustering algorithm, on the other hand, the order of observations in the same cluster is not relevant. Shifting whole clusters makes it possible to compare the distances between them.

The ability to reorder rows and columns quickly is crucial if we want to rapidly draw conclusions from graphical displays. Similarities between objects become far more obvious if these objects are close together. Therefore, ways of reordering and sorting rows and columns must be provided by the graphical user interface.

Bertin (1967, p. 36), proposes that the categories of a matrix be ordered “suitably” to visualise clusters in the data. Falguerolles et al. (1997) formalised Bertin's idea by introducing a purity function to measure the “simplicity” of a Bertin plot. Optimal orderings correspond to a minimum in the purity function. Depending on the data and the exact definition of the purity function, there are, of course, various ways in which the categories could be reordered. For row and column scores and a purity function which counts the number of pairs in “correct” order, reordering according to the purity function means sorting based on the scores.

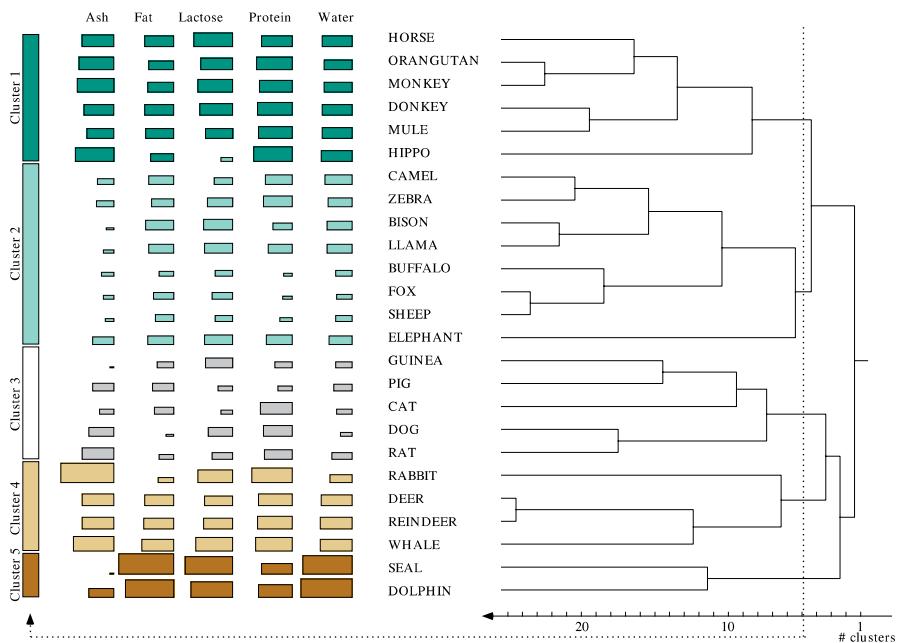


Figure 13.12. [This figure also appears in the color insert.] (Weighted) mosaicplot and dendrogram of the Mammals data clustered via complete linkage

13.3.3 Others

Multiple Bars

A multiple bar display is a variant of the mosaicplot that is very closely related to the fluctuation diagram. Starting from tiles of the same size, the tiles are shrunk to represent the cell's size. However, unlike fluctuation diagrams, the tiles in multiple bar displays are shrunk in one dimension only. This makes them look like barcharts. The structure in the mosaicplot determines the exact layout; switching the order of the variables (except for the last variable) will shift the smaller barcharts around, but will not change their size or appearance. Switching the last variable will realign single bars. In Fig. 13.13, the structure underlying the plot is gender, vehicles', age, i.e., the barcharts display the number of accident victims by age group. There is one barchart for each combination of vehicle type and gender. Overall, more men than women are reported to have accidents. The age distributions vary a lot between vehicle types, but are fairly similar between the two genders, with the exception of motorcycles. A lot more men than women are involved in motorcycle accidents.

Same-Bin-Size Displays

As the name suggests, the same-bin-size variant of a mosaicplot contains equally sized bins. This makes the same-bin-size display a mosaicplot with a special form of weight variable: W^i , the weight of observation i , is chosen to be proportional to

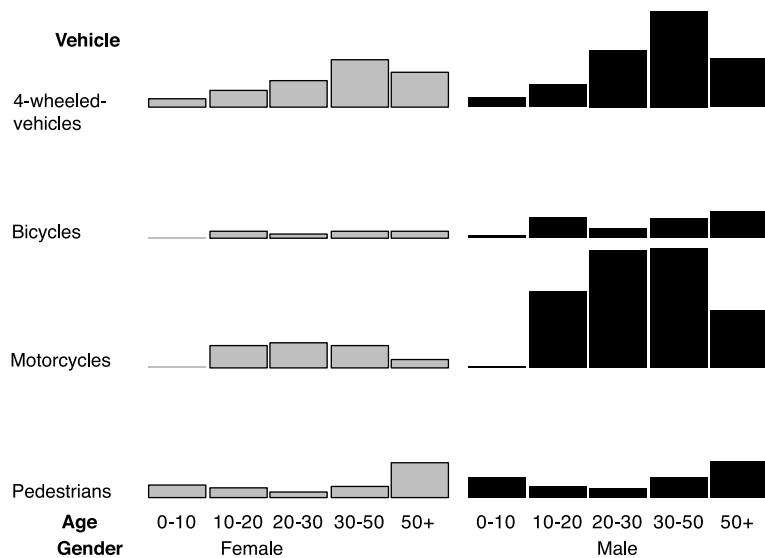


Figure 13.13. A multiple bars display of the Accident Victims data (Bertin, 1967, p. 30). More men than women are among the victims. The difference is particularly prominent for motorcycle accidents, which is the largest source of accidents for men. For women, the largest source of accidents is four-wheeled vehicles, except for girls under the age of ten and women above fifty. For both of these age groups, most of the accidents happen to pedestrians

the inverse of the number of cases in the cell into which observation i falls. Without any additional information, this plot is therefore a rather boring one. Where are the advantages of it?

There are two main areas where same-bin-size displays have proven to be very useful; comparisons of conditional distributions (visualized by highlighting heights), and searches for data patterns (e.g., the empty-bin pattern in sparse data cubes).

Figure 13.14 shows a default mosaicplot that is changed into its same-bin-size variant. This involves a change of vertical scale. This change in scale makes it much eas-

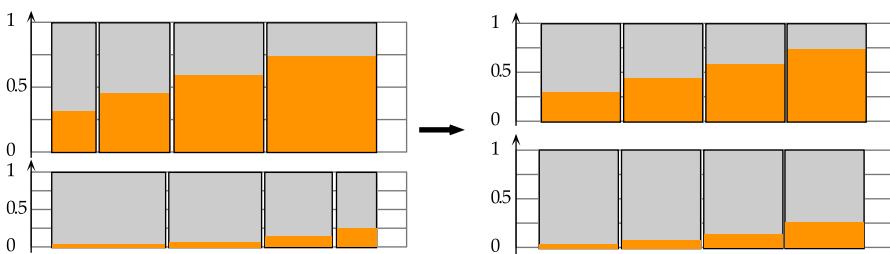


Figure 13.14. Changing from a default mosaicplot to a same-bin-size variant. The changes means that the highlighted heights of the upper and lower rows can be compared more easily, since the rows have identical vertical scales, instead of the different scales that they had in the mosaicplot

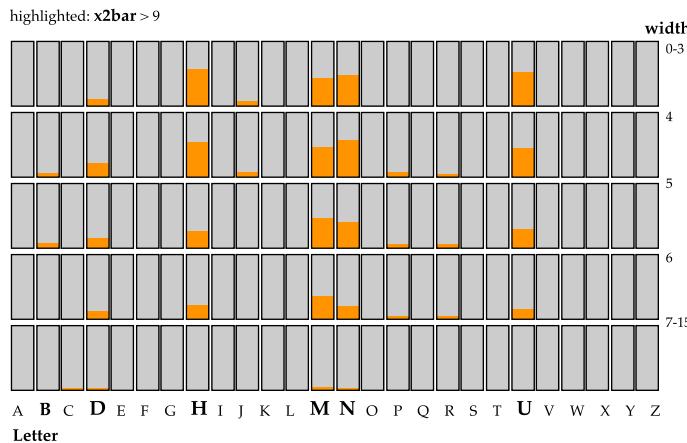


Figure 13.15. Two-dimensional mosaicplot of the Letter Recognition data. Due to the same-bin-size representation, the pattern for the highlighted cases is clearly visible

ier to compare the highlighted heights of the upper and lower rows; we only need to compare the positions on identical, although unaligned, scales in the same-bin-size display.

Figure 13.15 contains a mosaicplot of the Letters Recognition data (Frey and Slate, 1991). Cases with an $x\bar{2}bar$ value of at least 10 are highlighted. The mosaicplot shows all of the letters in the alphabet from left to right, classified according to five different classes of width. The top four rows all show the same pattern for the highlighted cases; the conditional probabilities are increased for the five letters D, H, M, N and U. B also shows increased proportions of highlighting in two of the rows.

The second application of the same-bin-size variation does not deal with additional information, but rather with missing information. If we have a sparse data cube, we can check for cells devoid of data. The goal now is to provide methods that provide a quick overview of the empty cells in the data set, and thus to deduce the empty-bin pattern within a dataset, which can help us to answer questions like:

1. How many empty bins are there?
2. Where do they occur?
3. Is there any recognizable pattern to their occurrence, or do they occur completely at random?

The order of variables is important, especially when looking for patterns and groups of combinations. Reordering the variables and collapsing over empty combinations can help.

Mondrian Displays

By default, mosaicplots are space-filling, since only thin spaces are present between the tiles. If these spaces are omitted we obtain another variant of a mosaicplot: the Mondrian display. While Mondrian displays are even more space-efficient than the

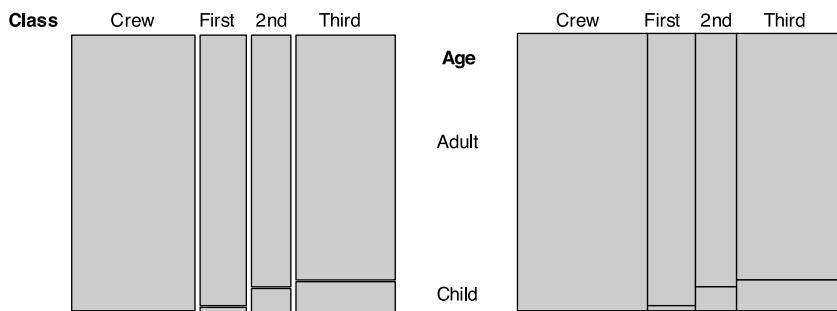


Figure 13.16. Regular (left) and Mondrian (right) mosaicplots of the Titanic data. Class is plotted versus Age. The empty crew/child cell disappears from the Mondrian display

standard mosaicplot, they are not well equipped to deal with empty combinations. Figure 13.16 shows two mosaicplots of the same aspect of the Titanic data: Class is plotted against Age. The empty bin (there were, by definition, no children among the crew members) disappears in the Mondrian display on the right. Sometimes, though, particularly in data with many categories, Mondrian displays can be beneficial since they utilize the tiny extra bit of screen space available (the spaces between the rectangles).

Related Work and Generalization

13.4

In contrast to the mosaicplot variants discussed earlier, this section is about actual extensions. While the variants show the same or a reduced amount (in the case of same-bin-size displays) of information, treemaps and trellis displays have a richer structure than a mosaicplot; indeed the mosaicplot is a special case of each of these.

Treemaps

13.4.1

Treemaps were introduced by (Johnson and Shneiderman, 1991) as a method of visualizing the file structure on a hard disk used by 14 users, in order to determine where and how space was used. Finding large files that could be deleted and determining which users consumed the most disk space were difficult tasks. Treemaps are designed to display trees, such as directory trees. Associated with each node in a directory tree is a numeric value that gives the disk space used by the files in the subtree rooted at the node. Each node is displayed as a rectangle; the size of this rectangle is proportional to the disk space it represents. All descendants of the node are displayed as rectangles inside this rectangle. By default, horizontal and vertical splits are alternated, just as in standard mosaicplots. Each file appears as a rectangle whose size is proportional to the file size, enabling users to spot large files at any level in the hierarchy.

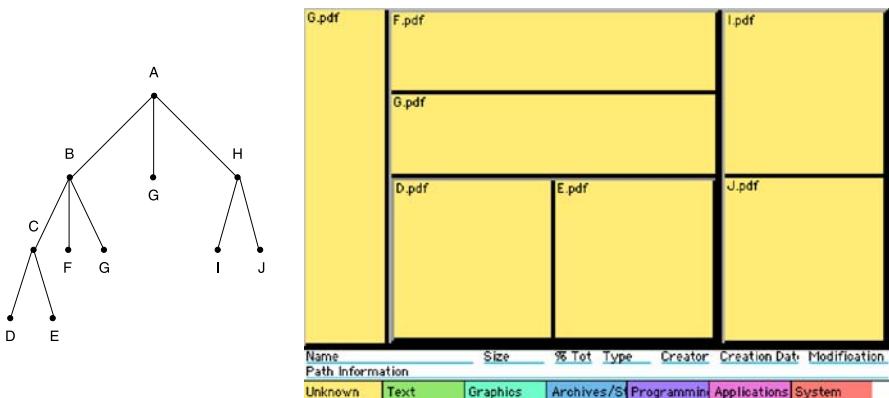


Figure 13.17. Tree (left) and accompanying treemap (right). In this example all nodes have equal weight. Only leaves of the tree are visible as tiles; the other nodes determine the structure

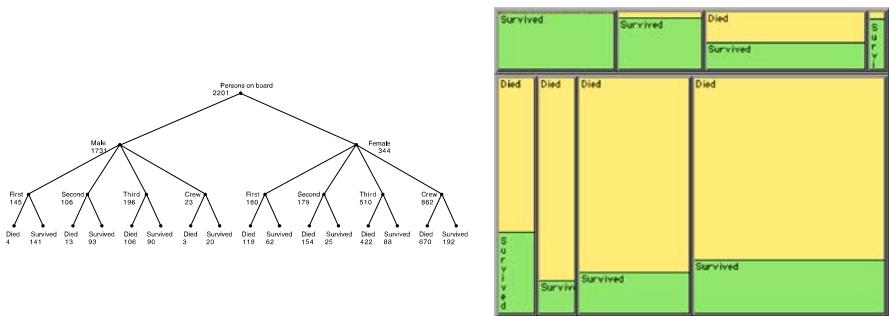


Figure 13.18. Tree (left) and accompanying treemap (right). The tree shows a contingency table for the Titanic data: the structure for the treemap on the right is gender', class, survived'

Treemaps and mosaicplots have a lot of features in common: both are area-based plots; both plots share a strict hierarchical construction; both strongly depend on the order of the variables. Treemaps have a richer structure than mosaicplots – any contingency table can be written in the form of a tree, but not vice versa (see Fig. 13.18 for example). When the tree shows a contingency table – all branches have the same depth and all nodes at a specific level are split along the same variables – a treemap simplifies to a mosaicplot. Both of the treemaps in Figs. 13.17 and 13.18 were created using the TreeViz software (Johnson and Shneiderman, 1991).

13.4.2

Trellis Plots

While a treemap represents an extension to the basic structure of a mosaicplot, a trellis plot (Becker et al., 1994) uses the same basic structure but enables more flexible displays. Trellis displays present the data in panels laid out in rows, columns and pages. In each panel of the trellis, a subset of the data is plotted using an appropriate display method, such as a barchart, a scatterplot, a dot plot, or a boxplot. Each

panel shows the relationship between certain variables given the values of other variables. The concept behind trellis displays is implemented in many data visualization systems. We are going to use the notation introduced by Deepayan Sarkar in his R package *lattice*. The structure of the plot that is produced is mostly controlled by the argument of the formula. The formula is generally of the form $y \sim x | g_1 * g_2 * \dots$, indicating that plots of y (on the y axis) versus x (on the x axis) should be produced, conditional on the variables g_1, g_2, \dots . If the conditioning variables are omitted, the trellis display consists of a single panel, showing the relationship between x and y . For low-dimensional plots, such as barcharts or dot plots, the y variable can be omitted too. A separate panel is produced for each unique combination of values of conditioning variables g_1, g_2, \dots using the points (x, y) for the subset of the data (also called the *packet*) defined by that combination. The order of the panels depends on the order in which the conditioning variables are specified, with g_1 varying fastest.

Figure 13.19 shows an example of a trellis display of three continuous variables – depth, latitude and longitude – representing the locations of earthquakes in the Tonga Trench. The location of each earthquake is plotted in a scatterplot, and the trellis display is organized according to earthquake depth. The following lines of R will produce the plot in Fig. 13.19.

```
> library(lattice)
> data(quakes)
> Depth <- equal.count(quakes$depth, number=8, overlap=.1)
> trellis.par.set(theme=col.whitebg())
> xyplot(lat ~ long | Depth, data=quakes, asp=1.0, pch=20)
```

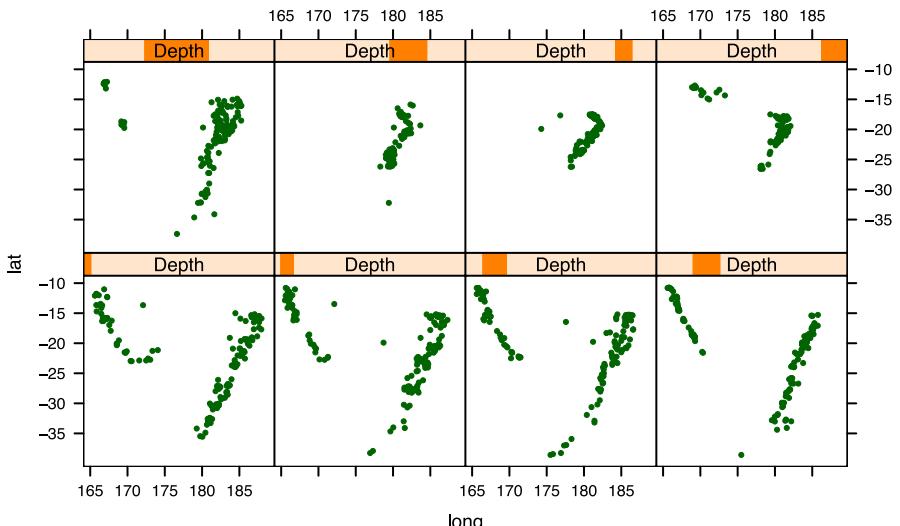


Figure 13.19. Scatter plots of locations of earthquakes in a trellis framework. The observations in each panel are organized according to the depth at which the earthquake happened

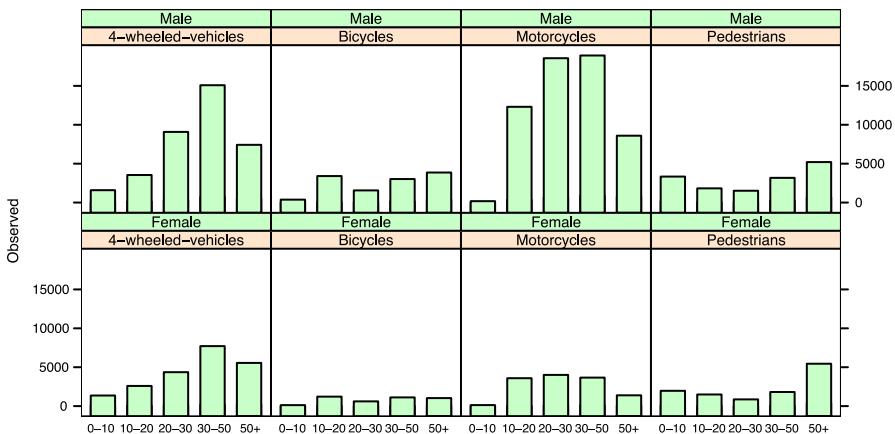


Figure 13.20. Trellis display of the Accident Victims data. The same variables are presented here as shown in Fig. 13.13

Trellis displays and mosaicplots are very closely related from a conceptual point of view. The multiple barchart variant of the mosaicplot is essentially a trellis display of only categorical variables. The close relationship between mosaicplots and trellis displays is probably most obvious for the example of the accident victim data shown in Figs. 13.13 and 13.20, where Fig. 13.20 is produced by the following lines of code:

```
> library(lattice)
> barchart(Observed~Age|Vehicle*Gender, data=acc)
```

The formula notation of R also fits with our notation of the plot's structure: the formula $y \sim x | g_1 * g_2 * \dots * g_p$ corresponds to a mosaicplot with a structure of g_p, \dots, g_1, x, y , i.e., the order of the variables is inverted. Trellises are not very strict about the actual layout of the panels, but a similar approach to the default layout of mosaicplots – alternating between rows and columns, i.e., horizontal and vertical splits – is generally used for trellis displays too.

The big difference between trellis displays and mosaics is, of course, the ability of trellis displays to flexibly display observations using different display types, whereas mosaics are restricted to counting the number of observations in each combination of the conditional variables and then displaying this number as a proportionally sized rectangle.

13.4.3 Pivot Tables

While pivot tables are not primarily a graphical technique, they are still closely related to mosaicplots and so are discussed here. Pivot tables are included in any Excel distribution and are therefore widely available, although not widely used. Pivot tables allow us to summarize categorical variables. Figure 13.21 shows Excel's pivot table assistant: using drag and drop from a list of all variables, variables can be placed in the

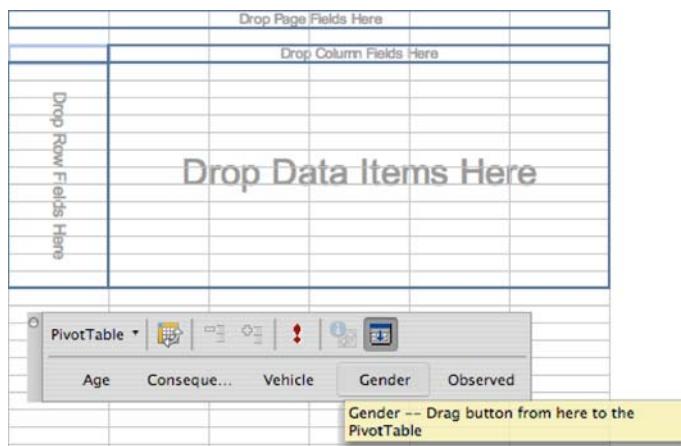


Figure 13.21. Excel's assistant for setting up a pivot table using drag and drop from a list of variables

rows and columns of a data table. After the variables of the frame have been set up, the data variable is picked and placed in the middle of the data table.

The values of the data variable are summarized according to the combination of the variables of the frame. Excel provides various options for summarizing (see Fig. 13.22): counting observations, summarizing their values, picking minima, maxima or averages, and so on.

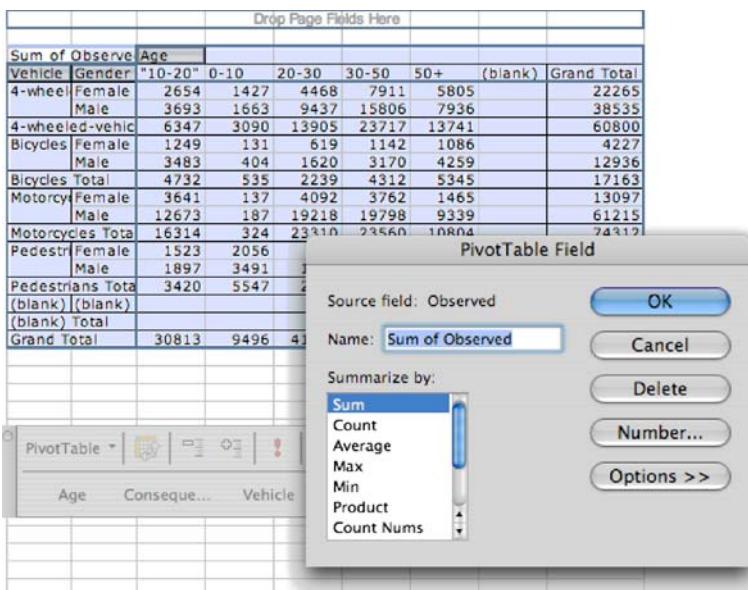


Figure 13.22. Pivot table for the Accident Victims data. *Front:* dialog window showing the summarizing options available for in Excel's pivot tables

Therefore, pivot tables have the same structure as mosaicplots, but allow different summaries from mosaicplots. A pivot table's structure carries slightly less information than a mosaicplot, as the order of the variables in rows and columns is stored separately. A structure of the form $X_1, X'_2, X_3, X', \dots$ is stored in two separate lists: a list of row variables X_1, X_3, \dots and a list of column variables X_2, X_4, \dots (together with the order of the variables in each list). This means that a pivot table is unable to distinguish between the structures X_1, X'_2, X_3 or X_1, X_3, X'_2 or X'_2, X_1, X_3 . On top of the pure cross-tabulation, pivot tables usually provide marginal information: row sums and column sums are given, as well as totals by variable and grand totals. This information is incorporated into a mosaic implicitly because of the structure, but is not readily available; since mosaics are constructed hierarchically, a low-dimensional mosaicplot is “included” in a high-dimensional mosaicplot. Onlookers must visually “remove” the last splits performed and recombine tiles to see the low-dimensional mosaic – which is only feasible in the standard form of mosaicplot. This corresponds to reading Fig. 13.1 backwards from left to right. The use of different ways of summarizing the observations corresponding to the combinations of conditional variables may lead to new variants of mosaicplot. Each new summary will mean a change in the way that the mosaics must be interpreted, as the tile area does not then correspond to the probability estimate for the joint distribution of the variables.

13.5

Implementations

Implementations of mosaicplots become more and more frequent – almost all software packages now have some functionality for creating mosaicplots. Perhaps the most notable of them is the implementation of mosaicplots in R by John Emerson (1998), as well as the fully interactive implementations in Manet (Unwin et al., 1997), Mondrian (Theus, 2002), and KLIMT (Urbanek, 2002).

Treemaps were first introduced in the Macintosh software TreeViz (Johnson and Shneiderman, 1991). They are now more widely available in a variety of forms; for example, fully interactive and linked treemaps can be found in the software package KLIMT, the “Map of the Market” (Wattenberg, 1998) shows a squarified treemap of stocks available at <http://www.smartmoney.com>, and the Java software TreeMap (Bederson et al., 2002) provides a very much improved treemap version of TreeViz.

Implementations of pivot tables are widespread, although only in disguise. The name “pivot table” seems to be used exclusively in Excel; other names for them include cross-tabulations and (multivariate) data summaries. There are various functions in R that allow cross-tabulations in specific situations, such as *table*, *xtabs*, *ftable*, A unified framework for cross-tabulations is provided by Hadley Wickham in his *recast* package (Wickham, 2006).

Both mosaicplots and their extensions can benefit from each other; by accommodating some of the features of the extensions, mosaics should become more applicable and flexible. The extensions, on the other hand, could possibly make use of the mosaicplot's variants.

References

- Agresti, A. (1990). *Categorical Data Analysis.*, New York: Wiley.
- Becker, R.A., Cleveland, W.S. and Shyu, M. (1994). Trellis Displays: Questions and Answers. Research report no 9/94. Murray Hill, NJ: AT&T Bell Laboratories.
- Bederson, B.B., Shneiderman, B. and Wattenberg, M. (2002). Ordered and Quantum Treemaps: Making Effective Use of 2D Space to Display Hierarchies. *ACM Transactions on Graphics (TOG)*, 21:833–854.
- Bertin, J. (1967). *Semiologie Graphique*. Paris: Editions Gauthier-Villars.
- Bhapkar, V. and Koch, G. (1968). Hypotheses of ‘No interaction’ In Multidimensional Contingency Tables. *Technometrics*, 10:107–123.
- Dawson, R.J.M. (1995). The “unusual episode” data revisited. *Journal of Statistics Education*, 3.
- Emerson, J.W. (1998). Mosaic displays in S-PLUS: a general implementation and a case study. *Statistical Computing and Graphics Newsletter*, 9:17–23.
- Falguerolles, A., Friedrich, F. and Sawitzky, G. (1997). A Tribute to J. Bertin’s Graphical Data Analysis. In: Bandilla, W. and Faulbaum, F. (eds) *Advances in Statistical Software 6*. Stuttgart: Lucius and Lucius, pp. 11–20.
- Frey, P.W. and Slate, D.J. (1991). Letter Recognition Using Holland-Style Adaptive Classifiers. *Machine Learning*, 6:161–182.
- Friendly, M. (1992). Mosaic displays for loglinear models. in *Proceedings of the statistical graphics section*, ASA, pp. 61–68.
- Friendly, M. (1994). Mosaic Displays for Multi-Way Contingency Tables, *Journal of the American Statistical Association*, 89:190–200.
- Friendly, M. (1995). Conceptual and visual models for categorical data. *Amer. Statistician*, 49:153–160.
- Friendly, M. (1999). Extending Mosaic Displays: Marginal, Conditional and Partial Views of Categorical Data. *Journal of Computational and Graphical Statistics*.
- Gentleman, R. and Ihaka, R. (1995). The R Home Page. <http://www.stat.auckland.ac.nz/rproj.html>.
- Hartigan, J.A. (1975). *Clustering algorithms*. Wiley Series in probability and mathematical statistics. New York: Wiley.
- Hartigan, J.A. and Kleiner, B. (1981). Mosaics for Contingency Tables. In *13th Symposium on the Interface*, New York: Springer, pp. 268–273.
- Hartigan, J.A. and Kleiner, B. (1984). A mosaic of television ratings. *American Statistician*, 38:32–35.
- Hofmann, H. (2000). Exploring categorical data: interactive mosaic plots. *Metrika*, 51:11–26.
- Hofmann, H. (2001). Generalized Odds Ratios for Visual Modelling. *Journal of Computational and Graphical Statistics*, 10:1–13.
- Hofmann, H. (2003). Constructing and Reading Mosaicplots. *Computational Statistics and Data Analysis*, 43:565–580.
- Hofmann, H., Siebes, A. and Wilhelm, A.F. (2000). Visualizing association rules with interactive mosaic plots. In *Proc. of the 6th Int'l conf. on Knowledge Discovery and data mining*, ACM-SIGKDD, Boston, MA, pp. 227–235.

- Johnson, B. and Shneiderman, B. (1991). Treemaps: a space-filling approach to the visualization of hierarchical information structures. In *Proceedings of the 2nd International IEEE Visualization Conference*, pp. 284–291.
- Sewell, W. and Shah, V. (1968). Social class, parental encouragement and educational aspirations. *American Journal of Sociology*, 73:559–572.
- Shneiderman, B. (1992). Tree Visualization with Tree-Maps: A 2-D SpaceFilling Approach. *ACM Transactions on Graphics*, 11:92–99.
- Theus, M. (2002). Interactive Data Visualization using Mondrian. *Journal of Statistical Software*, 7.
- Theus, M. and Lauer, S. (1999). Visualizing Loglinear Models. *Journal of Computational and Graphical Statistics*, 3:396–412.
- Unwin, A.R., Hawkins, G., Hofmann, H. and Siegl, B. (1997). MANET – Extensions to Interactive Statistical Graphics for Missing Values. In *New Techniques and Technologies for Statistics II*, Amsterdam: IOS Press, pp. 247–259.
- Urbanek, S. (2002). Different ways to see a tree - KLIMT. In *14th Conference on Computational Statistics, COMPSTAT*, Heidelberg: Physica, pp. 303–308.
- Wattenberg, M. (1998). Map of the Market, <http://www.smart-money.com/market-map/>.
- Wickham, H. (2006). *recast* – an R package, <http://cran.r-project.org/>.
- Wilkinson, L. (1999). *The Grammar of Graphics*., New York, Springer.

Parallel Coordinates: Visualization, Exploration and Classification of High-Dimensional Data

III.14

Alfred Inselberg

14.1	<i>Introduction</i>	644
	Origins	644
	The Case for Visualization	646
14.2	<i>Exploratory Data Analysis with \parallel-coords</i>	648
	Multidimensional Detective	648
	An Easy Case Study: GIS Data	649
	Compound Queries: Financial Data	655
	Hundreds of Variables	663
14.3	<i>Classification</i>	664
14.4	<i>Visual and Computational Models</i>	668
14.5	<i>Parallel Coordinates: Quick Overview</i>	671
	Lines	671
	Planes and Hyperplanes	672
	Nonlinear Multivariate Relations: Hypersurfaces	674
14.6	<i>Future</i>	676

A dataset with M items has 2^M subsets, any one of which may be the one we really want. With a good data display, our own fantastic pattern-recognition abilities can not only sort through this combinatorial explosion, but they can also extract insights from the visual patterns. These are the core reasons for data visualization. With parallel coordinates (abbrev. \parallel -coords), the search for multivariate relations in high-dimensional datasets is transformed into a 2-D pattern recognition problem. In this chapter, the guidelines and strategy for knowledge discovery using parallel coordinates are illustrated on various real datasets, one with 400 variables from a manufacturing process. A geometric classification algorithm based on \parallel -coords is presented and applied to complex datasets. It has low computational complexity, providing the classification rule explicitly and *visually*. The minimal set of variables required to state the rule are found and ordered by their predictive value. A visual economic model of a real country is constructed and analyzed to illustrate how multivariate relations can be modeled using hypersurfaces. The overview at the end provides a basic summary of \parallel -coords and a prelude of what is on the way: the distillation of relational information into patterns that eliminate need for polygonal lines altogether.

14.1

Introduction

14.1.1

Origins

Over half of our sensory neurons are devoted to vision, endowing us with tremendous pattern recognition abilities. The goal of visualization is to couple this talent to our problem-solving capabilities, in order to obtain insights from images. In order to make it easier to visualize multivariate/multidimensional problems, numerous mappings that visually encode multidimensional information in a two-dimensional plane (see Friendly, 2005; Tufte, 1983, 1990, 1996), have been invented. Such mappings augment our perception, which is limited by our habitation of a three-dimensional world. Great successes in this field, like Minard's "Napoleon's March to Moscow," Snow's "dot map," and others are, however, ad hoc (i.e., one-of-a-kind) and exceptional. Succinct multivariate relations are rarely made apparent by static displays. To overcome this problem, we must incorporate the interactivity enabled by the technological advances of the last half-century. In turn, this raises the issues of how to create effective GUIs (graphical user interfaces), queries, exploration strategies, and of course good information-preserving displays, but we are getting ahead of ourselves.

Legend has it that Archimedes, while constructing a proof, was so absorbed in a diagram when he was killed by a Roman soldier that he pleaded "Do not disturb my circles" as he was being struck by a sword. Visualization first flourished in geometry (which makes Archimedes' the first recorded death in defense of visualization). The utilization of diagrams is strongly interwoven within the testing of conjectures and the construction of proofs. This essence of visualization was eventually abstracted

and adapted into the more general problem-solving process. We form a *mental image* of the problem we are trying to solve, and at times we say that we *see* when we mean that we understand.

My interest in visualization was also sparked and nourished while learning geometry. Later, while studying multidimensional geometry I became frustrated by the absence of visualization. I wondered why geometry was being tackled without using (the fun and benefits of) pictures? Was there a generic way of creating accurate pictures of multidimensional problems, like Descartes' epoch-making coordinate system? What is "sacred" about orthogonal axes, which quickly "use up" the plane associated with them? After all, parallelism rather than orthogonality is the fundamental concept in geometry, and these are **not** equivalent terms, for orthogonality requires a prior concept of "angle." I played with the idea of a multidimensional coordinate system based on parallel coordinates, in which, in principle, lots of axes could be placed and viewed. Encouraged in 1959 by Professors S.S. Cairns and D. Bourgin, both topologists at the University of Illinois where I was studying, I derived the basic $\text{point} \leftrightarrow \text{line}$ duality followed by the $N - 1 \text{ points} \leftrightarrow N\text{-dimensional line}$ correspondence, realizing in the process that *projective* rather than Euclidean space is involved.

In 1977, while teaching linear algebra to a large class of unruly engineers, I was challenged to *show* spaces with more than three dimensions. I recalled parallel coordinates (abbreviated \parallel -coords), and the question of how multidimensional lines, planes and other objects "look" was raised. This triggered the systematic development of \parallel -coords, and a comprehensive report Inselberg (1981) documented the initial results. Noted on the first page is the superficial resemblance to nomography, where the term "parallel coordinates" was used (Brodetzky, 1949). In nomography, which declined with the advent of computers, there are graphical computational techniques involving "functional scales" (Otto, 1963, p. 66), often placed in parallel, for problems with usually two or three variables. Until very recently, I was not aware (I am indebted to M. Friendly for pointing this out in (Friendly, 2005)) of d'Ocagne's marvellous monograph (d'Ocagne, 1885) on parallel coordinates for two dimensions, where a $\text{point} \leftrightarrow \text{line}$ duality was studied and, together with a $\text{point-curve} \leftrightarrow \text{line-curve}$ correspondence, was applied to interesting numerical problems. D'Ocagne pursued computational applications of nomography (d'Ocagne, 1899) rather than developing a *multidimensional system of parallel coordinates* – a systematic body of knowledge consisting of theorems on the representation of multidimensional objects, their transformations and geometrical construction algorithms (i.e., for intersections, interior points, etc.), which is where we came in.

This is a good time to recoup and trace the development of \parallel -coords. In 1980, it was first presented at an international conference (Inselberg, 1980), and prior to that at seminars like that at the University of Maryland, where I was fortunate to meet Ben Shneiderman, whose encouragement and visualization wisdom I have benefited from ever since. A century after the publication of d'Ocagne's monograph, Inselberg (1985) (coincidentally) appeared. This period of research – performed in collaboration with my esteemed colleagues the late B. Dimsdale (a long-time associate of John von Neumann), A. Hurwitz, who was invaluable in every aspect leading to three US patents (Collision Avoidance Algorithms for Air-Traffic Control), Rivero and Insel-

berg (1984), together with students T. Chomut (Chomut, 1987) (who implemented the first \parallel -coords EDA¹ software), and the ubiquitous M. Boz (Inselberg et al., 1991) – was very fruitful, as indicated by the partial list (Inselberg, 1984, 1985, 1989; Inselberg et al., 1987) (applications to statistics) and Inselberg and Dimsdale (1990). It paved the way for new contributors and users: R.P. Burton's group at Brigham Young University (since 1983) Cluff et al. (1991), Cohan and Yang (1986), Hinterberger (first studied data densities using \parallel -coords, 1987) Schmid and Hinterberger (1994), Helly (1987), Fiorini and Inselberg (1989), Gennings et al., contributed a sophisticated statistical application (Gennings et al., 1990)(response surfaces based on \parallel -coords), Wegman (greatly promoted EDA applications) (Wegman, 1990), and Desai and Walters (1991). The results obtained by Eickemeyer (1992), Hung and Inselberg (1992) and Chatterjee (1995) were seminal. Progress continued through Chatterjee et al. (1993), Ward (1994), Jones (1996), to the most recent work of Yang (2005) and Hauser (2005), which increased the versatility of \parallel -coords. At the current time of writing, a query for “parallel coordinates” on Google returned more than 60,000 “hits.”

14.1.2 The Case for Visualization

Searching a dataset with M items for “interesting” (a term that depends on the objectives) properties is inherently a difficult task. There are 2^M possible subsets, any one of which could most closely satisfy the objectives. The visual cues that our eyes can pick out from a decent data display can enable us to quickly sort through this combinatoric explosion. How this is done is only part of the story here. Clearly, if the transformation *data* \rightarrow *picture* loses information, a great deal is lost right at the start. When *exploring* a dataset with N variables rather than *presenting* it (as pie charts, histograms, etc.), use a good display to:

1. **preserve information** – it should be possible to reconstruct the dataset completely from the picture,
2. **have low representational complexity** – the computational cost of constructing the display should be low,
3. **work for any N** – it should not be limited by the dimensions of the data,
4. **treat each variable uniformly**,
5. **exhibit invariance under projective transformations** – the dataset should be recognizable after rotations, translations, scalings and perspective transformations,
6. **reveal multivariate relations in the dataset** – this is the most important and controversial single criterion,
7. **be based on a rigorous mathematical and algorithmic methodology** – thus eliminating ambiguity in the results.

Neither completeness nor uniqueness is claimed for this list, which should invite criticism and changes. Further commentary on each item, illustrated by examples and comparisons, is listed below in the same order.

¹ abbreviation of exploratory data analysis

1. The numerical values of each N -tuple (i.e., each data point) should be recoverable from the scatterplot matrix (abbr. SM) and the \parallel -coords display. By contrast, this may not be necessary or desirable for presentation graphics.
2. In the pairwise scatterplot matrix, the N variables appear $N(N - 1)/2$ times. By contrast there are only N axes in \parallel -coords, although there is an additional preprocessing cost, as pointed out later. For $N \geq 10$, the practical barriers due to the required display space and visual difficulties limit the use of a SM. These barriers are less stringent for \parallel -coords.
3. Even when the perspective is used effectively, orthogonal coordinates are inherently limited to $N = 3$ due to the dimensionality of our existence. With some “trickery,” the illusion of a few more dimensions can be added. For \parallel -coords, implementation capabilities rather than conceptual barriers determine the maximum feasible N .
4. In the “Chernoff faces” display, for example, each variable corresponds to a specific facial feature and is treated accordingly. The correspondence *facial feature* \rightarrow *variable* is arbitrary. Choosing a different correspondence gives a different display. The fun is that there is no general way to show that the two different displays portray the same dataset. Of course, this is also true for general “glyph” displays.
5. This is true for SM and for \parallel -coords, although it has not been implemented in general. Incidentally, this is a wonderful M.Sc. thesis topic!
6. The real value of visualization, in my opinion, is not the ability to see “zillions of objects,” but to recognize **relations** among them. We know that projections lose information, which can possibly be recovered using interactivity. Nevertheless, important clues that can *guide* the interaction are lost. So I prefer to start with a display where all of the information is there even though it may be tricky to uncover. What visual cues are available and how they guide the exploration are crucial determining factors.
7. The value of rigor is self-evident.

These and additional issues comprising the discovery process are better appreciated via the exploration of three real datasets. The basic queries are introduced in the first example from GIS and are subsequently combined, with boolean operators, for use on financial data. An example with 400 variables is briefly discussed before moving on to automatic classification. Visualization and \parallel -coords play key roles in the algorithm’s conception, internal functions and the visual presentation of the classification rule. The minimal basis of the variables is found and ordered according to their predictive value.

The overview at the end provides background on optimizing the use of \parallel -coords and its applications. This involves:

1. learning *patterns* corresponding to the basic relations and seeking them out for EDA,
2. understanding the design and use of *queries*,
3. the relevance of \parallel -coords to other sophisticated statistical applications like *response surfaces* (Gennings et al., 1990), and
4. applications to *regression*, as in the example at the end,

5. understanding that the relational information resides in the *crossings*,
6. **concentrating the relational information in the data into clear patterns that eliminate the need for polygonal lines altogether.** This last point is illustrated by the “proximate planes” example, which encourages research into efficient (parallel) algorithms for accomplishing this on general datasets. We eliminate the “clutter” by reducing the display into patterns corresponding, at least approximately, to the multivariate relations in the data.

Before we dive into the specifics of \parallel -coords, we provide a visualization challenge for the reader to ponder. For a plane

$$\pi : c_1x_1 + c_2x_2 + c_3x_3 = c_0 , \quad (14.1)$$

allow the coefficients to each vary within a small interval. This generates a family of “close” (let’s call them *proximate*) planes:

$$\Pi = \{ \pi : c_1x_1 + c_2x_2 + c_3x_3 = c_0, \quad c_i \in [c_i^-, c_i^+], \quad i = 0, 1, 2, 3 \} . \quad (14.2)$$

These are the planes generated by small rotations and translations of π with respect to the three coordinate axes. Together they form a “twisted slab” which, even in 3-D using orthogonal axes, is difficult to visualize. Conversely, given lots of points in 3-D, how can we show (using **any** general visual method you can think of) that they lie on a twisted slab, and how can we visualize such a form precisely; for $N = 3$ and then for *any N*? In the meantime, you can project, pursue, scale, reduce, scintillate, regress, corollate or tribulate to your heart’s content, but please do not peek at the answer, which is given at the end of the chapter.

14.2

Exploratory Data Analysis with \parallel -coords

14.2.1

Multidimensional Detective

Parallel coordinates transform multivariate relations into 2-D patterns suitable for exploration and analysis. For this reason they are included in lots of software tools. The queries “Parallel Coordinates + Software” on Google returned about 31,000 “hits” and “Scatterplot matrix + Software” about 15,000 at the time of writing. Irrespective of the apparent relative ratio, the fact that the numbers are comparable astounded me, having heard the appellations “esoteric,” “unnatural,” “difficult,” “squiggly,” and others used in association with \parallel -coords.

The exploration (the venerable name “exploratory data analysis,” EDA, is used interchangeably with the currently more fashionable “visual data mining”) paradigm is that of a detective, starting from the data, searching for clues leading to conjectures, testing, backtracking until *voila* ... the “culprit” is discovered. The task is especially intricate when many variables (i.e., dimensions) are involved, which calls for the employment of a *multidimensional* detective (abbrev. MD). As if there were

any doubts, our display of choice is \parallel -coords, where the data appear in the form of squiggly blotches, and which (using *queries*) the MD skilfully dissects in order to find precious hidden secrets.

At this point, it is worth considering how similar queries are performed using other exploration methodologies, including the ubiquitous spreadsheets. More importantly, which visual clues prompt the use of such queries. A few basics (see Sect. 14.5 or the more detailed references Inselberg, 1999, 2008) are recalled here. In \parallel -coords, due to the *point* \leftrightarrow *line* and other dualities, some but not all actions are best performed in the dual. The queries, which are the “cutting tools,” operate on the display (i.e., dual). The design of these queries should exploit the methodology’s strengths and avoid its weaknesses, rather than mimic the action of queries operating on standard “nondual” displays. Just like a surgeon has many specialized cutting tools, one of our early software versions had lots of specialized queries. However, not only was it difficult to classify and remember them all, but they still could not handle all of the situations encountered. After experimentation, I opted for a few (three) intuitive queries, called **atomic**, which can be combined via boolean operations to form complex intricate cuts. Even for relatively small datasets, the \parallel -coords display can look uninformative and intimidating. A lack of understanding of the basics of the underlying geometry and poor query choice can limit the use of \parallel -coords to unrealistically small datasets.

Summarizing, the requirements for successful exploratory data analysis are:

- an informative display that does not lose any of the information about the data,
- good query selection,
- skilful interaction with the display.

An Easy Case Study: GIS Data

14.2.2

The first admonition is:

- **do not let the picture intimidate you,**

which can easily happen if you take an uninformed look at Fig. 14.2, which is the dataset that we are going to explore. It consists of over 9,000 measurements with nine variables; the first two, (X, Y) , specify the location on the map of the portion of Slovenia shown in Fig. 14.1 (left) where seven types of ground emissions have been measured by satellite. The ground location, (X, Y) , of one data item is shown in Fig. 14.1 (right), which corresponds to the region shown in the map and remains open during the exploration. The query used to select the data item (as shown in Fig. 14.2) is called *Pinch*. It is activated by the button **P** on the toolbar. Using this query, a bunch of polygonal lines (i.e., data items) can be selected (they are “pinched” between the axes). Moving the cursor changes the position of the selected arrowhead, which is the larger of the two shown. In due course various parts of the GUI will be illustrated (*Parallax*, proprietary software from MDG Ltd., All Rights Reserved, is used with permission).

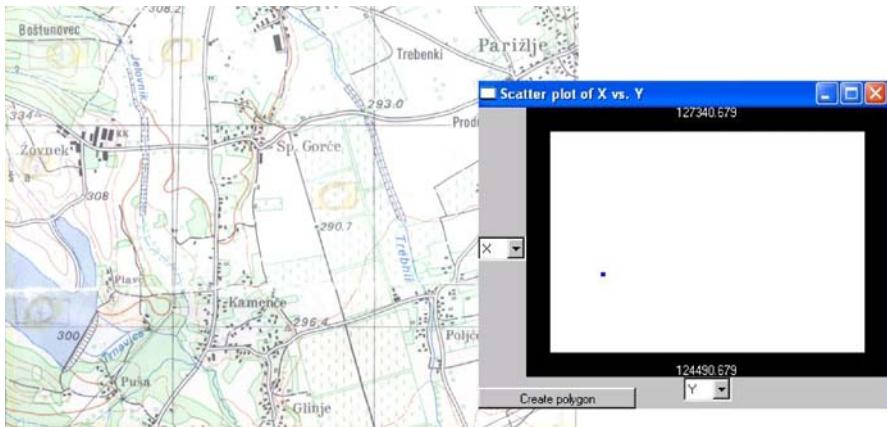


Figure 14.1. (Left) Region of Slovenia where seven types of ground emissions were measured by the LandSat Thematic Mapper, as shown in subsequent figures (thanks to Dr. Ana Tretjak and Dr. Niko Schlamberger, Statistics Office of Slovenia). (Right) The display is the map's rectangular region, while the dot marks the position where the seven-tuple shown in the next figure was measured

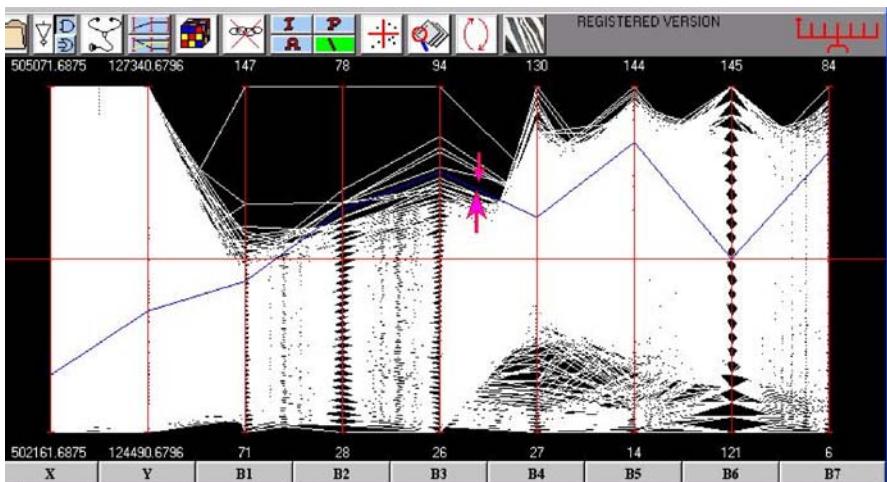


Figure 14.2. A query in Parallax showing a single data item: the X, Y position (as shown on the right of Fig. 14.1) and the values of the seven-tuple ($B_1, B_2, B_3, B_4, B_5, B_6, B_7$) at that point

Aside from starting the exploration without any bias, it is essential that you
— understand the objectives.

Here the task is to detect and locate various ground features (i.e., built-up areas, vegetation, water, etc.) on the map. There is a prominent lake in the lower-left corner that has an unusual shape with an upward pointing “finger.” This brings us to the next admonition: no matter how messy it looks,

— carefully scrutinize the data display for clues and patterns.

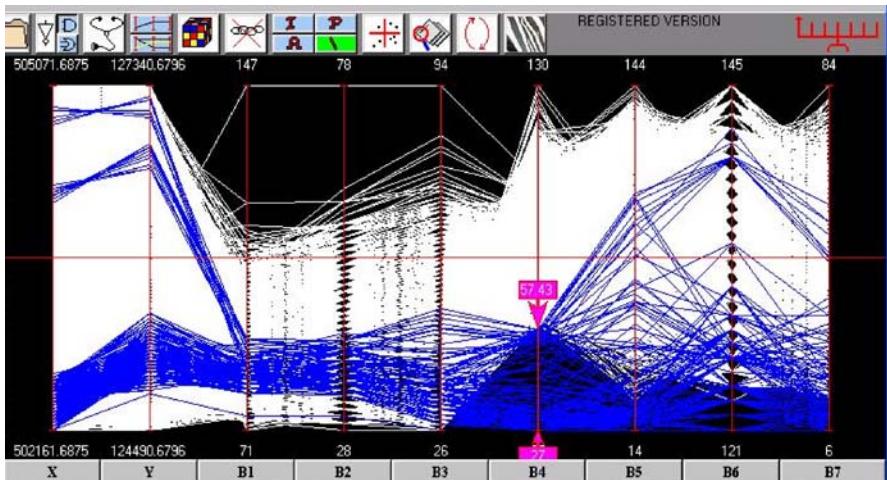


Figure 14.3. Finding water regions. The contrast due to density differences around low values of B_4 is the *visual cue* that prompts this query

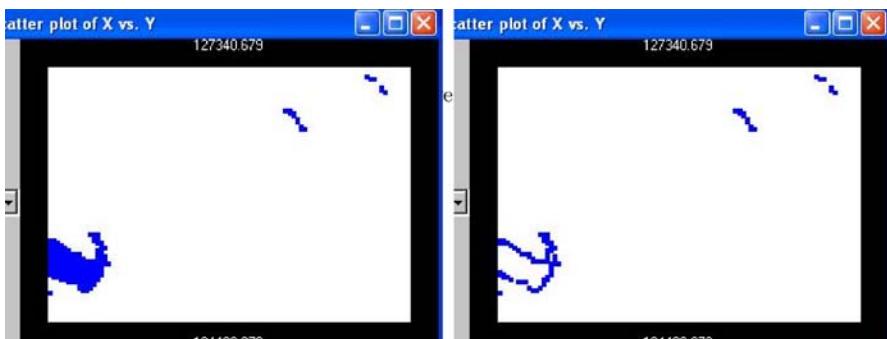


Figure 14.4. (Left) The lake (the result of the query shown in Fig. 14.3) and (right) just its boundary (the result of the query shown in Fig. 14.5)

Follow up on anything that catches the eye: gaps, regularities, holes, twists, peaks and valleys, density contrasts (like the one at low values of B_3 through B_7). Using the *Interval* query, activated by the I button, starting at the minimum, we grab the low range of B_4 (between the arrowheads), stopping at the dense part as shown in Fig. 14.3. The result, shown on the left of Fig. 14.4, is amazing. We have found the water – the lake is clearly visible, together with two other regions which turn out to be small streams upon comparing with the map. With our scrutiny rewarded, we recall the adage that

— a good thing may be worth repeating.

Searching for density variations within the lower interval of B_4 , we note that the lowest part is much denser than the rest. Experimenting a bit, and appreciating the

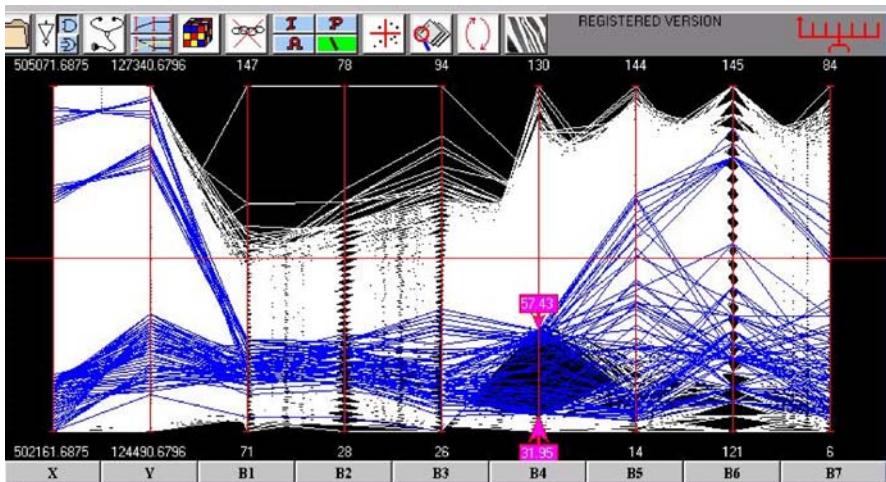


Figure 14.5. Query finding the water's edge

importance of interactivity, we select the sparse portion, Fig. 14.5, which defines the water's edge (Fig. 14.4, right), and in fact more. By dropping the lower arrow, we see the lake filling up starting from the edge (i.e., shallow water first). Therefore, the lower values of $B4$ reveal the water and the lowest “measure” shows the water's depth; not bad for a few minutes of playing around.

But all this pertains to a single variable, when we are supposed to be demonstrating *multivariate* exploration. This is a valid point, but we did *pick* $B4$ among several variables. Further, this is a useful “warm-up” for the more involved example that we encounter next, since it has enabled us to present two of the queries. The astute observer has probably already noticed the regularity, the vertical bands, between the $B1$, $B2$ and $B3$ axes. This is where the *angle* query, activated by the **A** button, comes into play. As the name implies, it selects groups of lines within a user-specified angle range. A data subset which corresponds to regions on the map with high vegetation is selected between the $B2$ and $B3$ axes as shown in Fig. 14.6 (left); the inter-axis distance is increased to better illustrate the vertical bands. Clicking the **A** button and placing the cursor at the middle of one axis opens an angle with its vertex in the mid-range of the previous (left) axis. The range of this angle is controlled by the arrow movements on the right axis. Actually, this “rule” (i.e., relation between some parameters) for finding vegetation can be refined by tweaking a couple more parameters. This raises the topic of rule-finding in general – *classification* – which is taken up in Sect. 14.3.

The *angle* and *pinch* queries are motivated by the ℓ line \rightarrow point $\bar{\ell}$ correspondence in $\|\text{-coords}$ illustrated in Fig. 14.7. To clarify the notation, m is the slope of the line ℓ and b is the intercept (i.e., the line's intersection with the x_2 axis). As seen from its x -coordinate, the point $\bar{\ell}$ lies between the parallel axes when $m < 0$, to the right of the \bar{X}_2 axis for $0 < m < 1$ and to the left of \bar{X}_1 for $m > 1$. Lines with $m = 1$ are mapped to the direction with slope b/d in the xy -plane. Here d is the inter-axis distance and b is the constant (intercept) in the equation of ℓ . This shows, that dualities properly reside

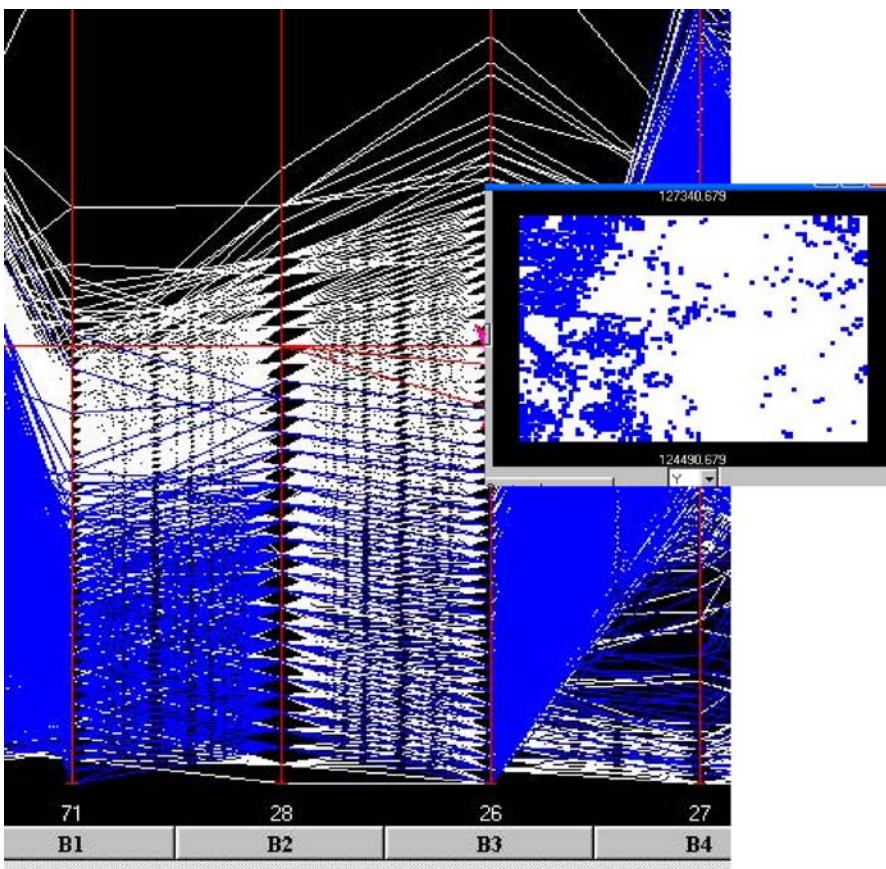


Figure 14.6. Finding regions with vegetation using the *angle query* (left) between the B_2 and B_3 axes. Note the arrowheads on the B_3 axis, which specify the range of angles for the selected lines

in the *projective plane* (the directions are the ideal points) rather than the Euclidean plane. For sets of points that have a “general” direction with a negative slope (i.e., are “negatively correlated”), the lines representing them in \parallel -coords cross each other in-between the axes, and they can be selected with the *pinch* query. For positively correlated sets of points, their corresponding lines cross outside the axes and can be selected with the *angle* query. All of this exemplifies the need to understand some of the basic geometry in order to use the queries effectively and of course to design them properly. Now that we have introduced the three atomic queries, the next stage is to show how they can be combined to construct complex queries.

However, prior to this, Fig. 14.6 (left) begs the question: “what if the B_2 and B_3 axes were *not* adjacent?” Then the pattern and hence their pairwise relation would be missed. Hence the permutation of the axes used for the exploration is important. In particular, what is the minimum number of permutations among N -axes containing the *adjacencies* for all pairs of axes? It turns out Wegman (1990) that M permutations

are needed for even $N = 2M$, and $M+1$ for odd $N = 2M+1$. It is fun to see why. Label the N vertices of a graph with the indices of the variables X_i , $i = 1, \dots, N$, as shown in Fig. 14.8 for $N = 6$. An edge joining vertex i with j signifies that the axes indexed by i, j are adjacent. The graph on the left is a *Hamilton path*, because it contains all of the vertices. Such paths have been studied since Euler first did so in the eighteenth century, and have modern applications to the “travelling salesman” problem and others

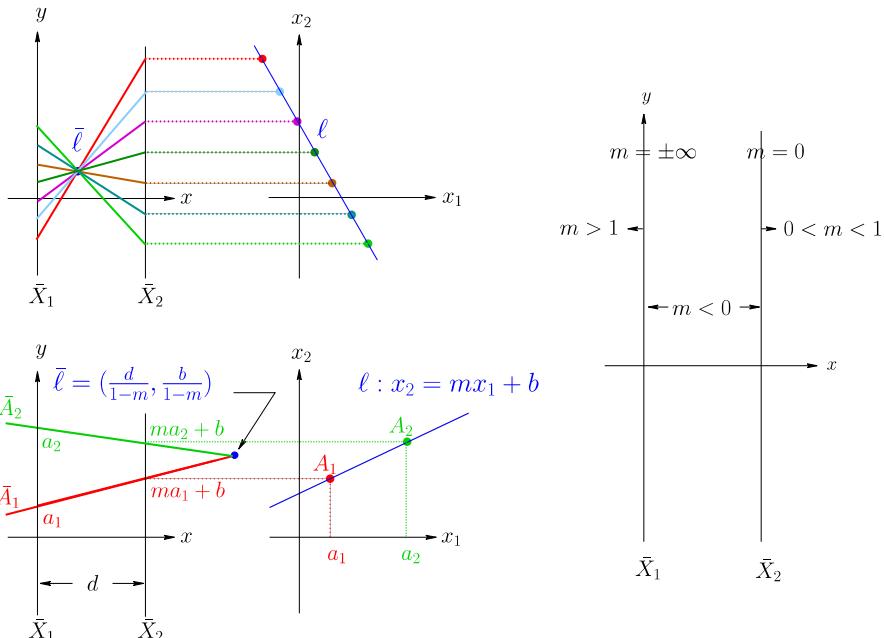


Figure 14.7. (Left) Parallel coordinates induce a point $\bar{\ell} \leftrightarrow \ell$ line duality. (Right) The horizontal position of the point $\bar{\ell}$ representing the line ℓ is determined only by the slope of the line m . The vertical line $\ell : x_1 = a_1$ is represented by the point $\bar{\ell}$ at the value a_1 on the \bar{X}_1 axis

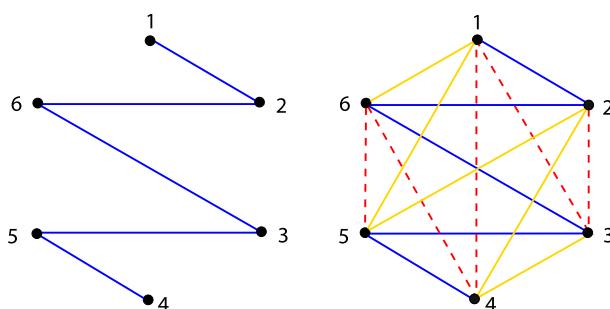


Figure 14.8. (Left) First Hamiltonian path on the vertices labeled 1, ..., 6 corresponding to (axis) index permutation 126354. (Right) The complete graph as the union of the three distinct Hamiltonian paths starting successively at the vertices 1, 2, 3

(Harary 1969, pp. 66; Bollobas 1979, pp. 12). The graph corresponds to the axis index permutation **126354**. On the right, the union with the additional two Hamiltonian paths, starting at vertices 2 and 3, forms the complete graph that contains all possible edges. Hence, the three permutations **126354**, **231465**, **342516** contain all possible adjacent pairs (try it!). The remaining permutations are obtained from the first by successively adding 1 mod 6, and this works for all N .

Returning to EDA, the icon with the *Rubik's Cube* on *Parallax*'s toolbar activates a permutation editor which automatically generates the Hamiltonian permutations (abbr. HP). After scrutinizing the dataset display, the recommended next step is to run through the $O(N/2)$ HP. This is how all nice adjacencies such as the one in Fig. 14.6 are discovered. Then, using the editor, patch your own custom-made permutation containing all of the parts you like into the HP. With this preprocessing cost, referred to earlier in list item 2 of the *Introduction*, the user can set their own best permutation to work with. Of course, there is nothing to prevent one from including axes several times in different positions and experimenting with different permutations during the course of the exploration.

Compound Queries: Financial Data

14.2.3

Next up is the financial dataset in Fig. 14.9. Here the goal is to discover relations that are useful for investments and trading. The data for the years 1986 (second tick on the third axes) and 1992 are selected. In 1986 the **Yen** had the greatest volatility among the three currencies, interests varied in the mid-range, gold had a price gap, while **SP500** was uniformly low. By comparison, in 1992, the **Yen** was stable while **Sterling** was very volatile, interest and the price of gold were low, and the **SP500** was uniformly high. Two *interval* queries are combined with the OR boolean operator (i.e., union) to obtain this picture. We continue to

- “look for the gold” by checking out patterns that catch our attention.

The data for 1986 is isolated in Fig. 14.10 and the lower range in the gold price gap is selected. Gold prices were low until the second week in August, when they jumped and stayed high. The exploration was carried out in the presence of four financial experts, who carefully recorded the relation between low **Yen**, high **3MTB** rates and low **Gold** prices. By the way, a *low* rate of exchange for the **Yen** means that the Yen has high value relative to the \$.

There are two bunches of crossing lines between the sixth and seventh axes in Fig. 14.9, which together comprise more than 80% of the dataset. This, and recalling the previous discussion on the *line* \leftarrow *point* mapping in Fig. 14.7, suggests a strong negative correlation between **Yen** and **3MTB** rates. The smaller cluster in Fig. 14.11 is selected. Moving from the top range of any of the two axes, applying the **I** query and lowering the range causes the other variable's range to rise – a nice way of showing negative correlation interactively.

For the contrarians among us, we also check for positive correlation in Fig. 14.12. We find that it exists when **Gold** prices are low to mid-range, as happened for a period in the 1990s. This is a free investment tip for bucking the main trend, as shown in Fig. 14.11. It is also a nice opportunity to show the *inversion* feature activated by the

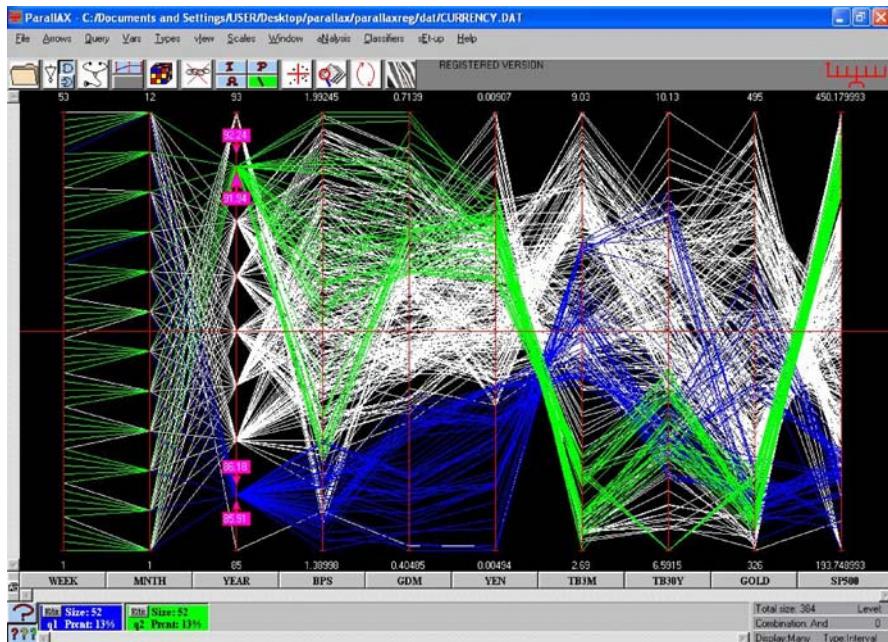


Figure 14.9. Financial data, Week-on Monday, Month, Year (first three axes fix the date); Sterling, Dmark, Yen rates per \$ (4th, 5th, 6th axes); 3MTB, 30YTB interest rates in % (7th, 8th axes); Gold in \$/ounce (9th axis), SP500 index values (10th axis)

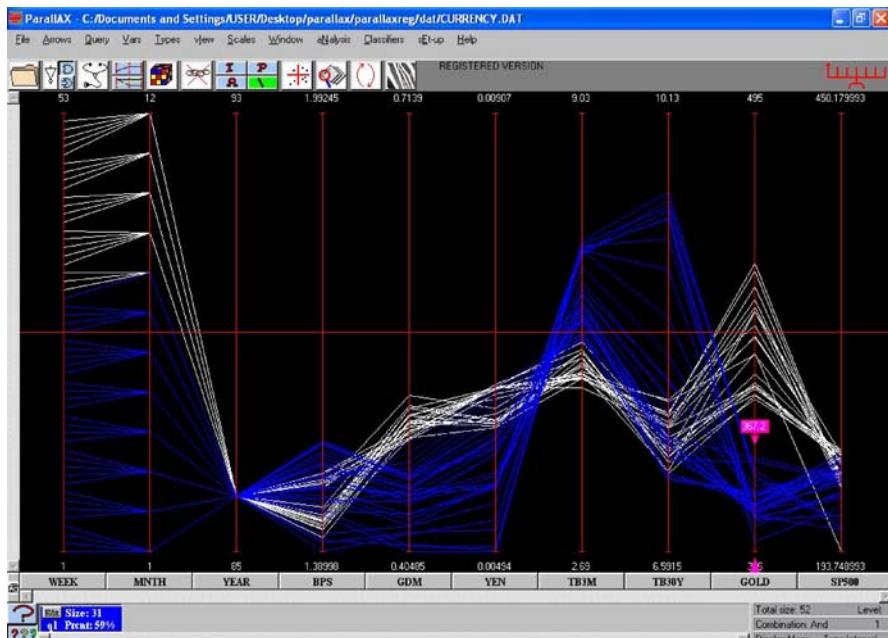


Figure 14.10. In 1986 gold prices jumped in the second week of August. Note the correlation between the low Yen, the high 3MTB rates the and low Gold price range

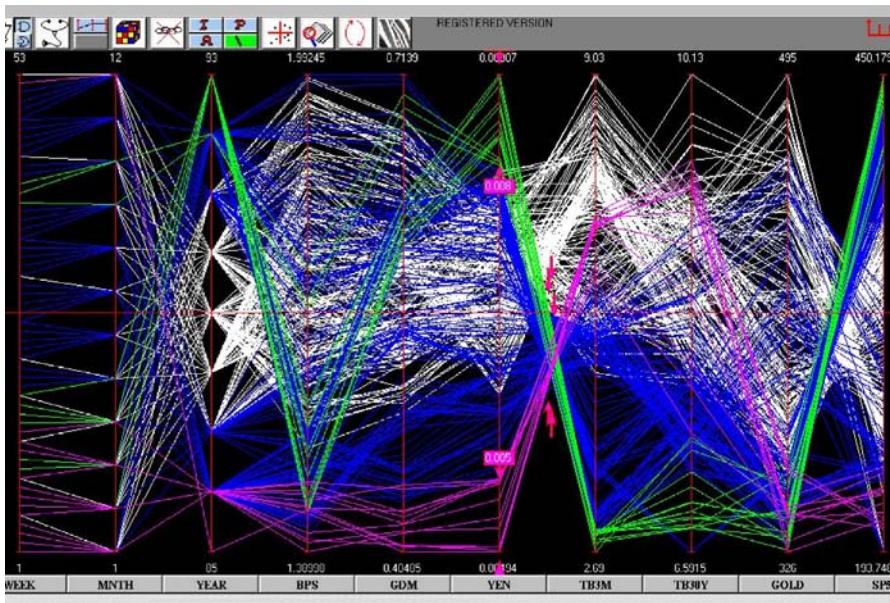


Figure 14.11. The crossing lines between the 6th and 7th axes in Fig. 14.9 show strong negative correlation between the Yen and 3MTB rates. One cluster is selected with the *pinch* query and combined with the high and low ranges on the Yen axis. Data for the years 1986 and 1992 are selected

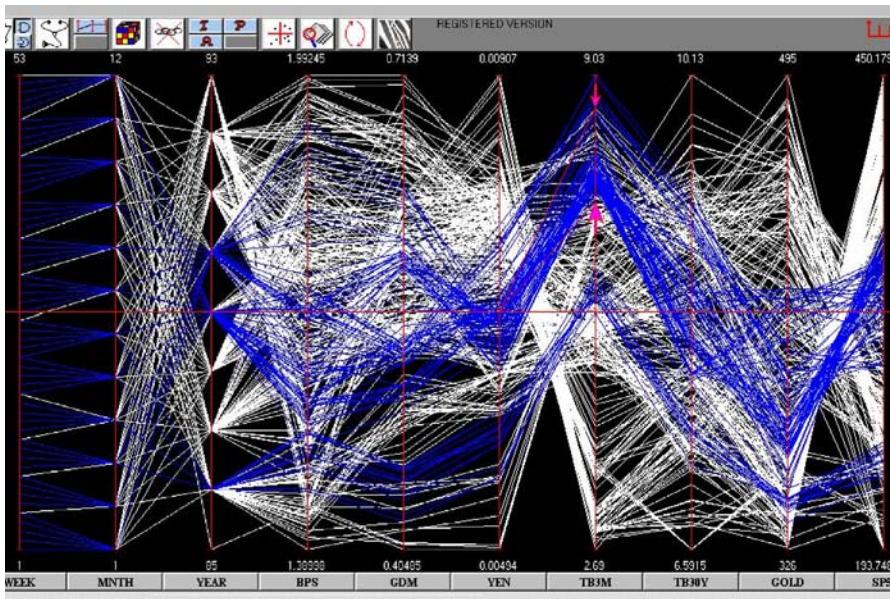


Figure 14.12. A cluster (with positive correlation) that shows that Yen and 3MTB rates move together when Gold prices are low to mid-range

icon with two cyclical arrows. A variable is selected and the min/max values on that axes are inverted. Diverging lines (as obtained for positive correlation) now intersect in Fig. 14.13, making it easier to visually spot the crossing and hence the correlation. Actually, it is worth working with the **A** query, experimenting with various angle ranges and using the inversion to check out or confirm special clusters.

- **vary one of the variables watching for interesting variations in the other variables.**

Doing this on the **Yen** axis (see Fig. 14.14), we strike another gold connection. The (rough) intersection of a bunch of lines joining **Yen** to the **Dmark** corresponds, due to the duality, to their rate of exchange. When the rate of exchange changes, so does the intersection and the price of gold! In other words, movements in currency exchange rates and the price range of gold go together. Are there any indications that are associated with the high range of gold? The top price range is selected (Fig. 14.15), and prompted by the results from the previous query, we check out the exchange rate between **Sterling** and **Dmark** (or **Yen**). The results are stunning: a perfect straight line. The slope is the rate of exchange, which is constant when **Gold** tops out. The relation between **Sterling** and **Dmark** is then checked for different price ranges of **Gold** (Fig. 14.16), and the only regularity found is the one straight line above. Aside from the trading guideline it establishes, we could say that this suggests behind-the-scenes manipulation of the gold market...but we won't. We banish any such thought and proceed with the boolean complement (Fig. 14.17) of an **I** (or any other) query. Not finding anything, we select a narrow but dense range on the **Yen** (Fig. 14.18) and notice an interesting relation between **Dmark**, interest rates and **Gold**.

There is an exploratory step akin to “multidimensional contouring,” which we fondly call the *zebra*; it is activated by the last icon button on the right, with the appropriate skin-color. A variable axis is selected (the **SP500** axis in Fig. 14.19), divided into a number (user-specified) of intervals (four here), and colored differently. This shows the connections (influence) of the intervals for the remaining variables, which are richly structured here, especially for the highest range. So what does it take for the **SP500** to rise? This is a good question and helps introduce Parallax’s classifier. The result, shown in Fig. 14.20, confirms the investment community’s experience that low **3MTB** and **Gold** predict high **SP500**. A comparison with the results obtained on this dataset using other visualization tools would be instructive, although unfortunately they are not available. Still, let us consider how such an analysis would be performed using a scatterplot matrix. There are ten variables (axes), which require 45 pairwise scatterplots each; even with a large monitor screen, these could be no larger than about $2.5 \times 2.5 \text{ cm}^2$. Varying one or more variables in tandem and observing the effects simultaneously over all of the variables in the 45 squares is possible but quite a challenging task. By contrast, the effects of varying **Dmark**, for stable **Yen**, on the two interest rates, **Gold** as well as the remaining variables are easily seen in just *one* plot using \parallel -coords: Fig. 14.18. This example illustrates the difficulties associated with high representational complexity (see Sect. 14.1.2, item 2), which is $O(N^2)$ for the scatterplot matrix but $O(N)$ for \parallel -coords, and made even clearer with the next dataset.

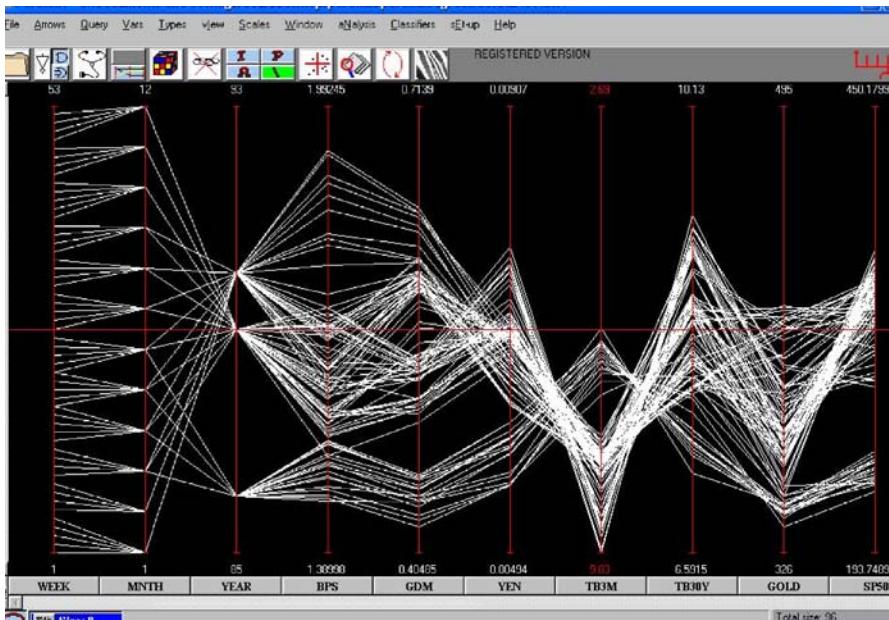


Figure 14.13. The 3MTB axis is inverted so that lines between the Yen-3MTB and 3MTB-30MTB axes in Fig. 14.12 now cross



Figure 14.14. Variations in the rate of exchange of the currencies correlate with movements in the price of Gold

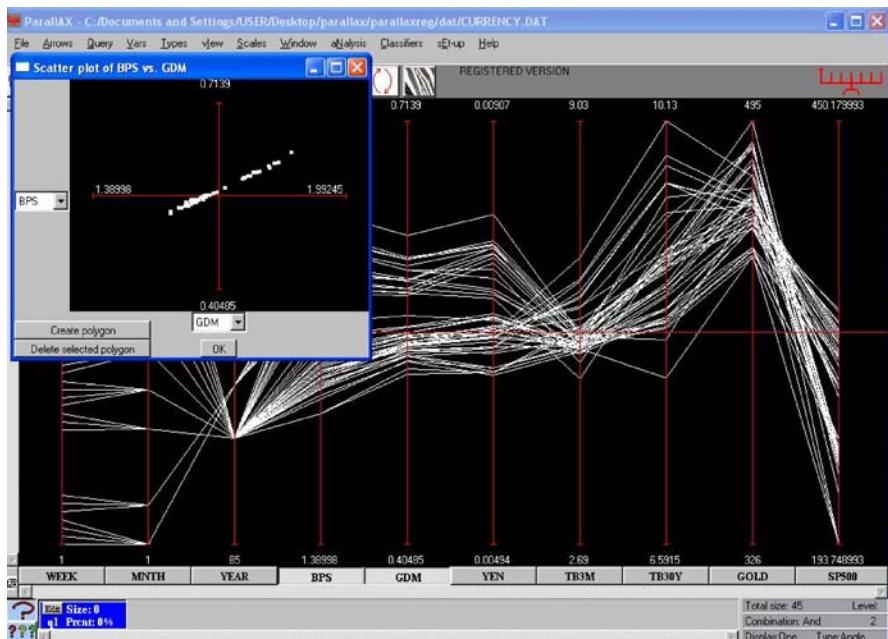


Figure 14.15. High Gold. Note the perfect straight line in the Sterling vs. Dmark plot. The slope is the rate of exchange between them, which remains constant when Gold prices peak

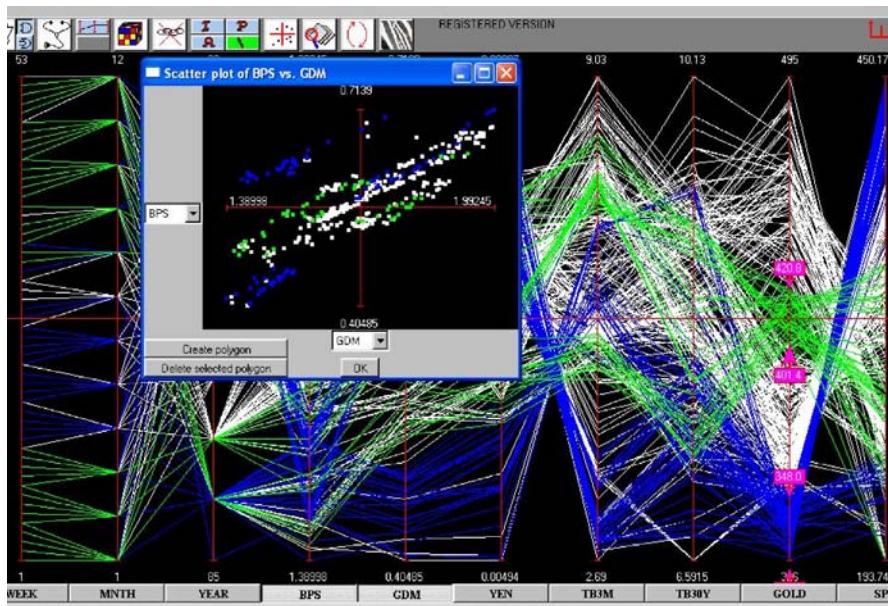


Figure 14.16. [This figure also appears in the color insert.] Two price ranges for Gold. The Sterling vs. Dmark plots for them show no regularity

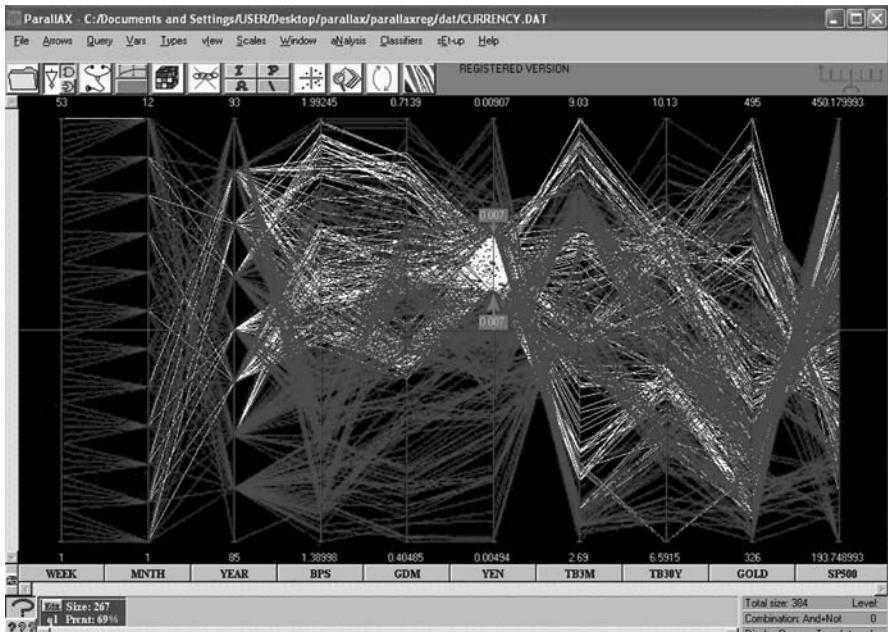


Figure 14.17. The complement of an I query

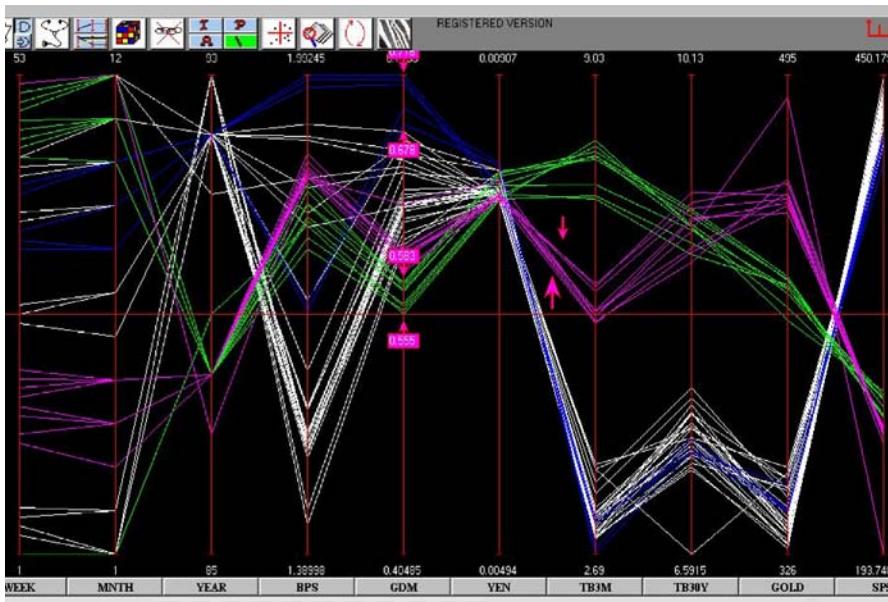


Figure 14.18. For the Yen trading in a narrow range, high Dmark goes with low 3MTB rates, low Dmark goes with high 3MTB rates, while mid 3MTB rates go with high Gold

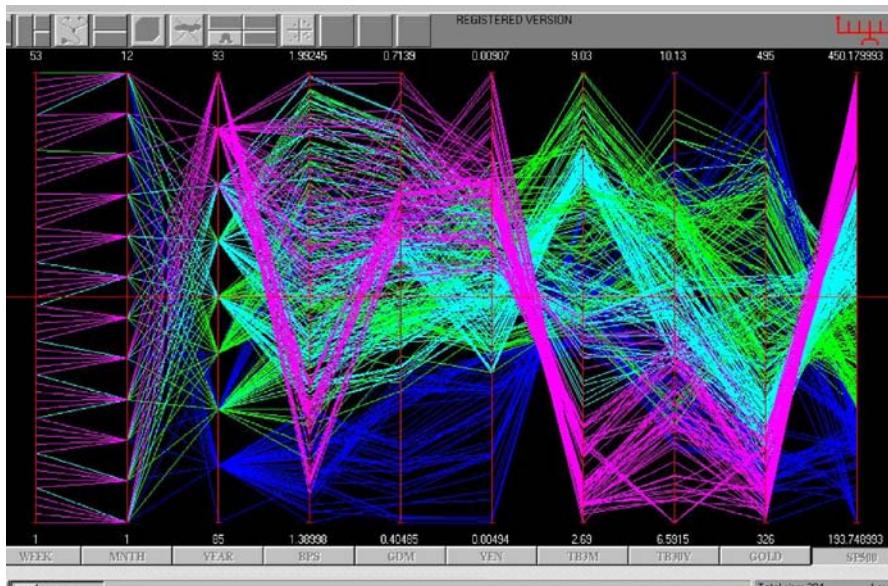


Figure 14.19. [This figure also appears in the color insert.] The zebra partitions and colors the parts differently. A variable, here the SP500 axis, is divided into equal (four in this case) intervals. This quickly reveals interrelationships. In particular, note those for the highest SP500 range and review the next figure...

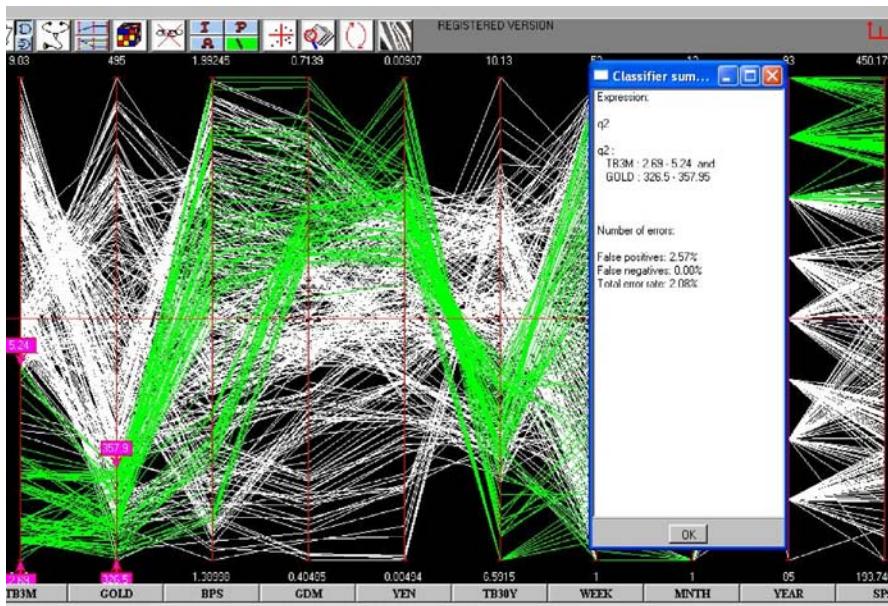


Figure 14.20. The rule for high SP500 is that both 3MTB (the “short-bond” as it is called) and Gold should be low, in this order of importance

Hundreds of Variables

One frequently asked question is “how many variables can be handled with \parallel -coords?” The largest dataset that I have effectively worked with had about 800 variables and 10,000 data entries. Using various techniques developed over the years and the automatic classifier discussed in the next section, it is possible to handle much larger datasets. Still, the relevant admonition is:

- be sceptical about the quality of datasets with large numbers of variables.

When hundreds or more variables are involved, it is unlikely that there are many people around who have a good feel for what is happening (as confirmed by my own experience). A case in point is the dataset shown in Fig. 14.21, consisting of instrumentation measurements of a complex process. An immediate observation was that many of the instruments recorded 0 throughout the period that the measurements were taken, something which had not been noticed previously. Another curiosity was the series of repetitive patterns on the right. It turns that several variables were measured in more than one location using different names. When the dataset was cleaned up (removing superfluous information), it was initially reduced to about 90 variables, as shown in Fig. 14.22, and eventually to about 30 that contained the information of real interest. By my tracking, the phenomenon of repetitive measurements is widespread, with at least 10% of the variables in large datasets being duplicates or near-duplicates, possibly due to instrumental nonuniformities, as suggested by the two-variable scatterplot in Fig. 14.22. Here, the repetitive observations were easily detected due to the fortuitous variable permutation in the display. Since repetitive

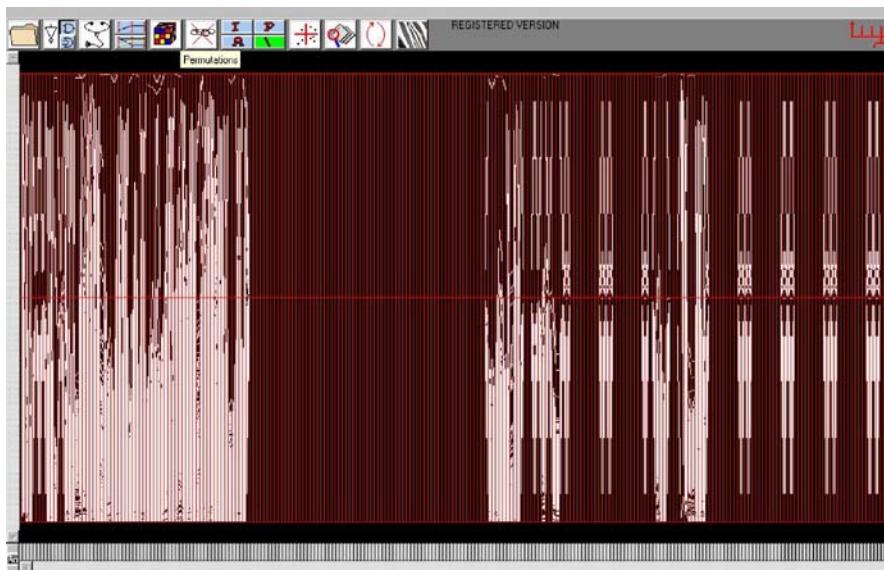


Figure 14.21. Manufacturing process measurements: 400 variables

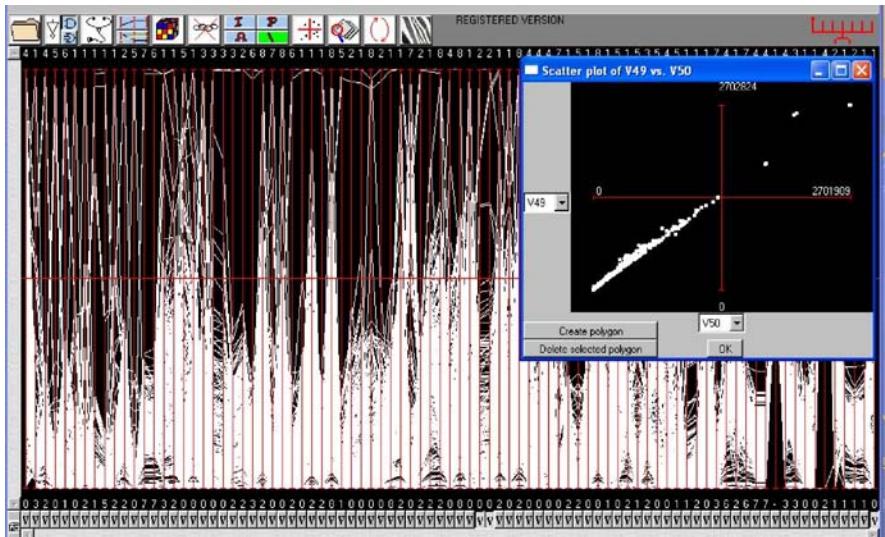


Figure 14.22. The manufacturing dataset after “cleanup,” which left about 90 variables

measurements occur frequently, it may be worth adding an automated feature to the software that detects and identifies the suspect variables.

This brief exposure is just an indication that large (in terms of dimensions – i.e., number of variables) datasets can still be usefully explored in \parallel -coords.

A different example of EDA on a process control dataset is given in Inselberg (1998), where compound queries turned out to be very useful. This reminds us to add, to the list of exploration guidelines, arguably the most important one:

- **test the assumptions, especially the “I am really sure of”s.**

14.3

Classification

Although it is fun to undertake this kind of exploration, the level of skill and patience required tends to discourage some users. It is not surprising then that the most persistent requests and admonitions have been for tools which, at least partially, automate the knowledge discovery process (Inselberg and Avidan, 1999).

Classification is a basic task in data analysis and pattern recognition, and an algorithm that performs it is named *Classifier* (Quinlan, 1993; Fayad et al., 1996; Mitchell, 1997). The input is a dataset P and a designated subset S . The output is a characterization, a set of conditions or rules, to distinguish elements of S from all other members of P , the “global” dataset. The output may also be that there is insufficient information to provide the desired distinction.

With parallel coordinates, a dataset P with N variables is transformed into a set of points in N -dimensional space. In this setting, the designated subset S can be de-

scribed by a hypersurface which only includes the points of S . In practical situations, the strict enclosure requirement is dropped and some points of S may be omitted (“false negatives”), while some points of $P - S$ are allowed (“false positives”) in the hypersurface. The description of such a hypersurface is equivalent to the rule for identifying, within some acceptable error, the elements of S . Casting the problem in a geometrical setting allows us to visualize how to approach the classification algorithm. This entails:

1. using an efficient “wrapping” (a convex-hull approximation) algorithm to enclose the points of S in a hypersurface S_1 containing S , and in general some points of $P - S$ too; so $S \subset S_1$,
2. the points in $(P - S) \cap S_1$ are isolated and the wrapping algorithm is applied to enclose them, and usually also some points of S_1 , producing a new hypersurface S_2 with $S \supset (S_1 - S_2)$,
3. the points in S not included in $S_1 - S_2$ are then marked for input to the wrapping algorithm, and a new hypersurface S_3 is produced containing these points as well as some other points in $P - (S_1 - S_2)$, resulting in $S \subset (S_1 - S_2) \cup S_3$,
4. the process is repeated, alternately producing upper and lower containment bounds for S ; termination occurs when an error criterion (which can be user-specified) is satisfied or when convergence is not achieved.

It can and does happen that the process does not converge when P does not contain sufficient information to characterize S . It may also happen that S is so “porous” (sponge-like) that an inordinate number of iterations are required. On convergence, say at step $2n$, the description of S is provided as:

$$S \approx (S_1 - S_2) \cup (S_3 - S_4) \cup \cdots \cup (S_{2n-1} - S_{2n}). \quad (14.3)$$

This is the terminating expression resulting from the algorithm that we call *Nested Cavities* (abbr. NC).

The user can select a subset of the variables available and restrict the rule generation to these variables. In certain applications, for example as in process control, not all of the variables can be controlled, and hence it is useful to have a rule that involves only the accessible (i.e., controllable) variables. An important fringe benefit is the useful ordering that emerges from dimensionality selection, which is completely dataset-specific. With the algorithm being display-independent, there is no inherent limitation on the size and number of variables in the dataset. Summarizing for NC,

- an approximate convex-hull boundary is obtained for each cavity,
- utilizing properties of the representation of multidimensional objects in \parallel -coords, a very low polynomial worst case complexity of $O(N^2|P|^2)$ is obtained for variable number N and dataset size $|P|$; it is worth contrasting this with the often unknown, unstated, or very high (even exponential) complexity of other classifiers,
- an intriguing prospect, due to the low complexity, is that the rule can be derived in near real-time, meaning that the classifier could be adapted to changing conditions,

- the minimal subset of variables needed for classification is found,
- the rule is given explicitly in terms of conditions on these variables, in terms of included and excluded intervals, and provides “a picture” that shows complex distributions with regions where there are data and “holes” with no data, thus providing insight to domain experts.

During 1990–1993 Michie et al. (1994), on behalf of the ESPRIT program of the European Union, made extensive studies of several classifiers applied to diverse datasets. About 23 different classifiers were applied to about 20 datasets for comparative trials in the *StatLog* project. This was designed to test classification procedures on large-scale commercially important problems in order to determine the suitability of the various techniques to industry. The results obtained by NC are compared with those obtained by other well-known classifiers used in *Statlog* on two benchmark datasets, and are shown in the accompanying tables.

1. *Satellite image dataset*. This has over 6,000 items; NC’s classification error was 9%, k-NN was next with 9.4%, the remaining classifiers gave errors of as much as 30%, and one was unable to classify at all. On a *Vowel recognition dataset* with about 1,000 data items, NC was top with 7.9%, next was CART-DB 10%, while the rest reached down to 66%, with many unable to provide a classification rule, as shown in Table 14.1.

Rank	Classifier	Error rate %	
		Train	Test
1	NC	4.3	9.0
2	k-NN	8.9	9.4
3	LVQ	4.8	10.5
4	DIPOL92	5.1	11.1
5	RBF	11.1	12.1
6	ALLOC80	3.6	13.2
7	IndCART	2.3	13.8
8	CART	7.9	13.8
9	Backprop	11.2	13.9
10	Baytree	2.0	14.7
11	CN2	1.0	15.0
12	C4.5	4.0	15.0
13	NewID	6.7	15.0
14	Cal5	12.5	15.1
15	Quadisc	10.6	15.5
16	AC ²	11.3	15.7
17	SMART	12.3	15.9
18	Cascade	11.2	16.3
19	Logdisc	11.9	16.3
20	Discrim	14.9	17.1
21	Kohonen	10.1	17.9
22	CASTLE	18.6	19.4
23	NaiveBay	30.8	28.7
24	ITrule	Failed	Failed

Table 14.1. Summary of the *Statlog* results and comparison with the Nested Cavities (NC) classifier for the satellite image data

2. *Vowel recognition data.* The data collection process involves digital sampling speech with acoustic signal processing, followed by recognition of the phonemes, groups of phonemes and words. The goal here is a speaker-independent rule based on ten variables of eleven vowels that occur in various words spoken (recorded and processed) by fifteen British male and female speakers. Deterding (1989) collected this dataset of vowels, which can be found in the CMU benchmark repository in the WWW. There are 528 entries for training and 462 for testing. Three other types of classifiers were also applied to this dataset: neural networks and k-NN by Robinson and Fallside (1988), and decision trees by Shang and Breiman (1996). For the sake of variety, both versions of our classifier were used and a somewhat different error test procedure was used. The results are shown in Table 14.2.
3. *A neural-pulse dataset.* This has interesting and unusual features. There are two classes of neurons, whose outputs to stimuli are to be distinguished. They consist of 32 different pulses measured in a monkey's brain (poor thing!). There are 600 samples with 32 variables (the pulses). This dataset was given to me by a very competent group (that of Prof. Coiffman, CS & Math. Depts. at Yale Univ.), who had been working on it but had been unable to obtain a viable rule with the classification methods they used. Remarkably, with NC convergence is obtained based on only nine of the 32 parameters. The resulting ordering shows a striking separation. In Fig. 14.23, the first pair of variables x_1, x_2 is plotted as originally given on the left. On the right, the best pair x_{11}, x_{14} , as chosen by the classifier's ordering, speaks for itself. By the way, to discover this finding manually would require the construction of a scatterplot matrix with 496 pairs, and then careful inspection and comparison of the individual plots. The implementation provides all the next best sections to complete the rule's visualization. The dataset consists of two "pretzel-like" clusters winding closely in 8-D, one (the complement in this case) enclosing the other. Note that the classifier can actually describe highly complex regions that carve the cavity shown. One can understand why the separation of clusters by hyperplanes or nearest-neighbor techniques can fail badly on such datasets. The rule has an error of 4%.

Table 14.2. Summary of classification results for the vowel dataset

Rank	Classifier	Testing mode	Test error rate %
1	Nested Cavities (NC)	Cross-validation	7.9
2	CART-DB	Cross-validation	10.0
3	Nested Cavities (NC)	Train & Test	10.5
4	CART	Cross-validation	21.8
5	k-NN	Train & Test	44.0
6	RBF	Train & Test	46.5
7	Multilayer perceptron	Train & Test	49.4
8	Single-layer perceptron	Train & Test	66.7

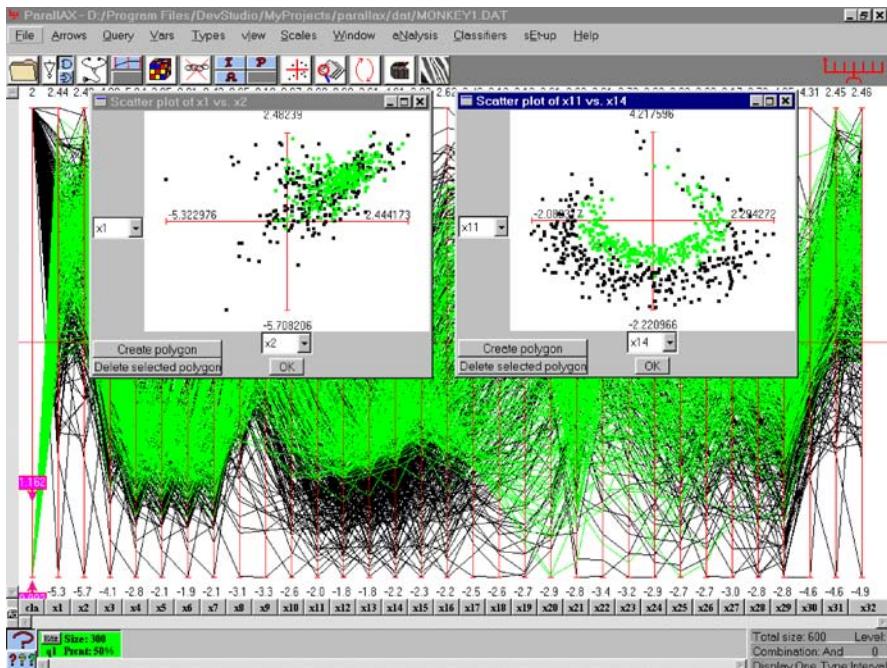


Figure 14.23. The monkey neural pulses dataset, showing the separation achieved by the first two of the nine (out of a total of 32) parameters obtained from the dimensionality selection

The rules are explicit, “visualizable” and yield dimensionality selection choosing and ordering the minimal set of variables needed to state the rule without loss of information. There are variations that apply to some situations where the NC classifier fails, such as the presence of several large “holes” (see Inselberg and Avidan (1999)). Further, the fact that the classification rule is the result of several iterations suggests heuristics for dealing with the pesky problem of over-fitting. The iterations can be stopped when the corrections in Eq. 14.3 become very small, i.e., S_i consists of a small number of points. The number of iterations is user-defined, and the resulting rule yields an error during the test stage that is more stable under variations in the number of points of the test set. In addition, the user can exclude variables from being used in the description of the rule; those ordered last are the ones that provide the smaller corrections and hence are more liable to over-correct.

14.4 Visual and Computational Models

Finally, we illustrate the methodology’s ability to model multivariate relations in terms of hypersurfaces – just as we model a relation between two variables by a planar region. Then, by using the interior point algorithm, as shown for example in Fig. 14.31,

we can do trade-off analyses, discover sensitivities, understand the impact of constraints, and in some cases perform optimization using the model. For this purpose, we shall use a dataset consisting of the outputs from various economic sectors and other expenditures of a particular (and real) country. It consists of the monetary values over several years for the **Agricultural**, **Fishing**, and **Mining** sector outputs, the **Manufacturing** and **Construction** industries, together with **Government**, miscellaneous spending and resulting GNP; eight variables altogether. We will not address the full ramifications of constructing a model from the data. Rather, we intend to illustrate how \parallel -coords can be used as a modeling tool. Using the least squares technique, we “fit” a function to this dataset and are not concerned at this stage about whether the choice is “good” or not. The function obtained bounds a region in R^8 , and is represented by the upper and lower curves shown in Fig. 14.24.

The picture is in effect a simple visual model of the country’s economy, incorporating its capabilities, limitations and interrelationships among the sectors, etc. A point inside the region satisfies all of the constraints simultaneously, and therefore represents (i.e., the eight-tuple of values) a feasible economic policy for that country. We can construct such points using the interior point algorithm. This can be done interactively by sequentially choosing values of the variables, and we see the result of one such choice in Fig. 14.24. Once the value of the first variable is chosen (in this case the agricultural output) within its range, the dimensionality of the region is reduced by one. In fact, the upper and lower curves between the second and third axes correspond to the resulting seven-dimensional hypersurface, and show the avail-

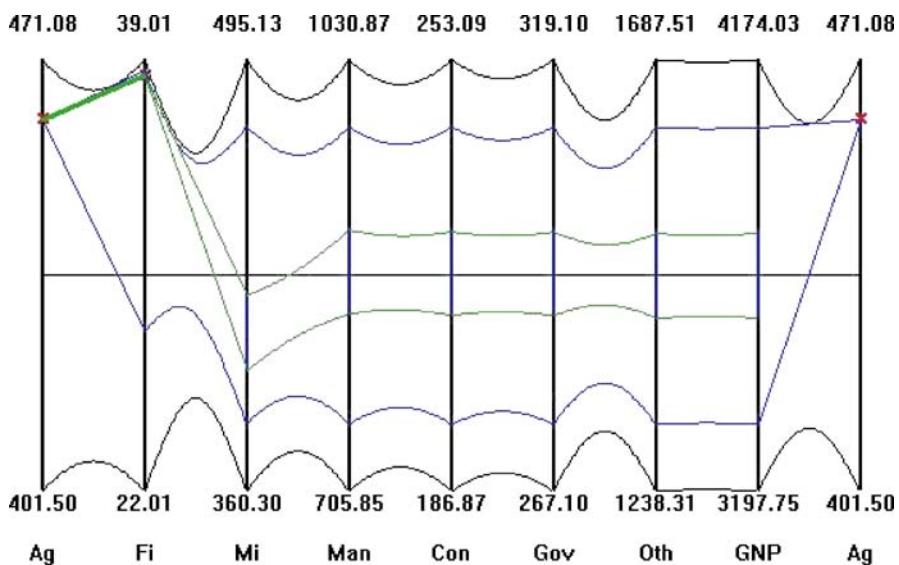


Figure 14.24. Model of a country’s economy: choosing high agricultural and high fishing output forces low mining output

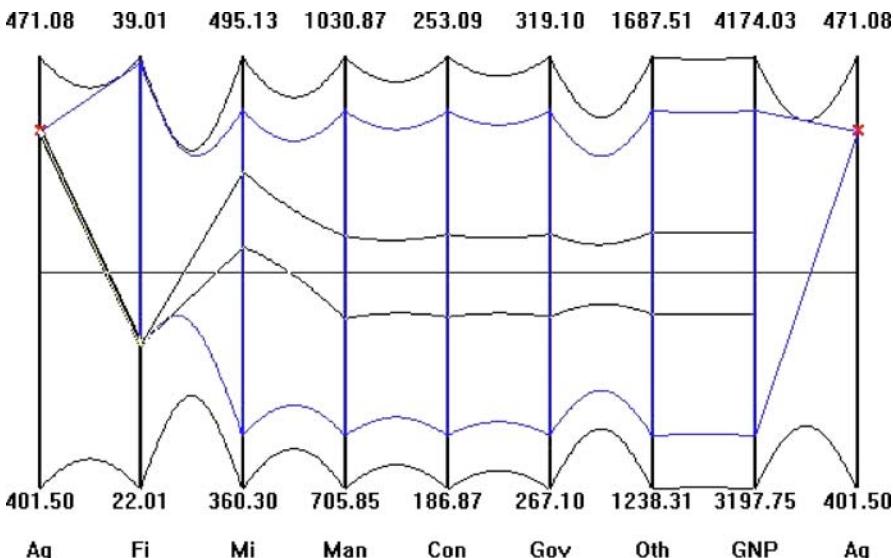


Figure 14.25. Competition for labor between the fishing and mining sectors: compare with previous figure

able range of the second variable (fishing) given the constraint. In fact, this can be seen (but is not shown here) for the other variables. That is, due to the relationship between the eight variables, a constraint on one of them impacts all of the remaining ones and restricts their ranges. The display allows us to experiment and actually see the impacts of such decisions downstream. By interactively varying the value chosen for the first variable, we found that it is not possible to have a policy that favors agriculture without also favoring fishing, and vice versa. Proceeding further, a very high value from the available range of fishing is chosen, and this corresponds to very low values of the mining sector. By contrast, in Fig. 14.24 we see that a low value in fishing yields high values for the mining sector. This inverse correlation was examined, and it was found that the country in question has a large number of migrating semi-skilled workers. When the fishing industry is doing well, most of them are attracted to it, leaving few available to work in the mines, and vice versa. A comparison between the two figures shows the competition for the same resource between mining and fishing. It is especially instructive to discover this interactively. The construction of the interior point proceeds in the same way. In Fig. 14.31 (right), the same construction is shown but for a more complex 20-dimensional hypersurface.

Parallel Coordinates: Quick Overview

14.5

Lines

14.5.1

An N -dimensional line ℓ can be described by the following $N - 1$ linear equations:

$$\ell : \begin{cases} \ell_{1,2} : x_2 = m_2 x_1 + b_2 \\ \ell_{2,3} : x_3 = m_3 x_2 + b_3 \\ \dots \\ \ell_{i-1,i} : x_i = m_i x_{i-1} + b_i \\ \dots \\ \ell_{N-1,N} : x_N = m_N x_{N-1} + b_N, \end{cases} \quad (14.4)$$

each with a pair of adjacently indexed variables. In the $x_{i-1}x_i$ plane, the relation labeled $\ell_{i-1,i}$, $N = 2, \dots, N$ is a line; based on the *line \leftrightarrow point* 2-D correspondence, this line can be represented by the point

$$\bar{\ell}_{i-1,i} = \left(\frac{1}{(1-m_i)} + (i-2), \frac{b_i}{(1-m_i)} \right) \quad (14.5)$$

Here the inter-axis distance is 1, so $i - 2$ is the distance between the y (or \bar{x}_1) and \bar{x}_{i-1} axes. Actually, any $N - 1$ independent equations like

$$\ell_{i,j} : x_i = m_{i,j} x_j + b_{i,j}, \quad (14.6)$$

can equivalently specify the line ℓ , since Eq. 14.6 is the projection of ℓ onto the $x_i x_j$ 2-D plane and $N - 1$ of these independent projections completely describe ℓ . There is a beautiful and very important relationship between three such points, as illustrated in Fig. 14.26 (left).

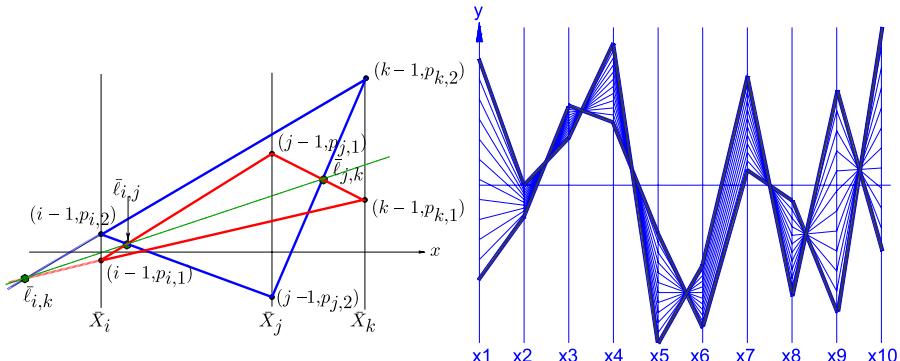


Figure 14.26. (Left) The three points $\bar{\ell}_{i,j}$, $\bar{\ell}_{j,k}$, $\bar{\ell}_{i,k}$ are collinear for $i \neq j \neq k$. (Right) A line interval in 10-D

For a line ℓ in 3-D, the three points $\bar{\ell}_{12}, \bar{\ell}_{13}, \bar{\ell}_{23}$ are collinear; we denote this line by $\bar{\ell}$, and any two represent ℓ . A polygonal line on all of the $N - 1$ points, as given by Eq. 14.5 or their equivalent, represents a point on the line ℓ . Conversely, two points determine a line ℓ . Starting with the two polygonal lines representing the points, the $N - 1$ intersections of their \bar{X}_{i-1}, \bar{X}_i portions are the $\bar{\ell}_{i-1,i}$ points for the line ℓ . A line interval in 10-D and several of its points is seen in Fig. 14.26 (right). By the way, it is essential to index the points $\bar{\ell}$.

14.5.2

Planes and Hyperplanes

While a line can be determined from its projections, a plane cannot, even in 3-D. A new approach is called for Eickemeyer (1992). Rather than discerning a p -dimensional object from its points, it is described in terms of its $(p-1)$ -dimensional subsets constructed from the points. Let's see how this works. In Fig. 14.27 (left), polygonal lines representing a set of coplanar points in 3-D are shown. Even the most persistent pattern-seeker will not detect any clues hinting at a relationship between the points, much less a linear one, based on this picture. Instead, for each pair of polygonal lines, the line $\bar{\ell}$ of the three-point collinearity described above is constructed. The result, shown on the right, is stunning. All the $\bar{\ell}$ lines intersect at a point, which turns out to be characteristic of coplanarity, although this is not enough information to specify the plane. Translating the first axis \bar{X}_1 to the position \bar{X}'_1 , one unit to the right of the \bar{X}_3 axis, and repeating the construction yields the second point shown in Fig. 14.28 (left). For a plane given by

$$\pi : c_1x_1 + c_2x_2 + c_3x_3 = c_0 , \quad (14.7)$$

the two points, in the order they are constructed, are respectively

$$\bar{\pi}_{123} = \left(\frac{c_2 + 2c_3}{S}, \frac{c_0}{S} \right), \quad \bar{\pi}'_{1'23} = \left(\frac{3c_1 + c_2 + 2c_3}{S}, \frac{c_0}{S} \right), \quad (14.8)$$

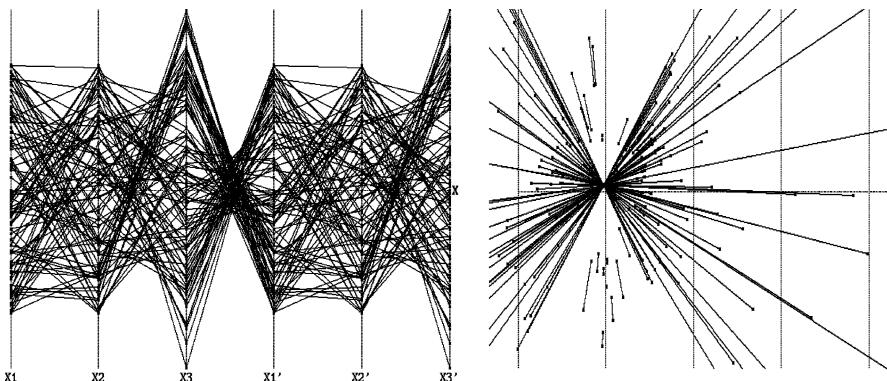


Figure 14.27. (Left) The polygonal lines on the first three axes represent a set of coplanar points in 3-D. (Right) Coplanarity! Lines are formed on the plane using the three-point collinearity intersect at a point

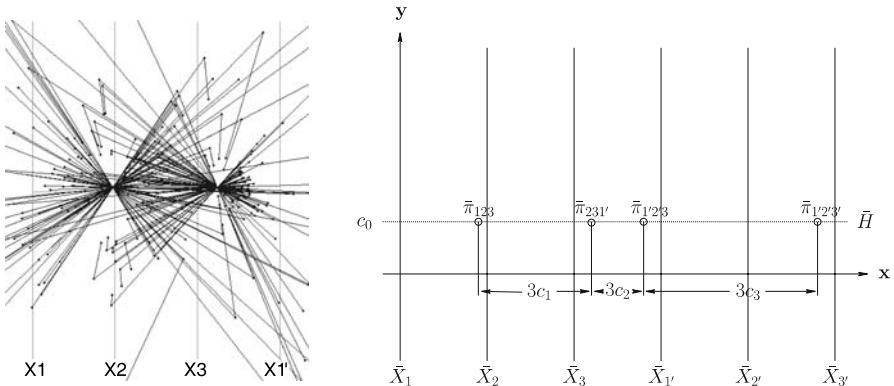


Figure 14.28. (Left) The two points at which the lines intersect uniquely determine a plane π in 3-D. (Right) The coefficients of $\pi : c_1x_1 + c_2x_2 + c_3x_3 = c_0$ can be read from the picture produced by four points that are similarly constructed by consecutive axis translations!

for $S = c_1 + c_2 + c_3$. The three subscripts correspond to the three variables that appear in the equation of the plane, and distinguish them from the points with two subscripts, which represent lines. The second and third axes can also be consecutively translated, as indicated in Fig. 14.27 (left), repeating the construction to generate two more points denoted by $\tilde{\pi}_{1'2'3}$, $\tilde{\pi}_{1'2'3'}$. These points can also be found in another, easier, way. The gist of all this is shown in Fig. 14.28 (right). The distance between successive points is $3c_i$. The equation of the plane π can actually be read from the picture!

In general, a hyperlane in N dimensions is represented uniquely by $N - 1$ points, each with N indices. There is an algorithm that constructs these points *recursively*, raising the dimensionality by one at each step, as is done here starting from points (zero-dimensional) and constructing lines (one-dimensional). By the way, all of the nice high-dimensional projective dualities, like *point* \leftrightarrow *hyperplane*, *rotation* \leftrightarrow *translation*, etc., hold. Further, a multidimensional object represented in \parallel -coords can still be recognized after it has been acted on by projective transformation (i.e., translation, rotation, scaling and perspective). The recursive construction and its properties are at the heart of the \parallel -coords visualization.

Challenge: Visualizing Families of Proximate Planes

Returning to 3-D, it turns out that for points like those in Fig. 14.27 which are “nearly” coplanar (i.e., have small errors), the construction produces a pattern that is very similar to that in Fig. 14.28 (left). A little experiment is in order. Let us return to the family of *proximate* (i.e., close) planes generated by

$$\Pi = \{ \pi : c_1x_1 + c_2x_2 + c_3x_3 = c_0, c_i \in [c_i^-, c_i^+], i = 0, 1, 2, 3 \} . \quad (14.9)$$

We randomly choose values for c_i within the allowed intervals to determine a plane $\pi \in \Pi$, keeping $c_0 = 1$ initially, and then we plot the two points $\tilde{\pi}_{123}$, $\tilde{\pi}_{1'23}$ as shown in Fig. 14.29 (left). Closeness is apparent, and more significantly the distribution of

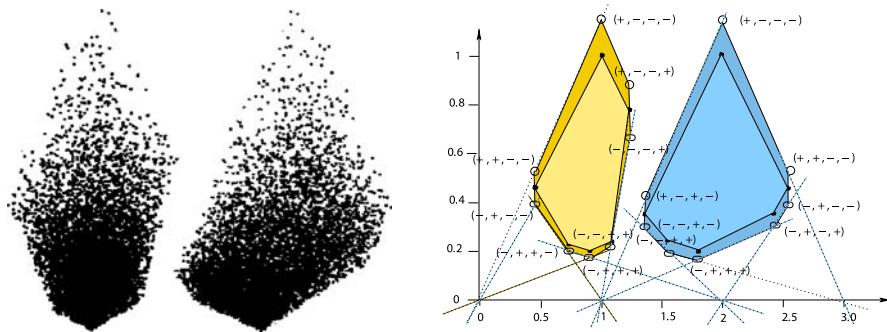


Figure 14.29. (Left) Pair of point clusters representing close planes. (Right) The hexagonal regions (interior) contain the points $\tilde{\pi}_{123}$ (left) and $\tilde{\pi}'_{123}$ for the family of planes with $c_0 = 1$ and $c_1 \in [1/3, 1.5]$, $c_2 \in [1/3, 2.5]$, $c_3 \in [1/3, 1]$. For c_0 varying ($c_0 \in [.85, 1.15]$ here), the (exterior) regions are octagonal with two vertical edges

the points is not chaotic. The outline of two hexagonal patterns can be discerned. The family of planes are “visualizable,” as are variations in several directions. It is possible to see, estimate and compare errors.

Let’s not maintain the suspense any longer: in 3-D the set of pairs of points representing the family of proximate planes form two convex hexagons when $c_0 = 1$ (an example is shown in Fig. 14.29, right), and are contained in octagons, each of which has two vertical edges for varying c_0 . In general, a family of proximate hyperplanes in N-D is represented by $N - 1$ convex $2N$ -agons when $c_0 = 1$ or $2(N + 1)$ -agons for varying c_0 (see Matskewich et al. (2000) and Inselberg (2008) for more recent results). These polygonal regions can be constructed with $O(N)$ computational complexity. Upon choosing a point in one of the polygonal regions, an algorithm matches the remaining $N - 2$ points from the remaining convex polygons that represent hyperplanes in the family. Each hyperplane in the family can be identified by its $N - 1$ points.

Earlier, we proposed that visualization is not about seeing lots of things, but rather it is about discovering **relations** among them. While a display of randomly sampled *points* from a family of proximate hyperplanes is utterly chaotic (the mess in Fig. 14.27, right, is from points in just *one* plane), their **proximate coplanarity relation** corresponds to a clear and compact pattern. With \parallel -coords, we can focus and *concentrate* the relational information rather than wallowing in the details, leading to the phrase “without loss of information” when referring to \parallel -coords. This is the methodology’s real strength, and where I believe the future lies. Here then is the visualization challenge. How else can we detect and see proximate coplanarity?

14.5.3 Nonlinear Multivariate Relations: Hypersurfaces

A relation between two real variables is represented geometrically by a unique region in 2-D. Analogously, a relation between N variables corresponds to a hypersurface in N-D, hence the need to say something about the representation of hypersurfaces in \parallel -coords. A smooth surface in 3-D (and also N-D) can be described as the envelope

of all of its tangent planes. This is the basis for the representation shown in Fig. 14.30 (left). Every point on the surface is mapped into the two points representing its *tangent plane at the point*. This generates two planar regions and $N - 1$ such regions in N -D. These regions are linked, just like the polygons above, to provide the $N - 1$ points representing each tangent hyperplane and therefore reconstruct the hypersurface. Classes of surfaces can be immediately distinguished by their \parallel -coords displays (see Hung and Inselberg (1992) and Inselberg (2008) for more recent results). For developable surfaces, the regions consist of boundary curves only (no interior points); the regions for ruled surfaces have grids consisting of straight lines; while quadric surfaces have regions with conic boundaries. These are just some examples.

There is a simpler but inexact surface representation that is quite useful when used judiciously. The polygonal lines representing points on the boundary are plotted, and their envelope “represents” the surface; the “ ” are a reminder that this is not a *unique* representation. Figure 14.31 (left) shows the upper and lower envelopes for a sphere in 5-D, which consist of four overlapping hyperbolae; this must be distinguished from that in Fig. 14.30 (right), which is exact as determined by the sphere’s tangent planes. By retaining the exact surface description (i.e., its equation) internally, interior points can be constructed and displayed, as shown for the 5-D sphere in Fig. 14.31 (left). The same construction is shown on the right, but for a more complex 20-dimensional convex hypersurface (“model”). The intermediate curves (upper and lower) also provide valuable information and preview coming attractions. They indicate the neighborhood of the point (represented by the polygonal line) and provide a feel for the local curvature. Note the narrow strips (as compared to the surrounding ones), which indicate the critical variables where the point is bumping the boundary. A theorem guarantees that a polygonal line which is in-between all of the intermediate curves/envelopes represents an interior point of the hypersurface, and all interior points can be found in this way. If the polygonal line is tangent to any one of the intermediate curves then it represents a boundary point, while it represents an exterior point if it crosses any one of the intermediate curves. The latter enables us to see, in

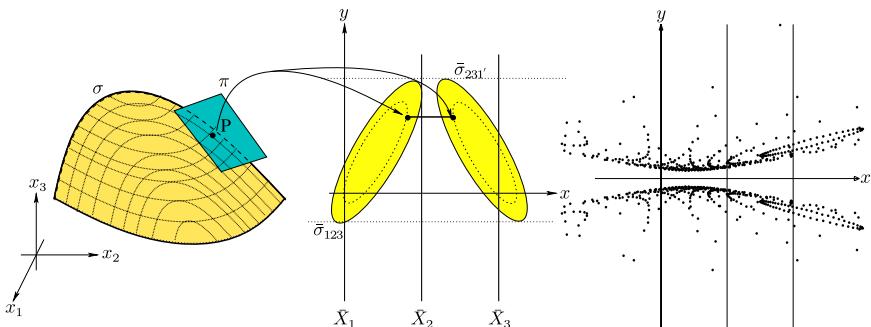


Figure 14.30. (Left) A smooth surface σ is represented by two planar regions $\bar{\sigma}_{123}, \bar{\sigma}_{231}$, consisting of pairs of points that represent its tangent planes. (Right) One of the two hyperbolic regions representing a sphere in 3-D

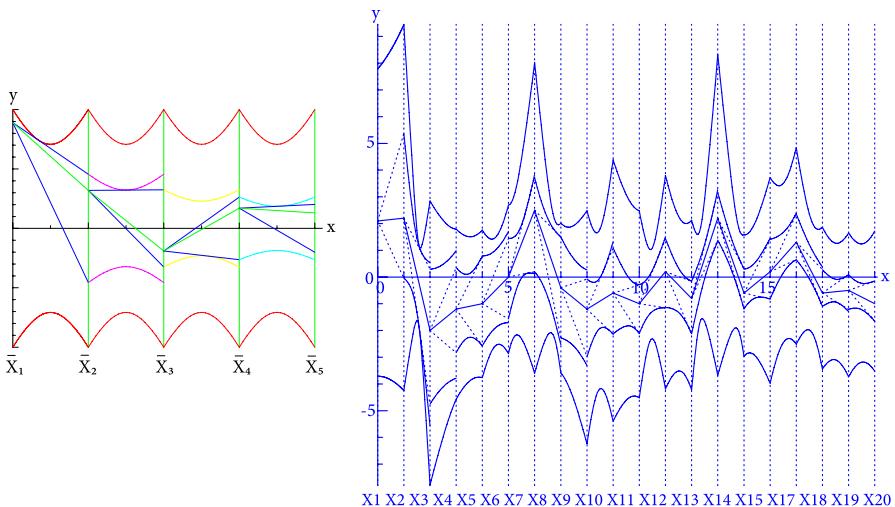


Figure 14.31. (Left) A sphere in 5-D, showing the construction of an interior point (polygonal line).

(Right) The general interior point (polygonal line) construction algorithm shown for a convex hypersurface in 20-D

an application, the first variable for which the construction failed and what is needed to make corrections. By varying the choice of value over the available range of the variable interactively, sensitive regions (where small changes produce large changes downstream) and other properties of the model can easily be elucidated. Once the construction of a point is complete, it is possible to vary the values of each variable and see how this effects the remaining variables. This enables us to perform trade-off analysis, thus providing a powerful tool for decision support, process control and other applications. As new data are made available, the model can be updated so that decisions can be based on the most recent information. This algorithm is used in an earlier example (see Figs. 14.24, 14.25).

14.6

Future

Searching for *patterns* in a \parallel -coords display is what skilful exploration is all about. If there are multivariate relations in the dataset, the patterns *are there*, although they may be covered by the overlapping polygonal lines; but that is not all. Our vision is not multidimensional. We do not perceive a three-dimensional room from its (zero-dimensional points), but from the two-dimensional planes that enclose and define it. The recursive construction algorithm approaches the visualization of p -dimensional objects from their $p-1$ -dimensional components (one dimension less) in exactly the same way. We advocate the inclusion of this algorithm in our armory of interactive analysis tools. Any p -dimensional relations that exist are revealed by the pattern of the representation of the tangent hyperplanes of the corresponding hypersurface. The

polygonal lines are completely discarded because the *relation is concentrated in the pattern*: Linear relations into points, proximate coplanarity into convex polygons, quadrics into conics and so on. Note further, again with reference to Figs. 14.26 and 14.27, that relational information resides at the *crossings*. Continuing, with the representation of relations by patterns with the pictures of a helicoid and Moebius strip in cartesian and \parallel -coords in Figs. 14.32, 14.33. These are state of the art results showing

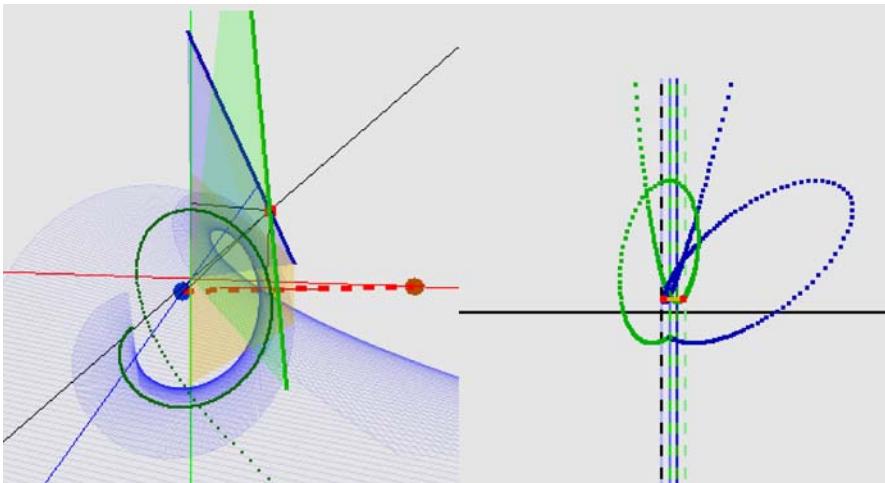


Figure 14.32. A 3-D helicoid in Cartesian and \parallel -coords. The two intersecting lines (left) specify one of the helicoid's tangent planes, as represented by a pair of points, one on each of the curves (right). A helicoid in N-D is represented by $N - 1$ such curves

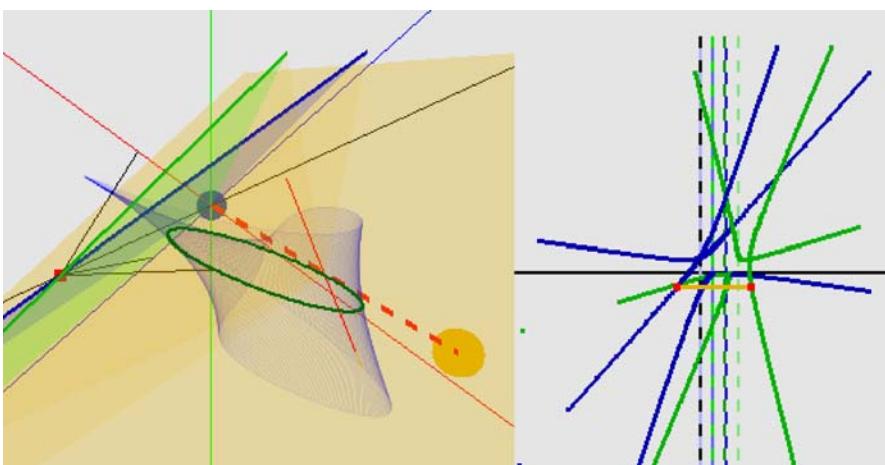


Figure 14.33. A 3-D Moebius strip in Cartesian and \parallel -coords. The two intersecting lines (left) specify one of the strip's tangent planes, as represented by a pair of points, one on each of the curves (right). A Moebius strip in N-D is represented by $N - 1$ such curves

what is achievable and how easily it generalizes to N-D. Can one imagine a higher dimensional helicoid much less a *non-orientable* surface like the Moebius strip. It is possible to do such a process on a dataset but at present this is computationally slow. The challenge is to speed up the algorithm for real-time response and break the gridlock of multidimensional visualization. There will still be work and fun for the multidimensional detectives visually separating and classifying the *no longer hidden* regions identifying complex multivariate relations. This is our vision for the, hopefully near, future.

Acknowledgement. I am indebted to the editors and referees for their careful corrections and helpful suggestions.

References

- Bollobas, B. (1979). *Graph Theory*. Springer, New York.
- Brodetsky, O.S. (1949). *A First Course in Nomography* (first published in 1920). G. Bell and Sons, London.
- Chatterjee, A. (1995). *Visualizing Multidimensional Polytopes and Topologies for Tolerances*. Ph.D. Thesis, Dept. Comp. Sci., USC, Los Angeles, CA.
- Chatterjee, A., Das, P.P. and Bhattacharya, S. (1993). Visualization in linear programming using parallel coordinates. *Pattern Recognition*, 26-11:1725–36.
- Chomut, T. (1987). *Exploratory Data Analysis in Parallel Coordinates*. M.Sc. Thesis, Dept. Comp. Sci, UCLA, Los Angeles, CA.
- Cluff, E., Burton, R.P. and Barrett, W. (1991). A survey and characterization of multidimensional presentation techniques. *Journal of Imaging Technology*, 17:142–53.
- Cohan, S.M. and Yang, D.C.H. (1986). Mobility analysis in parallel coordinates. *J. Mech. & Mach.*, 21:63–71.
- Desai, A. and Walters, L.C. (1991). Graphical representation of data envelopment analyses: management implications from parallel axes representations. *Dec. Scien.*, 22(2):335–353.
- d'Ocagne, M. (1885). *Coordonnees paralleles et axiale*. Gautier-Villars, Paris.
- d'Ocagne, M. (1899). *Traite de Nomographie*. Gautier-Villars, Paris.
- Eickemeyer, J. (1992). *Visualizing p-flats in N-space using Parallel Coordinates*. Ph.D. Thesis, Dept. Comp. Sci., UCLA, Los Angeles, CA.
- Fayad, U.M., Piatesky-Shapiro, G., Smyth, P. and Uthurusamy, R. (1996). *Advances in Knowledge Discovery and Data Mining*. AAAI/MIT Press.
- Fiorini, P. and Inselberg, A. (1989). Configuration Space Representation in Parallel Coordinates. *IEEE Conf. Rob. & Aut.*, pp. 1215–1220.
- Friendly, M. and et al. (2005) *Milestones in Thematic Cartography*. www.math.yorku.ca/scs/SCS/Gallery/milestones/.
- Gennings, C., Dawson, K.S., Carter, W.H. and Myers, R.H. (1990). Interpreting plots of a multidimensional dose-response surface in a parallel coordinate systems. *Biometrics*, 46:719–35.
- Deterding, D.H. (1989). *Speaker Normalization for Automatic Speech Recognition*. Ph.D. Thesis, Cambridge University.

- Harary, F. (1969). *Graph Theory*. Addison-Wesley, Reading, MA.
- Hauser, H. (2005). Parallel Sets: Visual Analysis of Categorical Data. *Proc. IEEE Infovis*.
- Helly, J. (1987). Applications of Parallel Coordinates to Complex System Design and Operation. *Proc. Nat. Comp. Grap. Assoc.* vol. III, pp. 541–546.
- Hinterberger, H. (1987). *Data Density: A Powerful Abstraction to Manage & Analyze Multivariate Data*. Informatik-Dissertation #4 ETH, Zurich.
- Hung, C.K. and Inselberg, A. (1992). *Parallel Coordinate Representation of Smooth Hypersurfaces*. USC Tech. Report USC-CS-92-531, Los Angeles, CA.
- Inselberg, A. (1980). N-Dimensional Coordinates. In *Proc. IEEE Conf. Pict. Data Desc.*, Asilomar, CA, p. 136. IEEE Comp. Soc., Los Alamitos, CA.
- Inselberg, A. (1981). *N-Dimensional Graphics, Part I – Lines and Hyperplanes*, IBM LASC Tech. Rep. G320-2711. IBM LA Scientific Center, Los Angeles, CA.
- Inselberg, A. (1984). Parallel Coordinates for Multidimensional Displays. *Proc. IEEE Conf. on Spatial Infor.*, Sioux Falls, SD, pp. 312–324. IEEE Comp. Soc., Los Alamitos, CA.
- Inselberg, A. (1985). Intelligent Instrumentation and Process Control. *Proc. 2nd IEEE Conf. on AI Appl.*, Miami Beach, FL, pp. 302–307. IEEE Comp. Soc., Los Alamitos, CA.
- Inselberg, A. (1985). The plane with parallel coordinates. *Visual Computer*, 1:69–97.
- Inselberg, A. (1989). Discovering Multi-Dimensional Structure with Parallel Coordinates. In *Proc. of ASA Conf. Stat. Grap.*, pp. 1–16. Amer. Stat. Assoc., Alexandria, VA.
- Inselberg, A. (1998). Visual data mining with parallel coordinates. *Comput. Statit.*, 13-1:47–64.
- Inselberg, A. (1999). Don't panic... do it in parallel! *Comput. Statit.*, 14:53–77.
- Inselberg, A. (2008). *Parallel Coordinates: VISUAL Multidimensional Geometry and its Applications*. Springer, New York.
- Inselberg, A., Boz, M. and Dimsdale, B. (1991). *Planar Conflict Resolution Algorithm for Air-Traffic Control and the One-Shot Problem*, in IBM PASC Tech. Rep. G320-3559. IBM Palo Alto Scientific Center, Palo Alto, CA.
- Inselberg, A. and Dimsdale, B. (1990). Parallel Coordinates: A Tool For Visualizing Multi-Dimensional Geometry. *Proc. of IEEE Conf. on Visualization*, pp. 361–378. IEEE Comp. Soc., Los Alamitos, CA.
- Inselberg, A., Reif, M. and Chomut, T. (1987). Convexity algorithms in parallel coordinates. *J. ACM*, 34:765–801.
- Inselberg, A. and Avidan, T. (1999). The Automated Multidimensional Detective. In *Proc. of IEEE Information Visualization '99*, pp. 112–119. IEEE Comp. Soc., Los Alamitos, CA.
- Jones, C. (1996). *Visualization and Optimization*. Kluwer, Boston, MA.
- Matskewich, T., Inselberg, A. and Bercovier, M. (2000). Approximated Planes in Parallel Coordinates. *Proc. of Geometric Model. Conf.*, St. Malo, Vanderbilt Univ. Press, Nashville, TN, pp. 257–266.
- Michie, D., Spiegelhalter, D.J. and Taylor, C.C. (1994). *Machine Learning, Neural and Statistical Classification*. AI Series, Ellis Horwood, New York.

- Mitchell, T.M. (1997). *Machine Learning*. McGraw-Hill, New York.
- Otto, E. (1963). *Nomography* (translated from 1956 Polish Edition by J. Smolska). MacMillan, New York.
- Quinlan, J.R. (1993). *C4.5 : Programs for Machine Learning*. Morgan Kaufman, San Mateo, CA.
- Rivero, J. and Inselberg, A. (1984). Extension al Analisis del Espacio De Fase de Sistemas Dinamicos por las Coordinadas Paralelas. *Proc. VII Systems Engr Workshop*, Santiago, Chile.
- Robinson, A.J. and Fallside, F. (1988). A Dynamic Connectionist Model of Phoneme Recognition. *Proc. of 1st European Neural Network Conf.* (nEURO).
- Schmid, C. and Hinterberger, H. (1994). Comparative Multivariate Visualization Across Conceptually Different Graphic Displays. In *Proc. of 7th SSDBM*. IEEE Comp. Soc., Los Alamitos, CA.
- Shang, N. and Breiman, L. (1996). *Distribution based trees are more accurate*. Proc. of 1st Inter. Conf. on Neural Info. Proc. (ICONIP96), 133.
- Tufte, E.R. (1983). *The Visual Display of Quantitative Information*. Graphic Press, Connecticut.
- Tufte, E.R. (1990). *Envisioning Information*. Graphics, Cheshire, CT.
- Tufte, E.R. (1996). *Visual Explanation*. Graphics, Cheshire, CT.
- Ward, M.O. (1994). XmdvTool: integrating multiple methods for visualizing multivariate data *Proc. IEEE Conf. on Visualization*, San Jose, CA, pp. 326–333. IEEE Comp. Soc., Los Alamitos, CA.
- Wegman, E. (1990). Hyperdimensional data analysis using parallel coordinates. *J. Amer. Stat. Assoc.*, 85:664–675.
- Yang, L. (2005). Pruning & visualizing generalized association rules in parallel coordinates. *IEEE Know. & Data Engr.*, 15(1):60–70.

Matrix Visualization

III.15

Han-Ming Wu, ShengLi Tzeng, Chun-Hou Chen

15.1	<i>Introduction</i>	682
15.2	<i>Related Works</i>	682
15.3	<i>The Basic Principles of Matrix Visualization</i>	683
	Presentation of the Raw Data Matrix	684
	Seriation of Proximity Matrices and the Raw Data Matrix.....	686
15.4	<i>Generalization and Flexibility</i>	690
	Summarizing Matrix Visualization.....	690
	Sediment Display.....	691
	Sectional Display	692
	Restricted Display	692
15.5	<i>An Example</i>	693
15.6	<i>Comparison with Other Graphical Techniques</i>	697
15.7	<i>Matrix Visualization of Binary Data</i>	700
	Similarity Measure for Binary Data	700
	Matrix Visualization of the KEGG Metabolism Pathway Data	702
15.8	<i>Other Modules and Extensions of MV</i>	704
	MV for Nominal Data.....	704
	MV for Covariate Adjustment.....	704
	Data with Missing Values.....	705
	Modeling Proximity Matrices.....	705
15.9	<i>Conclusion</i>	705

Introduction

The graphical exploration of quantitative/qualitative data is an initial but essential step in modern statistical data analysis. Matrix visualization (Chen, 2002; Chen et al., 2004) is a graphical technique that can simultaneously explore the associations between thousands of subjects, variables, and their interactions, without needing to first reduce the dimensions of the data. Matrix visualization involves permuting the rows and columns of the raw data matrix using suitable seriation (reordering) algorithms, together with the corresponding proximity matrices. The permuted raw data matrix and two proximity matrices are then displayed as matrix maps via suitable color spectra, and the subject clusters, variable groups, and interactions embedded in the dataset can be extracted visually.

Since the introduction of exploratory data analysis (EDA, Tukey, 1977), boxplots and scatterplots, aided by interactive functionality, have provided the statistical community with important graphical tools. These tools, together with various techniques for reducing dimensions, are useful for exploring the structure of the data when there are a moderate number of variables and when the structure is not too complex. However, with the recent rapid advances in computing, communications technology, and high-throughput biomedical instruments, the number of variables associated with the dataset can easily reach tens of thousands, but the need for practical data analysis remains. Dimension reduction tools often become less effective when applied to the visual exploration of information structures embedded in high-dimensional datasets. On the other hand, matrix visualization, when integrated with computing, memory, and display technologies, has the potential to enable us to visually explore the structures that underlie massive and complex datasets.

This chapter on matrix visualization unfolds as follows. We briefly review studies in this field in the next section. The foundation of matrix visualization, under the framework of generalized association plots (GAP, Chen, 2002), is then discussed in Sect. 15.3, along with some related issues. This is followed, in Sect. 15.4, by some generalization. Section 5 provides a matrix visualization examples involving 400 variables (arrays) and 2000 samples (genes). A comparison of matrix visualization with other popular graphical tools in terms of efficiency versus number of dimensions is then given in Sect. 15.6. Section 15.7 illustrates matrix visualization for binary data, while Sect. 15.8 discusses generalizations and extensions. We conclude this chapter with some perspectives on matrix visualization in Sect. 15.9.

Related Works

The concept of matrix visualization was introduced by Bertin (1967) as a reorderable matrix for systematically presenting data structures and relationships. Carmichael and Sneath (1969) developed taxometric maps for classifying OUTs (operational taxonomy units) in numerical phenetics analysis. Hartigan (1972) introduced the direct clustering of a data matrix, later known as block clustering (Tibshirani, 1999). Lenstra

(1974) and Slagle et al. (1975) related the traveling salesman and shortest spanning path problems to the clustering of data arrays. The color histogram of Wegman (1990) was the first color matrix visualization to be reported in the statistical literature. Minnotte and West (1998) extended the idea of color histograms to a data image package that was later used for outlier detection (Marchette and Solka, 2003).

Some matrix visualization techniques were developed to explore only proximity matrices: Ling (1973) looked for factors of variables by examining relationships using a shaded correlation matrix; Murdoch and Chow (1996) used elliptical glyphs to represent large correlation matrices; Friendly (2002) proposed corrgrams (similar to the reorderable matrix method) to analyze multivariate structure among the variables in correlation and covariance matrices. Chen (1996, 1999, and 2002) integrated the visualization of a raw data matrix with two proximity matrices (for variables and samples) into the framework of generalized association plots (GAP). The Cluster and TreeView packages of Eisen et al. (1998) are probably the most popular matrix visualization packages due to the proliferation of gene expression profiling for microarray experiments.

The permutation (ordering) of the columns and rows of a data matrix, and proximity matrices for variables and samples, is an essential step in matrix visualization. Several recent statistical works have touched on the issue of reordering variables and samples: Chen (2002) proposed the concept of the relativity of a statistical graph; Friendly and Kwan (2003) discussed the idea of effect-ordering of data displays; Hurley (2004) used scatterplot matrices and parallel coordinates plots as examples to address the issue of placing interesting displays in prominent positions. Different terms (such as the reorderable matrix, the heatmap, the color histogram, the data image and matrix visualization) have been used in the literature to describe these related techniques. We use matrix visualization (MV) to refer to them all.

The Basic Principles of Matrix Visualization

15.3

We use the GAP (Chen, 2002) approach to illustrate the basic principles of matrix visualization for continuous data, using the 6400 genes and 851 microarray experiments collected in the published yeast expression database for visualization and data mining (Marc et al., 2001), which is designated henceforth as Dataset 0. Detailed descriptions of data preprocessing are given for the yeast Microarray Global Viewer (<http://transcriptome.ens.fr/ymgv/>). For the purposes of illustration, we have selected 15 samples and 30 genes across these samples (“Dataset 1”), where rows correspond to genes and columns to microarray experiments (arrays). In various gene expression profile analyses, the roles played by rows and columns are often interchangeable. This interchangeability is well suited to the GAP approach to matrix visualization, where samples and variables are treated symmetrically and can be interchanged directly.

Presentation of the Raw Data Matrix

The first step in the matrix visualization of continuous data is the production of a raw data matrix $X_{30 \times 15}$, and two corresponding proximity matrices for the rows, $R_{30 \times 30}$, and the columns, $C_{15 \times 15}$, which are calculated with user-specified similarity (or dissimilarity) measures. The three matrices are then projected through suitable color spectra to construct corresponding matrix maps in which each matrix entry (raw data or proximity measurement) is represented by a color dot. The left panel in Fig. 15.1 shows the raw data matrix of \log_2 -transformed ratios of expressions coded by a bidirectional green–black–red spectrum for Dataset 1, with Pearson correlations for between-array relations coded by a bidirectional blue–white–red spectrum, and Euclidean distances for between-gene relations coded by a unidirectional rainbow spectrum.

In the raw data matrix map, a red (green) dot in the ij th position of the map for $X_{30 \times 15}$ means that the i th gene at the j th array is relatively up (down)-regulated. A black dot stands for a relatively nondifferentially expressed gene/array combination. A red (blue) point in the ij th position of the $C_{15 \times 15}$ matrix map represents a positive (negative) correlation between arrays i and j . Darker (lighter) intensities of color stand for stronger absolute correlation coefficients, while white dots represent no correlations. A blue (red) point in the ij th position of the $R_{30 \times 30}$ matrix map represents a relatively small (large) distance between genes i and j , while a yellow dot represents a median distance.

Data Transformation

It may be necessary to apply transformations such as log, standardization (zero mean, unit variance), or normalization (normal score transformation) to the raw data before the data map is constructed or proximity matrices calculated in order to get a meaningful visual representation of the data structure, or comparable visual effects between displays. The transformation–visualization process may have to be repeated several times before the embedded information can be fully explored.

Selection of Proximity Measures

Proximity matrices have two major functions: (1) to serve as the direct visual representation of the relationships among variables and between samples; (2) to serve as the medium used to reorder the variables and samples for better visualization of the three matrix maps. The selection of proximity measures in matrix visualization plays a more important role than it does in numerical or modeling analyses. Pearson correlation often serves as the measure of proximity between variables, while Euclidean distance is commonly employed for samples (Fig. 15.1). For potential non-linear relationships, Spearman's rank correlation and Kendall's tau coefficient can be used instead of the Pearson correlation to assess the between-variable relationship, while some nonlinear feature extraction methods such as the Isomap (Tenenbaum et al., 2000) distance can be used to measure nonlinear between-sample distances. More sophisticated kernel methods can also be applied when users see the need for them.

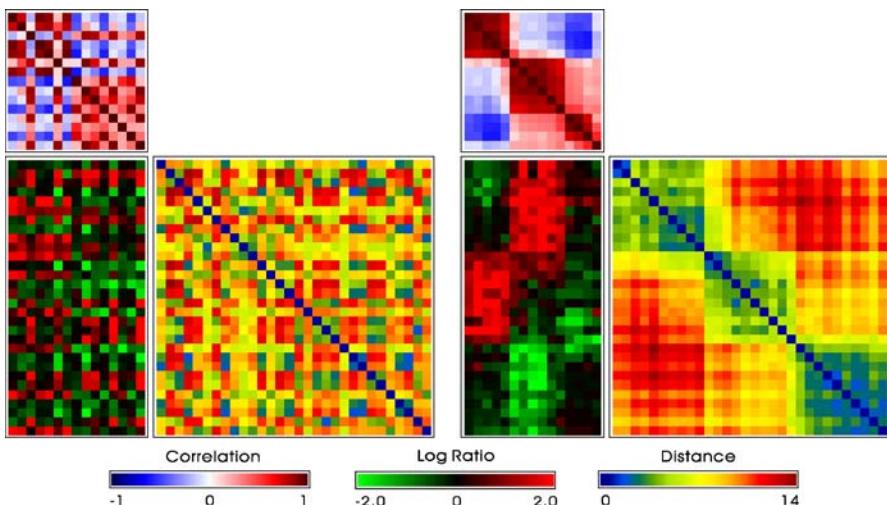


Figure 15.1. [This figure also appears in the color insert.] *Left:* unsorted data matrix (log ratio gene expression) map with two proximity matrix (Pearson correlation for arrays and Euclidean distance for genes) maps for Dataset 1. *Right:* application of elliptical seriations to the three matrix maps on the left panel

Color Spectrum

The selection of an appropriate color spectrum can be critical and is user-dependent in visualization of and information extraction from data and proximity matrices. The selection of a suitable color spectrum should focus on the capacity to express numerical nature individually and globally in the matrices. The choices for gene expression profiles that we mentioned above may well give way to others in different circumstances. Thus, illustrated in Fig. 15.2 is a correlation matrix map of fifty psychosis disorder variables (Chen, 2002) coded with four different bidirectional color spectra. While displays (a) and (b) appear more agreeable to our human perception, displays (c) and (d) actually provide better resolution for distinguishing different levels of correlation intensities. The relative triplet color codes (red, green, blue) in the RGB cube for these four color spectra are shown in Fig. 15.3.

Display Conditions

The display conditions are analogous to data transformations for colors. Usually, the whole color spectrum is used to represent the complete range of values in the data matrix. The matrix conditions can be switched to row or column conditions to emphasize individual variable distributions or subject profiles. For a bidirectional color spectrum (green–black–red for differential gene expressions, blue–white–red for correlation coefficients), the center matrix condition symmetrizes the color spectrum around the baseline numeric value (1:1 for log₂ ratio gene expression, zero for the correlation coefficient). On occasion, we might want to downweight the effects of extreme values in the dataset, and it is possible to use ranks as a replacement for numerical values. This is termed the rank matrix condition.

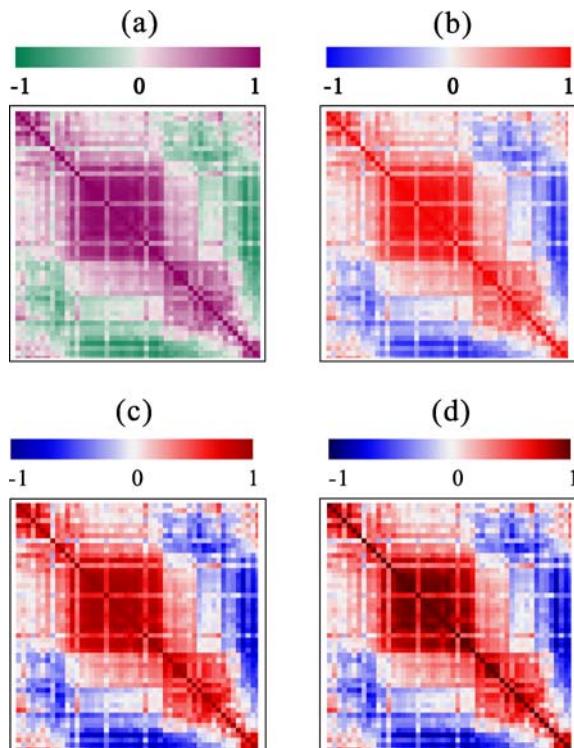


Figure 15.2. [This figure also appears in the color insert.] Four color spectra applied to the same correlation matrix map for fifty psychosis disorder variables (Chen, 2002)

Resolution of a Statistical Graph

If the data matrix or proximity matrices contain potential extreme values, the relative structure of these extreme values compared to the main data cloud will dominate the overall visual perception of the raw data map and the proximity matrix maps. This problem can be handled by using rank conditions or by compressing the color spectrum to a suitable range. We can apply a logarithm or similar transformation to reduce the outlier effect or to simply remove the outlier.

Seriation of Proximity Matrices and the Raw Data Matrix

Without suitable permutations (orderings) of the variables and samples, matrix visualization is of no practical use for visually extracting information (Fig. 15.1, left panel). It is necessary to compute meaningful proximity measures for variables and samples, and to apply suitable permutations to these matrices, before matrix visualization is used to reveal the information structure of the given dataset. We discuss some con-

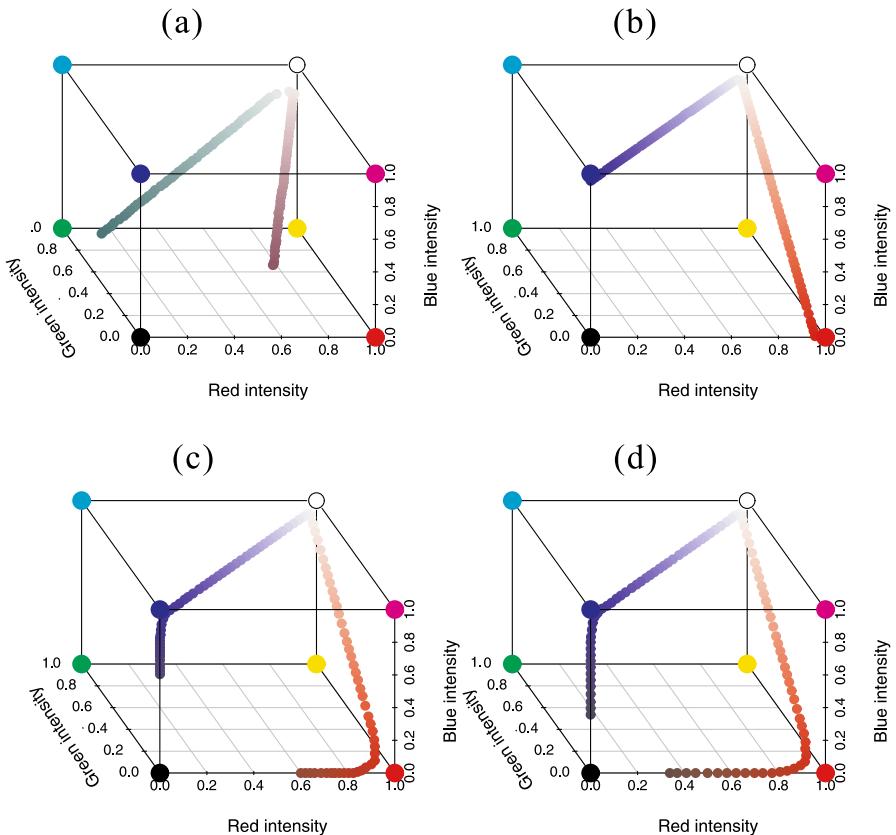


Figure 15.3. Relative (red, green, blue) hues in the RGB cubes for the four color spectra in Fig. 15.2

cepts and criteria for evaluating the performances of different seriation algorithms in reordering related matrices below.

Relativity of a Statistical Graph

Chen (2002) proposed a concept, the relativity of a statistical graph, for evaluating general statistical graphic displays. The idea is to place similar (different) objects at closer (more distant) positions in a statistical graph. In a continuous display, such as the histogram or a scatterplot, relativity always holds automatically. This is illustrated by the histogram of the Petal Width variable and the scatterplot of the Petal Width and Petal Length variables for 150 Iris flowers shown in Fig. 15.4 (Fisher, 1936). Two flowers, denoted with the \times and \circ symbols, are placed next to each other on these two displays automatically, because they share similar petal widths and lengths. Friendly and Kwan (2003) proposed a similar concept to order information in general visual displays, which they called the effect-ordered data display. Hurley (2004) also studied related issues using examples involving scatterplot matrices and parallel coordinates plots.

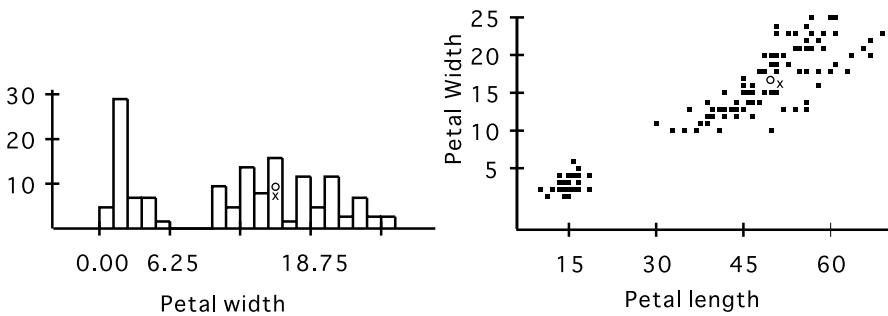


Figure 15.4. Concept of the relativity of a statistical graph for a continuous dataset (the Iris data)

The relativity concept does not usually hold for a matrix visualization or parallel coordinates type of display, since one can easily destroy the property with a random permutation. It is common practice to apply various permutation algorithms to sort the columns and rows of the designated matrix, so that similar (different) samples/variables are permuted to make them closer (more distant) rows/columns.

Global Criterion: Robinson Matrix

It is usually desirable to permute a matrix to make it resemble a Robinson matrix (Robinson, 1951) as closely as possible, because of the smooth and pleasant visual effect of permuted matrix maps. A symmetric matrix is called a Robinson matrix if its elements satisfy $r_{ij} \leq r_{ik}$ if $j < k < i$ and $r_{ij} \geq r_{ik}$ if $i < j < k$. If the rows and columns of a symmetric matrix can be permuted to those of a Robinson matrix, we call it pre-Robinson. For a numerical comparison, three anti-Robinson loss functions (Streng, 1978) are calculated for each permuted matrix, $D = \{d_{ij}\}$, for the amount of deviation from a Robinson form with distance-type proximity:

$$\begin{aligned} AR(i) &= \sum_{i=1}^p \left[\sum_{j < k < i} I(d_{ij} < d_{ik}) + \sum_{i < j < k} I(d_{ij} > d_{ik}) \right], \\ AR(s) &= \sum_{i=1}^p \left[\sum_{j < k < i} I(d_{ij} < d_{ik}) \cdot |d_{ij} - d_{ik}| + \sum_{i < j < k} I(d_{ij} > d_{ik}) \cdot |d_{ij} - d_{ik}| \right], \\ AR(w) &= \sum_{i=1}^p \left[\sum_{j < k < i} I(d_{ij} < d_{ik}) |j - k| |d_{ij} - d_{ik}| + \sum_{i < j < k} I(d_{ij} > d_{ik}) |j - k| |d_{ij} - d_{ik}| \right]. \end{aligned}$$

$AR(i)$ counts only the number of anti-Robinson events in the permuted matrix; $AR(s)$ sums the absolute values of the anti-Robinson deviations; $AR(w)$ is a weighted version of $AR(s)$ penalized by the difference in the column indices of the two entries.

Elliptical Seriation

Chen (2002) introduced a permutation algorithm called rank-two elliptical seriation that extracts the elliptical structure of the converging sequence of iteratively formed

correlation matrices using eigenvalue decomposition. Given a p -dimensional proximity matrix D , a sequence of correlation matrices $R = (R^{(1)}, R^{(1)}, \dots)$ is iteratively formed from it. Here $R^{(1)}$ is the correlation matrix of the original proximity matrix D , and $R^{(n)}$ is the correlation matrix of $R^{(n-1)}$ for $n > 1$. The iteratively formed sequence of correlation matrices gradually cumulates the variation information to the leading eigenvectors. At the iteration with rank two, there are only two eigenvectors left with nonzero eigenvalues, and all information is reduced to the ellipse spanned by the two eigenvectors. Every object has its relative position on this two-dimensional ellipse, and a unique permutation is obtained. Elliptical seriation usually identifies very good global permutations, and is useful for identifying global clustering patterns and smooth temporal gene expression profiles (Tien et al., 2006) by optimizing the Robinson criterion.

Local Criterion: Minimal Span Loss Function

The minimal span loss function $MS = \sum_{i=1}^{n-1} d_{i,i+1}$ for a permuted matrix $D = \{d_{ij}\}$ focuses on the optimization of local structures. The idea is to find a shortest path through all data elements, as in the traveling salesman problem. The local seriation method produces tighter blocks than the global method does around the main diagonal of the proximity matrix. In addition, we can combine the anti-Robinson measure and minimal span loss into a measure in which a band along the diagonal of a proximity matrix is selected with width w ($0 < w < n$), and the anti-Robinson measurement is computed within that band.

Tree Seriation

The hierarchical clustering tree with a dendrogram (Eisen et al., 1998) is the most popular method for two-way sorting the gene-by-array matrix map employed in gene expression profiling. The ordering of terminal nodes generated by an agglomerative hierarchical clustering tree automatically keeps good local grouping structure, since the tree dendrogram is constructed through a sequential bottom-up merging of “most similar” subnodes. On the other hand, a divisive hierarchical clustering tree usually retains better global patterns through a top-down splitting of “most heterogeneous” substructures. Divisive hierarchical clustering trees are rarely used due to their computational complexity.

Flipping of Intermediate Nodes

One critical issue when applying the leaves of the dendrogram in order to sort the rows/columns of an expression profile matrix is the flipping of the intermediate nodes. As illustrated in Fig. 15.5 with a schematic dendrogram (Fig. 15.5a), the $n - 1$ intermediate nodes for a dendrogram of n objects can be flipped independently (Fig. 15.5b), resulting in $2^{(n-1)}$ different dendrogram layouts (Figure 15.5c, for example) and corresponding permutations for the n objects with identical proximity matrices (Pearson correlation or Euclidean distance) and the same tree linkage method (single, complete, average or centroid). The flipping mechanism of intermediate nodes can be guided by either an external or an internal reference list. For example, the Cluster

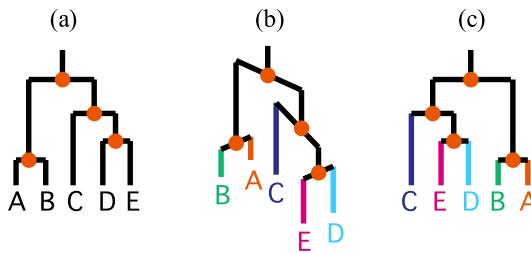


Figure 15.5. Flipping mechanism for intermediate nodes of a dendrogram

software developed by Eisen's lab (1998) guides the tree flips based on the average expression level. He also suggests that one can use the results of a one-dimensional self-organizing map (SOM, Kohonen, 2001) to guide the tree seriation. This makes the tree seriation as close to the external references as possible. In Alon et al. (1999), it is suggested that one should order the leaf nodes according to the similarity between a node and its parent's siblings. Bar-Joseph (2001) proposed a fast optimal leaf ordering method for hierarchical clustering that maximizes the sum of the similarities of adjacent leaves in the ordering. These are two examples of internal references.

15.4 Generalization and Flexibility

15.4.1 Summarizing Matrix Visualization

Sorted matrix maps are capable of displaying the raw expression patterns and the association structures among genes and arrays. One can go one step further and identify clusters in the permuted matrix maps using the dendrogram branching structure or other partitioning methods, such as the converging sequence of Pearson's correlation matrices (Chen, 2002) and block searching (Hartigan, 1972). Once the partitioned matrix maps are obtained (Fig. 15.6, left panel), a summarizing matrix visualization which Chen (2002) coined "sufficient matrix visualization" can be constructed by representing individual data points and proximity measures in each identified subject–subject, variable–variable and subject–variable block by the summary statistic (means, medians or standard deviations) for that particular block.

The three maps in Fig. 15.6, right panel, summarize the sufficient information of the data matrix, and the corresponding proximity matrices for the gene expression profiles are shown in the left panel. In the sufficient MV of Fig. 15.6, right panel, users can easily extract the within and between correlation structure for the three array groups, the relative clustering patterns of the four gene clusters, and the interaction behavior of the four gene clusters on the three array groups. There are three requirements for ensuring the effectiveness of a sufficient MV at extracting the overall information structure embedded in the original data matrix and two proximity matrices: (1) appropriate permuted variables and samples; (2) carefully derived partitions for variables and samples, and; (3) representative summary statistics.

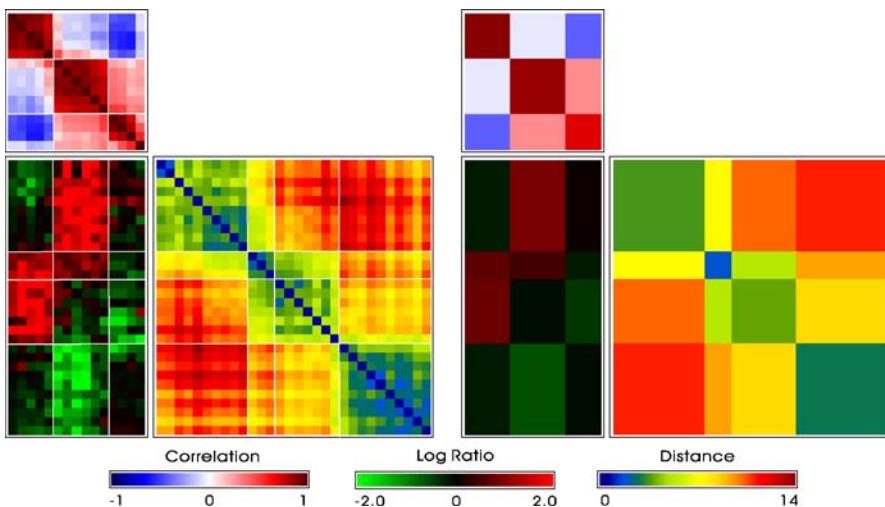


Figure 15.6. Left: partitioned data and proximity matrix maps for Dataset 1. Right: sufficient data and proximity matrix maps

Sediment Display

15.4.2

The sediment display of a row data matrix for rows (columns) is constructed by sorting the column (row) profiles for each row (column) independently according to their magnitudes. This display expresses the distribution structure for all rows (columns) simultaneously. The middle panel of Fig. 15.7 has the sediment display for all 30 gene expression profiles, while the right panel has the expression distributions for each of the 15 selected arrays. The sediment displays for genes and arrays convey similar information to that given by a boxplot when the color strips at the quartile positions are extracted.

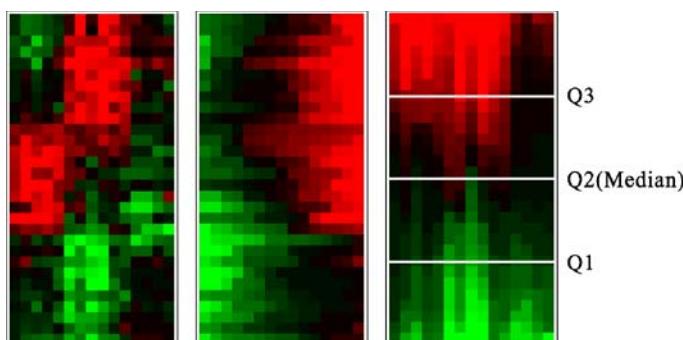


Figure 15.7. Sediment displays for genes (middle panel) and arrays (right panel) for the permuted data matrix (left panel) of Dataset 1

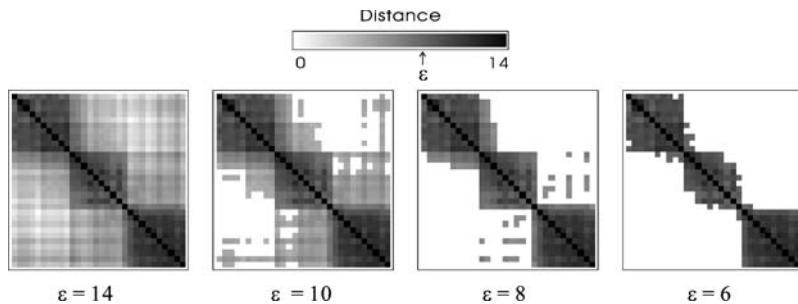


Figure 15.8. Sectional displays for the permuted gene distance map. Only distances smaller than the threshold, ϵ , are displayed

15.4.3

Sectional Display

The purpose of a sectional display is to exhibit only those numerical values that satisfy certain conditions in the data or the associated proximity maps. For example, one can choose to ignore the values below some threshold by not displaying the corresponding color dots. For a permuted distance map, one can emphasize more coherent neighboring structure by displaying only the corresponding neighbors dynamically. Figure 15.8 shows a series of such sectional displays (in grey scale) for the distance matrix for the genes in Fig. 15.1, right panel (and Fig. 15.6, left panel).

15.4.4

Restricted Display

Outlying data points or proximity measures can mask detailed color resolutions. This situation can be improved by displaying only rank conditions instead of original magnitudes, or by compressing the color spectrum to represent only the main body of the data values, i.e., one displays data values that fall within some range of the data using the whole color spectrum. Figure 15.9, left panel, shows a restricted display of Fig. 15.8 with an artificial outlier observation added. The relatively large distance of this outlier from the other observations causes the color spectrum to mask the main

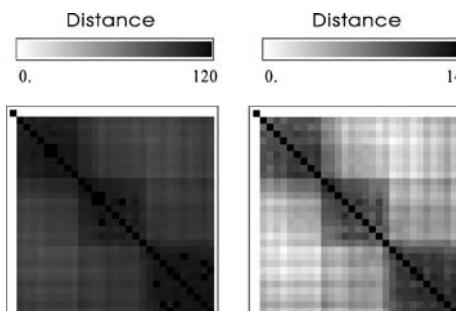


Figure 15.9. Original (left) and restricted (right) displays for the permuted gene distance map in Fig. 15.8 with an outlying gene added

feature embedded in the distance matrix. The right panel of Fig. 15.9 uses the whole grey spectrum to represent the distance range of 0–14 only, which reveals the main three-group structure. The use of nonlinear color mapping (for the distances), like the one implemented in MANET (Unwin, 1998), can also resolve this problem.

An Example

Construction of an MV Display

Many of the microarrays in Dataset 0 have lots of missing values due to technical issues and because different experiments studied different sets of genes in the yeast genome. Two thousand genes with four hundred arrays that had relatively few missing values were then selected from the original Dataset 0, resulting in Dataset 2. Illustrated in Fig. 15.10 is the MV display of Dataset 2. Pearson's correlation coefficient is used to measure associations between genes and between arrays, a common practice in gene expression profile analysis. Average linkage clustering trees are then grown on the two correlation matrices for genes and arrays. The relative positions of the terminal nodes of the two dendrograms are then used to sort the corresponding correlation matrix maps and the data matrix map (the gene expression profile). The basic gene clustering structure and array (experiments) grouping patterns can be identified using these tree-sorted matrix maps.

The enlarged, permuted data matrix map used for gene expression profiling is displayed in Fig. 15.11. Red dots represent a relatively high expression of message RNA for the gene-experiment combination, green dots indicate relatively low expression, and black dots designate relatively little differential expression. Missing values are coded in white, it is clear that many of the arrays (experiments) still contain some missing observations. Such an MV display presents each gene expression profile as a horizontal strip of color dots across all arrays (experiments), and the important visual information is carried by the relative variations in color hues.

Without suitable permutations that sort rows of similar genes so that they are close together and place identical arrays next to each other, so that the relativity property holds, an MV display is basically useless. Based on this two-way permuted display, one looks for horizontal strips of genes that share similar expression profiles, and vertical strips of arrays that exhibit close experimental results. The blocks of the two directions illustrate the interaction patterns of gene clusters and experiment groups. All of the numerical information is displayed in this raw expression profile map (with proximity maps for genes and arrays and corresponding dendrograms). Careful and patient examination of these color maps can lead to valuable insights into the embedded information structure.

Examination of an MV Display

As with other visualization tools, both proper training and experience are required to extract as much information out of these complex matrix visualization displays as possible. While examining complex MV displays such as those shown in Figs. 15.10 and 15.11, several general steps should be taken:

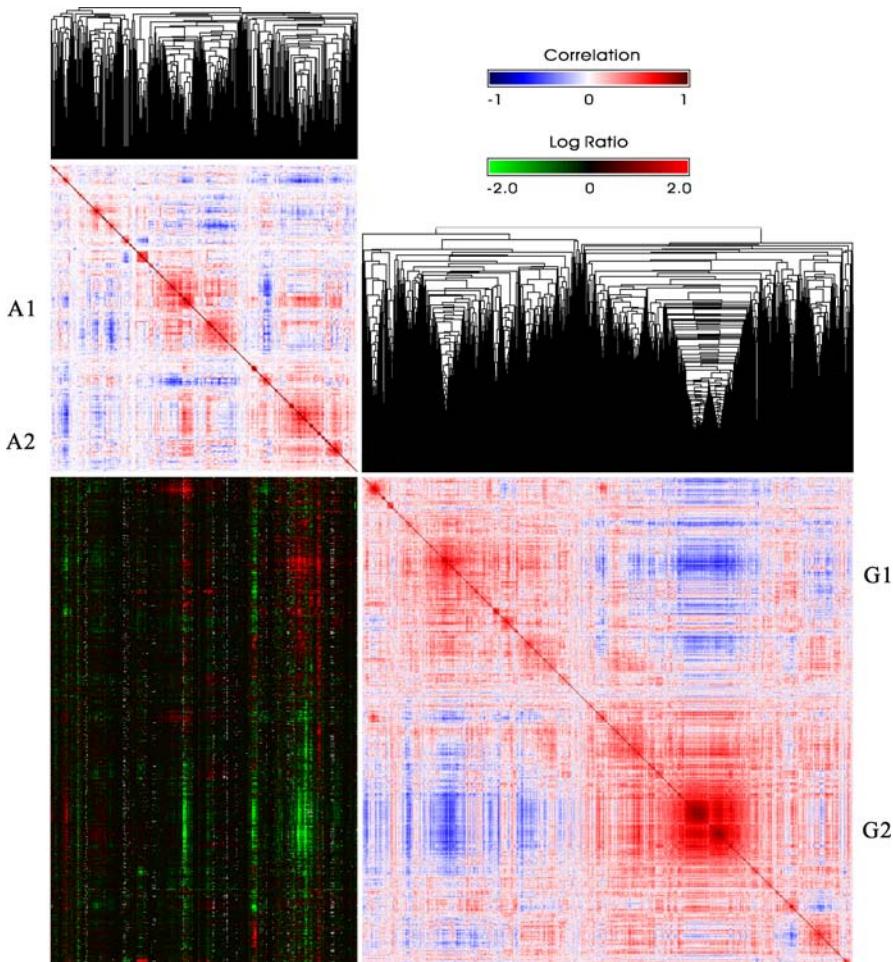


Figure 15.10. [This figure also appears in the color insert.] Data matrix map (\log_2 ratio gene expression) with two proximity matrix maps (Pearson correlation for both genes and arrays) for Dataset 2 permuted by two average linkage trees (for genes (rows) and arrays (columns))

1. For the column (array) proximity matrix:
 - a) Search for coherent clusters of arrays along the main diagonal of the correlation (maybe distance in other circumstances) matrix with dark red points. Two dominant groups of arrays can be identified around the middle and the lower-right corner of the correlation matrix, with several small but coherent clusters scattered along the main diagonal. Let's denote these two major groups of arrays as A1 and A2. The arrays grouped into these clusters must have similar expression patterns across all 2000 genes (which will be examined in later steps).

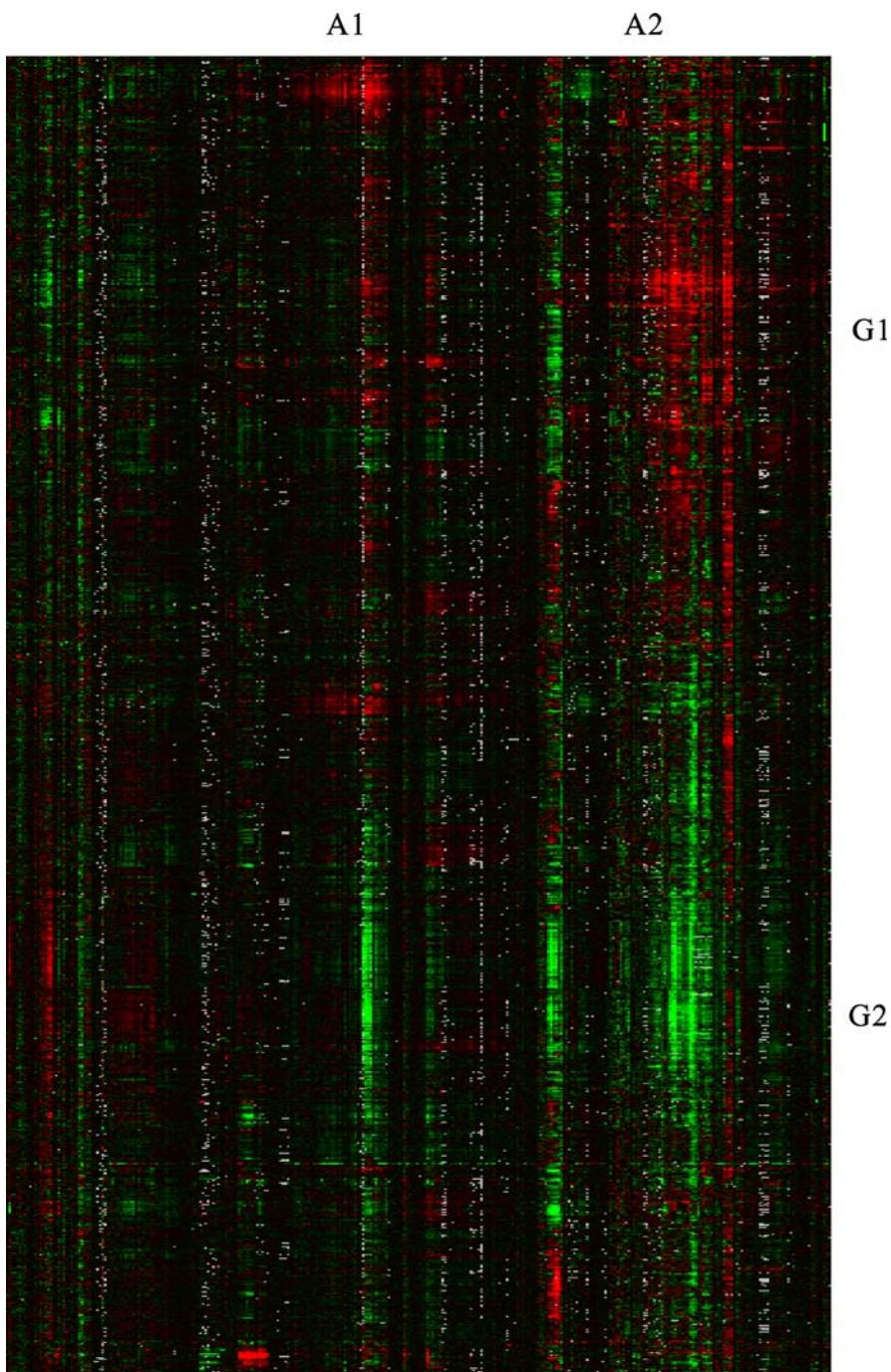


Figure 15.11. Enlarged expression profile matrix map of Fig. 15.10 (missing observations are coded in white)

- b) Look for interactions between the array clusters at off-diagonal locations. Various types of between-cluster correlation patterns with substructure are also easily pinned down.
 - c) The arrays represent many different biological assays for various functions of *Saccharomyces cerevisiae* yeast, such as cell-cycle control, stress (environmental changes, relevant drug-affected), metabolic/genetic control, transcriptional control and DNA binding (<http://transcript-ome.ens.fr/ymgv/>). Different biomedical assays activate and suppress expression patterns of certain functional groups of genes. We need to integrate this biological/medical knowledge with the numerical/graphical findings in 1a and 1b to validate known information and more importantly to explore and interpret novel interesting patterns.
 - d) Hierarchical clustering trees for arrays and genes also yield a partial visual exploration of the data and proximity structure, but this is not as comprehensive as direct visualization of the two proximity matrix maps, since the dendograms only retain some of the information in the proximity matrices from which they are constructed.
2. For the row (gene) proximity matrix:
 - a) Similar procedures to those described in 1a and 1b for arrays (columns) must be repeated for the gene (row) proximity matrix. Of particular interest is the dichotomous pattern of these 2000 genes. The up-regulated (red) genes at the upper half and the down-regulated (green) genes at the bottom half of the A2 arrays are responsible for this dichotomous structure. We denote these two clusters of genes as G1 and G2 here. Several small subclusters of genes within G1 and G2 can also be identified along the main diagonal.
 - b) It is necessary to go one step further and consult various annotation databases for more detailed interpretations of and explanations for the potential clusters of genes identified this way. Some of the genes have not been annotated yet. Their potential functions can be roughly determined through the positively correlated (up-regulated) gene clusters and the negatively correlated (down-regulated) patterns.
3. For the raw data (gene expression profile) matrix:
 - a) Many major and minor array groups and gene clusters were found in steps 1 and 2. In step 3, we use the raw data (gene expression profile) matrix map to search for the interaction patterns of the gene clusters on the array groups. It is also necessary to use vertical strips of expression profiles to contrast between array group structure variations and horizontal strips to distinguish between gene cluster distribution differences. With careful examination, one can associate certain blocks of expression in the raw data matrix with the formation of each array group and gene cluster and the between-group (cluster) differentiation.
 - b) There are about 10 000 (~1.25 %) missing observations in this data matrix of 400 arrays with 2000 genes. It is clear that the pattern of missing observa-

- tions is not a random one. Different array group and gene cluster combinations have different proportions of missing observations. The visualization of the missing structure is a great aid to users when they attempt to choose a more appropriate missing value estimate or imputation mechanism for further analyses.
- c) Visual exploration provides valuable insights into more advanced studies, such as the confirmation of existing metabolite pathways (see Sect. 7 for an example) and the exploration of novel pathways.

This paragraph has only discussed some general issues associated with examining an MV display. If we have access to expert knowledge and biologists familiar with experiments related to *Saccharomyces cerevisiae* yeast, there are actually many more interesting patterns that can be explored. In Figs. 15.10 and 15.11 we demonstrated an MV can easily handle thousands of samples. An MV display can also handle thousands of variables, since samples and variables are treated symmetrically in the MV framework.

Comparison with Other Graphical Techniques

15.6

In this section, we compare the visualization efficiencies of the scatterplot (SP), the parallel coordinates plot (PCP) (Inselberg, 1985; Wegman, 1990), and matrix visualization (MV), based on varying dimensionality of the dataset.

Low-Dimensional Data

For one-dimensional data, scatterplot and the PCP displays amount to dotplots, while a one-dimensional MV yields a colored bar chart. In any event, it is unlikely that any method of displaying one-dimensional data will prove more popular than the histogram. A scatterplot is the most efficient graphical display for two-dimensional data. While a PCP relies on the n connecting line segments between two vertical dotplots to represent the association between the two variables, MV displays each sample as a single row with two colored dots. The efficiency of scatterplots decreases as dimension increases. For three-dimensional data, a rotational scatterplot is commonly used to extract geometric structure by viewing a sequence of two-dimensional scatterplots over a range of angles controlled by the user. The usefulness of PCP and MV displays of three-dimensional data is a subtle point, and the best permutation of variables is definitely needed to enforce relativity for both types of displays.

High-Dimensional Data

A scatterplot matrix (SM) is used to simultaneously visualize the information structure embedded in all $C(p, 2)$ pairs of variables for data with more than three di-

mensions. Grand tours (Asimov, 1985) are sometimes undertaken in the hope of extracting high-dimensional data structure by rotating randomly projected three-dimensional plots. Dimension reduction techniques, such as principal component analysis, are also useful for displaying structural information from high-dimensional data in low-dimensional displays. Figure 15.12 shows a scatterplot matrix display of the first 30 variables (arrays) in Dataset 2, while Fig. 15.13 gives the corresponding PCP for these data. We note that a PCP of high-dimensional data with a large sample size can simultaneously display all of the samples, but it is usually necessary to use some interactive mechanism to select subsets of samples in order to study the relative structure across all variables, as in Fig. 15.13. Moreover, for these plots, more than one pixel width is needed to display each variable.

In general, a scatterplot matrix needs $C(p, 2) \times n$ dots to display a dataset with n samples measured on p variables, a PCP needs p vertical lines plus $(p - 1) \times n$ line segments, and an MV plot requires $n \times p$ dots. When p becomes large, larger than 15

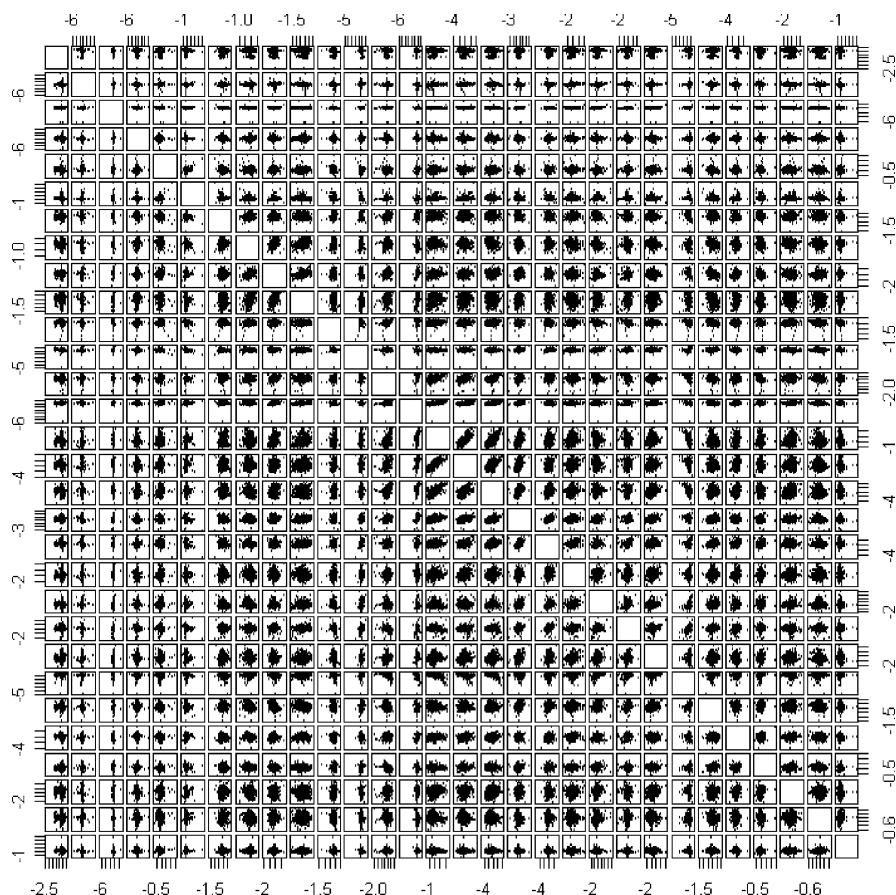


Figure 15.12. Scatterplot matrix for the first thirty arrays of Dataset 2

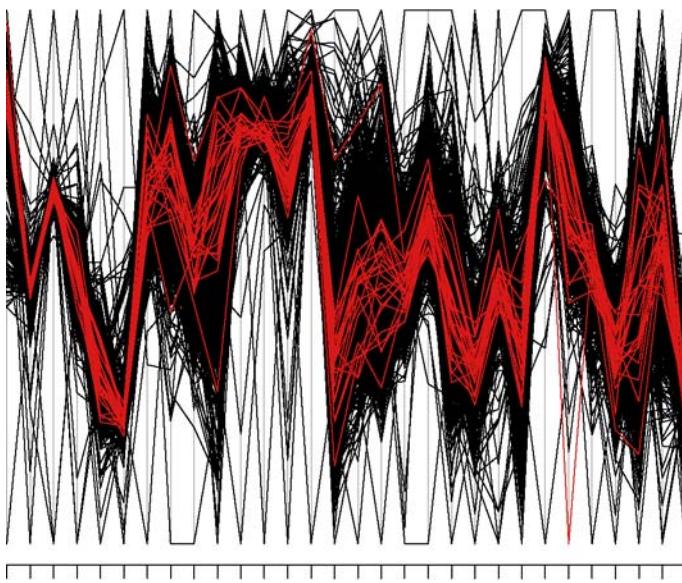


Figure 15.13. Parallel coordinates plot for the first thirty arrays of Dataset 2

for instance, a scatterplot matrix is basically useless. A PCP display does well for up to a few hundred variables, but founders for more due to the space required to display the line segments that connect sample points. A scatterplot matrix also wastes a high proportion of the display space. An MV display, on the other hand, utilizes every column pixel to display a variable on a computer screen. PCP has an advantage over MV on the sample side, but MV plots provide better resolution.

Overall Efficiency

Figure 15.14 is a diagram of efficiency against dimensionality for a conventional scatterplot (matrix) and dimension-free visualization tools such as the parallel coordinates plot (PCP) and matrix visualization (MV). While direct visual perception of the geometric pattern makes scatterplots the most efficient type of display for visualizing low-dimensional data, MV and PCP are definitely better for visualizing datasets with fifteen or more variables.

Missing Values

It is very difficult to display missing values in a scatterplot, while it is always possible to display missing values above or below the regular data range of each variable in a PCP display. The MANET system by Unwin et al. (1996) can be used to display missing information interactively. In an MV plot, a missing value can be simply displayed at the corresponding position (row and column) with a color that can be easily distinguished from the color spectrum of the numerical values. The missing values of the gene expression profiles in Figs. 15.10 and 15.11 are coded in white. With appropriate permutations for rows and columns, the corresponding variable-sample

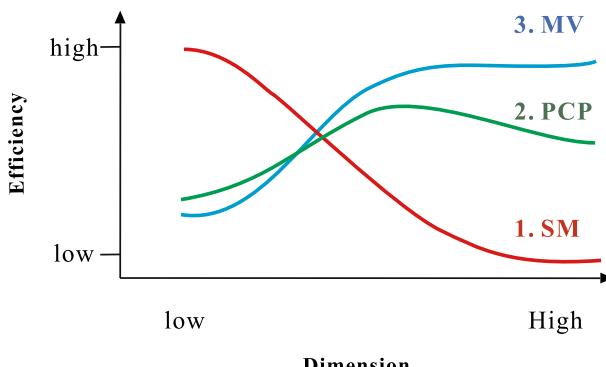


Figure 15.14. Schematic illustration of the relative efficiencies of the scatterplot matrix, the parallel coordinates plot, and matrix visualization, for varying numbers of dimensions

combinations for the missing structure can be accessed visually. MV users can benefit from a simple visual perception of the mechanism associated with the missing observations (random or not, ignorable or unignorable) before formal statistical modeling of the missing values is implemented.

15.7 Matrix Visualization of Binary Data

While scatterplots, PCP, and MV displays have their own advantages and disadvantages for continuous data structures of various dimensions, an MV display is the only statistical graph that can meaningfully display binary data sets over all dimensions. We use the KEGG (Kyoto Encyclopedia of Genes and Genomes) metabolism pathways (<http://www.genome.jp/kegg/pathway.html>) for *Saccharomyces cerevisiae* yeast to illustrate how an MV display can be generalized to visually extract all of the important information embedded in multivariate binary data. The KEGG website provides detailed information on the 1177 related genes involved in 100 metabolism pathways in *Saccharomyces cerevisiae* yeast. We simplified the complex information structure down to a two-way binary data matrix of 1177 genes by 100 pathways. This binary data matrix is called Dataset 3 in our study. A one (zero) encoded at the i th row and j th column of the matrix means that the i th gene is (not) involved in j th pathway activities.

15.7.1 Similarity Measure for Binary Data

The usual measures used to evaluate associations between samples and variables for continuous data – Euclidean distance and correlation coefficients – cannot be applied directly to binary data sets. Two issues are noted here in relation to the selection of similarity measures for binary data in an MV display.

Symmetric and Asymmetric Binary Variables

A binary variable is considered symmetric if both of its states are equally valuable; that is, there is no preference over which outcome should be coded as 0 or 1. A binary variable is asymmetric if the outcomes of the states are not equally important, such as the positive and negative outcomes of a disease diagnosis. Conventionally, the most important outcome (the rarer one) is coded 1, the other 0. Thus, asymmetric binary variables are often considered “monary” (as if there is only one state).

Sparseness and Dimensionality

Asymmetric binary variables are usually sparse in nature, and it is difficult to identify an appropriate association measure that could be used to assess the relationships among samples and between variables. Dimension reduction techniques also fail in attempts to summarize high-dimensional data structures in low-dimensional fashions. Listed in Fig. 15.15 are some similarity measures commonly applied to binary data. For sparse data, it is common practice to use the Jaccard coefficient instead of the simple matching coefficient.

Binary Data		Object B	
		1	0
Object A	1	a	b
	0	c	d
		(a+c)	(b+d)
			(a+b+c+d)

Similarity for Binary Data	Formula
Kulczynski	$\frac{a}{b+c}$
Rao	$\frac{a}{a+b+c+d}$
Jaccard	$\frac{a}{a+b+c}$
simple match	$\frac{a+d}{a+b+c+d}$
Sneath	$\frac{a}{a+2b+2c}$
Rogers	$\frac{a+d}{a+2b+2c+d}$
Hamman	$\frac{a+d-(b+c)}{a+b+c+d}$
Phi	$\frac{ad-bc}{\sqrt{(a+b)(a+c)(d+b)(d+c)}}$
Yule	$\frac{ad-bc}{ad+bc}$

Figure 15.15. Some similarity measures for binary data

15.7.2 Matrix Visualization of the KEGG Metabolism Pathway Data

The 1-Jaccard distance coefficient is used to compute the proximity matrices for both genes and pathways in Fig. 15.16. Elliptical seriations (Chen, 2002) are employed to permute the two 1-Jaccard distance matrices and the binary pathway data matrix. One can easily see, from the binary data matrix map and the proximity matrix map for genes, that there are many genes that are only involved in the activities of a single pathway. We then exclude those genes from further analysis, since they provide no association information. This reduces the original 1177 genes by 100 pathways binary matrix to a 432 genes by 88 pathways matrix (some pathways are also excluded after excluding genes). When there are too few horizontal or vertical pixels for an MV

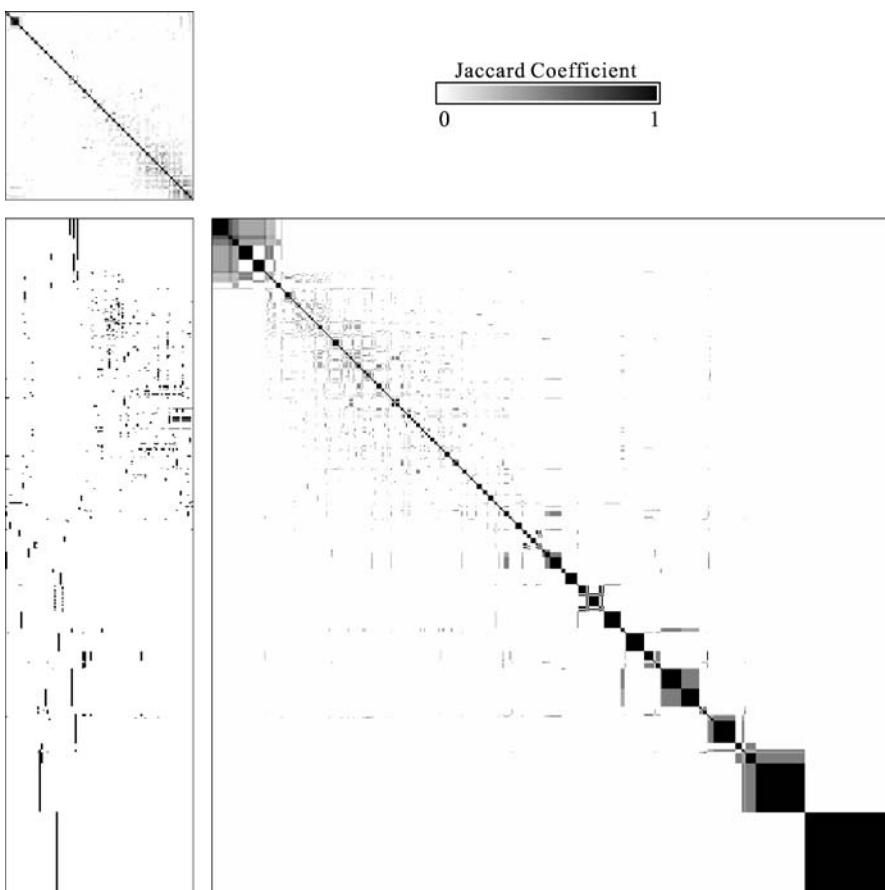


Figure 15.16. Binary data matrix map for the Dataset 3 (KEGG metabolite pathway database with 1177 genes (*rows*) for 100 pathways (*columns*)) with two Jaccard proximity matrices for genes and for pathways sorted by elliptical seriations in both directions

display, users can either use the scroll bars to visualize a certain region of the display or to zoom out in order to visualize the overall structure with an averaging effect, as used in a typical computer graph.

Average linkage clustering trees are then employed to sort the resulting 1–Jaccard distance matrices for genes and pathways and the corresponding binary data matrix, see Fig. 15.17. The association structure between genes and among pathways can now be comprehended using the three corresponding permuted matrix maps. In the upper left corner of the data matrix and the upper-left corners of the proximity maps for genes and pathways we can identify several groups of genes that are involved in the activities of only a few pathways, and several small groups of pathways that share

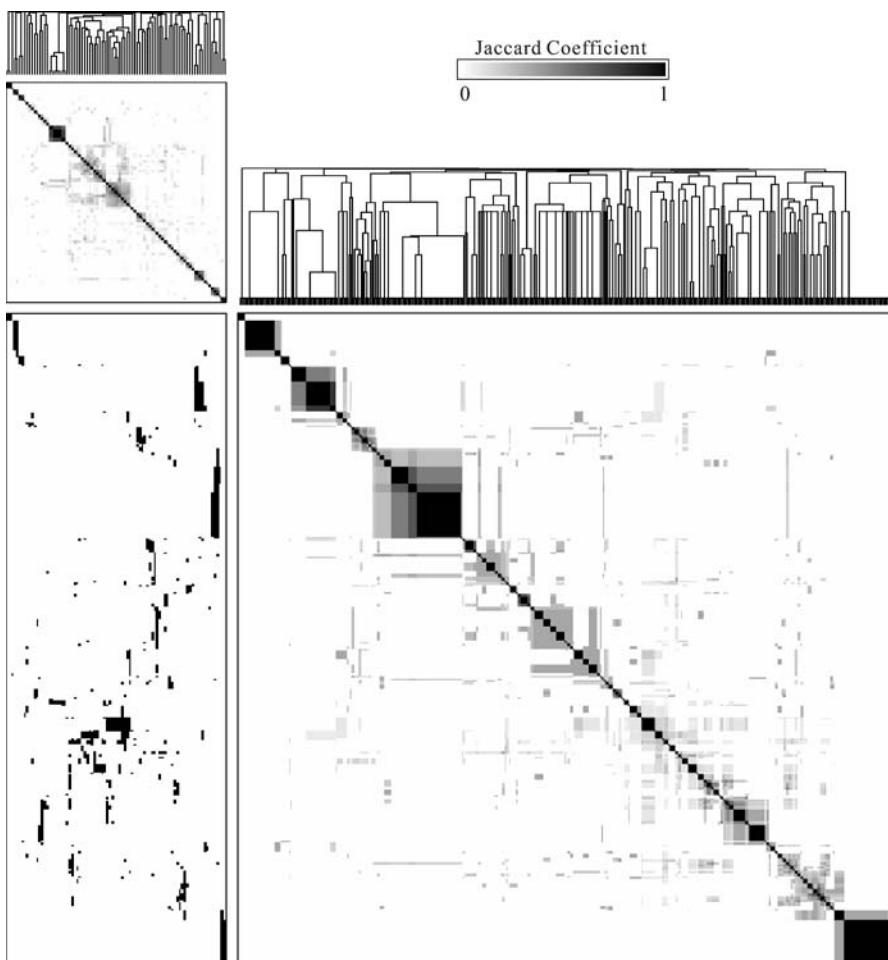


Figure 15.17. Binary data matrix for the reduced Dataset 4 (432 genes (*rows*) for 88 pathways (*columns*)) with two Jaccard proximity matrices for genes and for pathways sorted by average linkage trees in both directions

very similar groups of genes. The other genes and pathways have more complicated interactions between activities. It is of course possible to further exclude pathways and genes with simpler behavior, and to focus on the details of interactions of the more active genes and pathways.

15.8

Other Modules and Extensions of MV

So far we have introduced the fundamental framework for matrix visualization in the GAP (Chen, 2002) approach to the visualization of continuous and binary data matrices, with corresponding derived proximity matrices. We have also presented some generalizations, such as the sufficient MV and the sediment, sectional, and restricted displays. In practice, observed data can be highly complex, to the degree that basic matrix visualization procedures are not rich enough to comprehend the data structure. In some situations, one may not be able to apply MV directly to the given data or proximity matrices. This section discusses ongoing projects and future directions that will make matrix visualization a more promising statistical graphical environment. One important feature of the GAP (Chen, 2002) approach to matrix visualization is that it usually contains four basic procedures: (1) color projection of the raw data matrix; (2) computation of two proximity matrices for variables and sample; (3) color projection of the two proximity matrices, and; (4) permutations of variables and sample. Most extensions of MV are related to the first two procedures. It is simple to adapt the aforementioned algorithms for the other two steps once the first two procedures are fixed.

15.8.1

MV for Nominal Data

It is much more difficult to perform MV for nominal data than it is for binary data, since one can use black/white to code 1/0 if the binary data is asymmetric, and the Jaccard and other coefficients for binary data in order to derive the relationships between variables and between samples. There is no natural way to guide the color-coding for multivariate nominal data in such a way that the color version of the relativity of a statistical graph still holds (Chang et al., 2002). The derivation of meaningful between-variable and between-sample proximity measures for nominal data is another challenging issue. Chen (1999) and Chang et al. (2002) utilized the Homals (de Leeuw, 1998) algorithm and developed a categorical version of matrix visualization that naturally resolved the two critical problems.

15.8.2

MV for Covariate Adjustment

Quite often, covariate data (such as gender and age) are collected in a study in addition to the variables of primary interest. When the effects of covariates are an issue, covariate adjustment must be taken into consideration, much as in a formal statistical modeling process. Wu and Chen (2006) introduced a unified regression model

approach which partitions the raw data matrix into model and residual matrices, and ordinary MV can be applied to these two derived matrices. The covariate adjustment process is accomplished through by estimating conditional correlations. For a discrete covariate, a correlation matrix for variables is decomposed into within- and between-component matrices. When the covariate is continuous, the conditional correlation is equivalent to the partial correlation under the assumption of joint normality.

Data with Missing Values

15.8.3

The relativity of a statistical graph (Chen, 2002) is the main concept used in seriation algorithms to construct meaningful clustered matrices. This property can be used to develop a weighted pattern method to impute the missing values. The initial proximity measurements for rows and columns with missing values can be computed with pair-wise complete observations, and then imputed values are estimated and updated iteratively for the subsequent proximity calculations and imputation.

Modeling Proximity Matrices

15.8.4

Many statistical modeling procedures try to visually explore the high-dimensional pattern embedded in a proximity matrix that records pair-wise similarity or dissimilarity for a set of objects through a low-dimensional projection. Multidimensional scaling, hierarchical clustering analysis, and factor analysis are three such statistical techniques. Four types of matrices are usually involved in the modeling processes of these statistical procedures. The input proximity matrix is transformed into a disparity matrix prior to fixing the statistical model that summarizes the information in the output distance matrix. A stress (residual) matrix is calculated to assess the goodness of fit for the modeling. Such a study aims to create a comprehensive diagnosis environment for statistical methods through various types of matrix visualization for the numerical matrices involved in the modeling process.

Conclusion

15.9

Matrix visualization is the color order-based representation of data matrices. It is beneficial to employ human vision to explore the structure in a matrix in the pursuit of further appropriate mathematical operations and statistical modeling. A good matrix visualization environment helps us to gain comprehensive insights into the underlying process. Rather than rely solely on numerical characteristics, it is suggested that matrix visualization should be performed as a preliminary step in modern exploratory data analysis, and research into and applications of matrix visualization continue to be of much interest.

Matrix visualization displays provide five levels of information: (1) raw scores for every sample/variable combination; (2) an individual sample score vector across all

variables and an individual variable vector across all samples; (3) an association score for every sample–sample and variable–variable relationship; (4) a grouping structure for variables and a clustering effect for samples, and; (5) an interaction pattern for sample clusters on variable groups.

With the capacity to display thousands of variables in a single picture, the flexibility to work with all types of data, and the ability to handle various manifestations of extraordinary data patterns (missing value, covariate adjustment), we believe that matrix visualization has the potential to become one of the most important graphical tools applied to exploratory data analysis (EDA) in the future.

References

- Alon, U., Barkai, N., Notterman, D.A., Gish, K., Ybarra, S., Mack, D. and Levine, A.J. (1999). Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays, *Proc Natl Acad Sci USA*, 96(12):6745–6750.
- Asimov, D. (1985). The grand tour: a tool for viewing multidimensional data, *SIAM Journal of Scientific and Statistical Computing*, 6(1):128–143.
- Bar-Joseph, Z., Gifford, D.K. and Jaakkola, T.S. (2001). Fast optimal leaf ordering for hierarchical clustering, *Bioinformatics*, 17:S22–S29.
- Bertin, J. (1967). *Semiolegie Graphique*, Paris: Editions gauthier-Villars. English translation by William J. Berg. as Semiology of Graphics: Diagrams, Networks, Maps. University of Wisconsin Press, Madison, WI, 1983.
- Carmichael, J.W. and Sneath, P.H.A. (1969). Taxometric maps, *Systematic Zoology*, 18:402–415.
- Chang, S.C., Chen, C.H., Chi, Y.Y. and Ouyoung, C.W. (2002). Relativity and resolution for high dimensional information visualization with generalized association plots (GAP), *Section for Invited Papers, Proceedings in Computational Statistics 2002 (Compstat 2002)*, Berlin, Germany, 55–66.
- Chen, C.H. (1996). The properties and applications of the convergence of correlation matrices. In *1996 Proceedings of the Section on Statistical Graphics of the American Statistical Association*, 49–54.
- Chen, C.H. (1999). Extensions of generalized association plots, *1999 Proceedings of the Section on Statistical Graphics of the American Statistical Association*, 111–116.
- Chen, C.H. (2002). Generalized association plots: information visualization via iteratively generated correlation matrices, *Statistica Sinica*, 12:7–29.
- Chen, C.H., Hwu, H.G., Jang, W.J., Kao, C.H., Tien, Y.J., Tzeng, S. and Wu, H.M. (2004). Matrix visualization and information mining, *Proceedings in Computational Statistics 2004 (Compstat 2004)*, pp. 85–100, Physika Verlag, Heidelberg.
- Eisen, M.B., Spellman, P.T., Brown, P.O. and Botstein, D. (1998). Cluster analysis and display of genome-wide expression patterns, *Proc Natl Acad Sci USA*, 95:14863–14868.
- Fisher, R.A. (1936). The use of multiple measurements in taxonomic problems, *Annals of Eugenics*, 7:179–188.

- Friendly, M. (2002). Corrgrams: exploratory displays for correlation matrices. *The American Statistician*, 56(4):316–324.
- Friendly, M. and Kwan, E. (2003). Effect ordering for data displays. *Computational Statistics and Data Analysis*, 43(4):509–539.
- Hartigan, J.A. (1972). Direct clustering of a data matrix. *Journal of the American Statistical Association*, 78:123–129.
- Hurley, C.B. (2004). Clustering visualization of multidimensional data, *Journal of Computational and Graphics Statistics*, 13:788–806.
- Inselberg, A. (1985). The plane with parallel coordinates, *The Visual Computer*, 1:69–91.
- Lenstra, J. (1974). Clustering a data array and the traveling salesman problem, *Operations Research*, 22:413–414.
- Ling, R.L. (1973). A computer generated aid for cluster analysis, *Communications of the ACM*, 16(6):355–361.
- Marc, P., Devaux, F. and Jacq, C. (2001). yMGV: a database for visualization and data mining of published genome-wide yeast expression data, *Nucleic Acids Research*, 29(13):e63.
- Marchette, D.J. and Solka, J.L. (2003). Using data images for outlier detection, *Computational Statistics and Data Analysis*, 43:541–552.
- Michailidis, G. and de Leeuw, J. (1998). The Gifi system for descriptive multivariate analysis, *Statistical Science*, 13:307–336.
- Minnotte, M. and West, W. (1998). The data image: a tool for exploring high dimensional data sets, in *Proceedings of the ASA Section on Statistical Graphics*, Dallas, TX, 25–33.
- Murdoch, D.J. and Chow, E.D. (1996). A graphical display of large correlation matrices, *The American Statistician*, 50:178–180.
- Robinson, W.S. (1951). A method for chronologically ordering archaeological deposits, *American Antiquity* 16:293–301.
- Slagle, J.R., Chang, C.L. and Heller, S.R. (1975). A clustering and data-reorganizing algorithm, *IEEE Trans. Syst. Man Cybern.*, 5:125–128.
- Streng, R. (1991). Classification and seriation by iterative reordering of a data matrix. In *Classification, Data Analysis, and Knowledge Organization: Models and Methods with Applications* (Edited by H.H. Bock and P. Ihm), 121–130. Springer, New York.
- Tenenbaum, J.B., de Silva, V. and Langford, J.C. (2000). A global geometric framework for nonlinear dimensionality reduction, *Science*, 290(5500):2319–2323.
- Tibshirani, R., Hastie, T., Eisen, M., Ross, D., Botstein, D. and Brown, P. (1999). Clustering methods for the analysis of DNA microarray data. Technical Report, Stanford University, Oct. 1999.
- Tien, Y.J., Lee, Y.S., Wu, H.M. and Chen, C.H. (2006). Integration of clustering and visualization methods for simultaneously identifying coherent local clusters with smooth global patterns in gene expression profiles. Technical Report 2006-11, Institute of Statistical Science, Academia, Taiwan.
- Tukey, J.W. (1977). *Exploratory Data Analysis*. Addison-Wesley, Reading, MA.

- Unwin, A.R., Hawkins, G., Hofmann, H. and Siegl, B. (1996). Interactive graphics for data sets with missing values – MANET, *Journal of Computational and Graphical Statistics* 5:113–122.
- Unwin, A.R and Hofmann, H. (1998). New interactive graphics tools for exploratory analysis of spatial data. In *Innovations in GIS 5*, ed. S Carver, pp. 46–55. Taylor & Francis, London.
- Wegman, E.J. (1990). Hyperdimensional data analysis using parallel coordinates. *Journal of the American Statistical Association*. 85:664–675.
- Wu, H.M. and Chen, C.H. (2005). *Covariate adjusted matrix visualization*. Technical Report. Institute of Statistical Science, Academia, Taiwan.

Visualization in Bayesian Data Analysis

Jouni Kerman, Andrew Gelman, Tian Zheng, Yuejing Ding

16.1	<i>Introduction</i>	710
	The Role of EDA in Model Comprehension and Model-Checking	710
	Comparable Non-Bayesian Approaches	711
16.2	<i>Using Visualization to Understand and Check Models</i>	712
	Using Statistical Graphics in Model-Based Data Analysis	712
	Bayesian Exploratory Data Analysis	712
	Hierarchical Models and Parameter Naming Conventions	715
	Model-Checking	715
16.3	<i>Example: A Hierarchical Model of Structure in Social Networks</i>	716
	Posterior Predictive Checks	720
16.4	<i>Challenges Associated with the Graphical Display of Bayesian Inferences</i>	721
	Integrating Graphics and Bayesian Modeling	722
16.5	<i>Summary</i>	722

Introduction

Modern Bayesian statistical science commonly proceeds without reference to statistical graphics; both involve computation, but they are rarely considered to be connected. Traditional views about the usage of Bayesian statistics and statistical graphics result in a certain clash of attitudes between the two. Bayesians might do some exploratory data analysis (EDA) to start with, but once the model or class of models is specified, the next analytical step is to fit the data; graphs are then typically used to check convergence of simulations, or they are used as teaching aids or as presentation tools – but not as part of the data analysis. Exploratory data analysis appears to have no formal place in Bayesian statistics once a model has actually been fitted. According to this extreme view, the only connection between Bayesian inference and graphics occurs through convergence plots of Markov chain simulations, and histograms and kernel density plots of the resulting estimates of scalar parameters.

On the other hand, the traditional attitude of statistical graphics users is that “all models are wrong;” we are supposed to keep as close to the data as possible without referencing a model, since models incorporate undesirable subjective components and parametric assumptions into preliminary analysis. In true Tukey tradition, even if a graphical method can be derived from a probability model (e.g., rootograms from the Poisson distribution), we still don’t mention the model, because the graph should stand or fall on its own.

Given these seemingly incompatible attitudes, how can we then integrate the inherently model-based Bayesian inference with the (apparently) inherently model-aversive nature of statistical graphics? Our attitude is a synthesis of ideas adopted from statistical graphics and Bayesian data analysis. The fundamental idea is that we consider all statistical graphs to be *implicit or explicit comparisons* to a reference distribution; that is, to a model. This idea is introduced in Gelman (2004); the article proposes an approach that can be used to unify EDA with formal Bayesian statistical methods. The connection between EDA and goodness-of-fit testing is discussed in Gelman (2003). These two articles formalize the graphical model-checking ideas presented in Buja et al. (1996); Buja and Cook (1999); Buja et al. (1988), which have been informally applied in various contexts for a long time (e.g., Bush and Mosteller, 1955; Ripley, 1988).

The Role of EDA in Model Comprehension and Model-Checking

Exploratory data analysis, which is aided by the use of graphs, is done to look for patterns in the data. While the reference distributions in such cases are typically implicit, they are always there in the mind of the modeler. In an early article on EDA by Tukey (1972), he focused on the idea that “graphs intended to let us see what may be happening over and above what we have already described,” which suggests that these graphs can be built upon existing models. After all, to look for the un-

expected is to look for something that differs from something that we were expecting – the reference model. For example, even simple time series plots are viewed implicitly as comparisons to zero, a horizontal line, linearity, monotonicity, and so forth. Viewing two-way scatterplots usually implies a reference to an assumed model of independence. Before looking at a histogram, we have certain baselines of comparison (symmetric distribution, bimodal, skewed) in our minds. In the Bayesian sense, looking at inferences and deciding whether they “make sense” can be interpreted as a comparison of the estimates with our prior knowledge; that is, to a prior model.

The ideas that EDA gives us can be made more powerful if used in conjunction with sophisticated models. Even if one believes that graphical methods should be model-free, it can still be useful to have provisional models that make EDA more effective. EDA can be thought of as an implicit comparison to a multilevel model; in addition, EDA can be applied to inferences as well as to raw data.

In Bayesian probability model comprehension and model-checking, the reference distribution can be formally obtained by computing the predictive distribution of the observables, which is also called the replication distribution. Draws from the posterior predictive distribution represent our previous knowledge of the (marginal) distribution of the observables. The model fit can be assessed by comparing the observed values with posterior predictive draws; discrepancies represent departures from the model. Comparisons are usually best made via graphs, since the models used for the observables are usually complex. However, depending on the complexity of the model, highly sophisticated graphical checks often need to be devised and tailored to the model. In this article, we review these principles, show examples of how to apply them to data analysis, and discuss potential extensions.

Comparable Non-Bayesian Approaches

16.1.2

Our Bayesian data visualization tools make use of posterior uncertainty, as summarized by simulated draws of parameters and replicated data. A similar non-Bayesian analysis might compute point estimates for parameters and then simulate data using a parametric bootstrap. This reduces to (Bayesian) posterior predictive checking if the parameter estimates are estimated precisely (if the point estimate has no posterior variance).

A confidence interval (the point estimate plus or minus the standard error) approximately summarizes the posterior uncertainty about a parameter. In multilevel models, a common non-Bayesian approach is to compute point estimates for the hyperparameters and then simulate the modeled parameters.

The visualization tools described in this article should also work in these non-Bayesian settings.

16.2 Using Visualization to Understand and Check Models

The key to Bayesian inference is its unified treatment of uncertainty and variability; we would like to use this in data visualization (e.g., Wilkinson, 2005, Chap. 15) as well as in data analysis in general (Kerman and Gelman, 2007).

16.2.1 Using Statistical Graphics in Model-Based Data Analysis

EDA is the search for unanticipated areas of model misfit. Confirmatory data analysis (CDA), on the other hand, quantifies the extent to which these discrepancies could be expected to occur by chance. We would like to apply the same principles to the more complex models that can now be fitted, using methods such as Bayesian inference and nonparametric statistics. Complex modeling makes EDA more effective in the sense of being able to capture more subtle patterns in data. Conversely, when complex models are used, graphical checks are even more desirable in order to detect areas of model misfit.

We, like other statisticians, do statistical modeling in an iterative fashion, exploring our modeling options, starting with simple models, and expanding the models into more complex and realistic models, putting in as much structure as possible, trying to find deficiencies in our model at each stage, building new models, and iterating this process until we are satisfied. We then use simulation-based model checks (comparisons of observed data to replications under the model); to find patterns that represent deviations from the current model. Moreover, we apply the methods and ideas of EDA to structures other than raw data, such as plots of parameter inferences, latent data, and completed data (Gelman et al., 2005); Fig. 16.1 illustrates this.

At a theoretical level, we look at the model, identifying different sorts of graphical displays with different symmetries or invariances in an explicit or implicit reference distribution of the test variables. This serves two purposes: to add some theoretical structure to the graphics and EDA, so that graphical methods can be interpreted in terms of implicit models, and to give guidelines on how to express model checking as a graphical procedure most effectively.

16.2.2 Bayesian Exploratory Data Analysis

Bayesian inference has the advantage that the reference distribution – the predictive distribution for the data that could have been observed – arises directly from the posterior simulations (which are typically obtained using iterative simulation). Consequently, we can draw from this distribution and use these simulations to produce a graph comparing the predictions to the observed data. Such graphs can be customized to exploit symmetries in the underlying model to aid interpretation. The inclusion of imputed missing and latent data can result in more understandable completed-data exploratory plots.

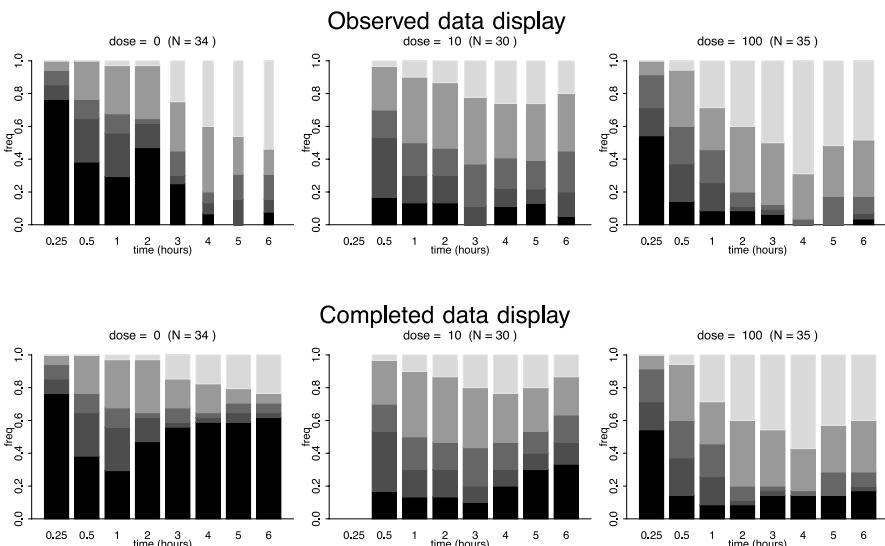


Figure 16.1. Summary of pain relief responses over time for different doses from a clinical trial with unignorable dropout. In each summary bar, the shadings from bottom to top indicate “no pain relief” through intermediate levels up to “complete pain relief.” The graphs in the *top row* only include the people that have not dropped out (the widths of the histogram bars are proportional to the number of subjects remaining at each time point). The graphs in the *bottom row* include everybody, with imputed responses for the dropouts. The bottom row of plots – which are based on complete data sets – are much more directly interpretable than the observed-data plots on the *top row*

The existence of an explicit underlying model is beneficial in that it suggests exploratory plots that address the fit of the data to whichever model is being fitted to them. Placing EDA within the general theory of model-checking potentially enables graphical methods to become a more acceptable presence in statistical modeling.

Our approach is to use statistical graphics in all stages of data analysis: model-building, model-fitting, model-checking and model-understanding. In all stages of data analysis, we need model-checking tools so that we can see the faults and shortcomings in our model. Understanding complex probability models often requires complex tools. Simple test statistics and p -values just do not suffice, so we need graphs to aid us. More often than not we need to customize the graphs to the problem; we are not usually satisfied with a standard graph. However, new and better standard methods of graphical display need to be developed.

Considering that graphs are equivalent to model checks, “exploratory” data analysis is not done only at the beginning of the modeling process; the model is re-explored at each stage after model fitting. The construction of a new, complex model may result in unforeseen problems during the fitting; in anticipation of these, we explore the fitted model with standard and customized statistical graphs that are then used to attempt to falsify the model and to find ways to improve it, or to discard it completely and start all over again. It is also standard for our practice not to perform model-

averaging or to concentrate on the selection of the predictor in regression problems; our models usually evolve to more and more complex ones.

The key idea in Bayesian statistics – as opposed to simply “statistical modeling” or “estimation” – is to work with posterior uncertainty in inferences. At the theoretical level, this means using random variables; at a more practical level, this implies the use of simulations that represent draws from the joint posterior distribution. This is seen most clearly in hierarchical modeling. Figure 16.2 shows an example of the visualization of posterior uncertainty in a hierarchical logistic regression model (Gelman et al., 2005).

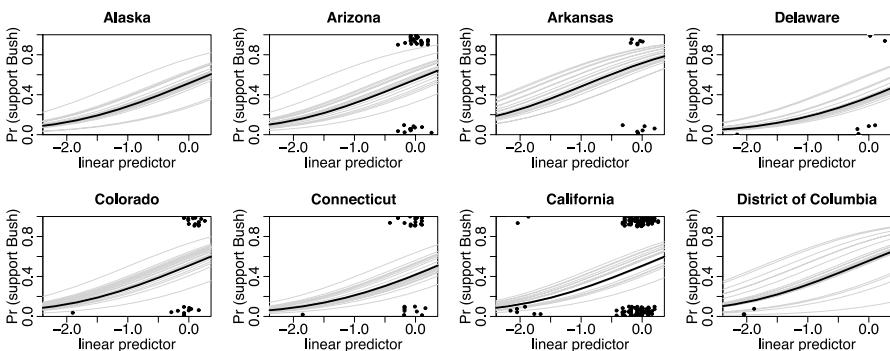


Figure 16.2. Displaying a fitted hierarchical logistic regression model, along with inferential uncertainty. Eight states are shown from a voting preference model $\text{Prob}(y_i = 1) = \text{logit}^{-1}(\alpha_{j[i]} + X_i\beta)$ that includes all 50 US states. The *solid black line* is the median estimate for the probability that a survey respondent in that state supported George Bush in his presidential campaign in 1988. The *gray lines* represent random draws from the posterior distribution of the logistic regression coefficients. The *dots* shown are the observed data (zeros and ones), vertically jittered to make them more distinguishable. This figure demonstrates several principles of Bayesian visualization: (1) small multiples: parallel plots display the hierarchical structure of the data and model; (2) graphs are used to understand the fitted model; (3) the fitted model and the data are shown on the same graph; (4) posterior uncertainty is displayed graphically. A principle of Bayesian inference is also illustrated: there were no data for Alaska but we were still able to draw predictions about the voting preference in this state from the model. General principles of “good graphics” are used: a common scale is used for all graphs, with bounds at 0 and 1; axes are clearly labeled; jittering is employed (which works for moderate sample sizes like that used in this example); thin lines and small dots are used. However here, the small plots should be ordered by some meaningful criterion, such as by decreasing support for Republicans, rather than alphabetically. The distribution of the linear predictor is skewed because the most important single predictor by far was the indicator for African-Americans, which has an expected value of 0.12. The common scaling of the axes means that we do not actually need to label the axes on each graph; however, we find that repeating the labels is convenient in this case. Labeling only some graphs (as done for trellis plots) saves space but makes the graph more of a challenge to read, especially when used as presentation graphics

Hierarchical Models and Parameter Naming Conventions

16.2.3

In hierarchical and other structured models, rather than display individual coefficients, we wish to compare the values within batches of parameters. For example, we might want to compare group-level means together with their uncertainty intervals. Posterior intervals are easily derived from the matrix of posterior simulations. Traditional tools for summarizing models (such as looking at coefficients and analytical relationships) are too crude to usefully summarize the multiple levels of variation and uncertainty that arise in such models. These can be thought of as corresponding to the “sources of variation” in an ANOVA table.

Hierarchical models feature multiple levels of variation, and hence feature multiple levels of batches of parameters. Hence, the choice of label for the batches is also important: parameters with similar names can be compared to each other. In this way, naming can be thought of as a structure analogous to hierarchical modeling. Instead of using generic $\theta_1, \dots, \theta_k$ for all scalar parameters, we would, for example, name the individual-level regression coefficients $\beta = (\beta_1, \dots, \beta_n)$, and the group-level coefficients $\alpha = (\alpha_1, \dots, \alpha_J)$, and the intercept μ . Figure 16.4 shows an example of why this works: parameters with similar names can be compared to each other. Rather than plotting posterior histograms or kernel density estimates of the parameters, we usually summarize the parameters (at least in a first look for the inferences) by plotting their posterior intervals.

Model-Checking

16.2.4

As stated earlier, we view statistical graphics as implicit or explicit model checks. Conversely, we view model-checking as a comparison of the data with the replicated data given by the model, which includes both exploratory graphics and confirmatory calculations such as p -values. Our goal is *not* the classical one of identifying whether the model fits or not – and it is certainly *not* the aim to classify models into correct and incorrect, which is the focus of the Neyman–Pearson theory for Type 1 and Type 2 errors. Instead, we seek to understand the ways in which the data depart from the fitted model. From this perspective, the two key components of exploratory model-checking are (1) the graphical display and (2) the reference distribution to which the data are compared.

The best display to use depends on the aspects of the model being checked, but in general, graphs of data or functions of data and estimated models (for example, residual plots) can be visually compared to corresponding graphs of replicated data sets. This is the fully model-based version of exploratory data analysis. The goal is to use graphical tools to explore aspects of the data that are not captured by the fitted model.

Example: A Hierarchical Model of Structure in Social Networks

16.3 As an example, we consider the problem of estimating the sizes of social networks Zheng et al. (2006). The model uses a negative-binomial model with an overdispersion parameter:

$$y_{ik} \sim \text{Negative-binomial}(\text{mean} = a_i b_k, \text{overdispersion} = \omega_k),$$

where the groups (subpopulations) are indexed by k ($k = 1, \dots, K$) and the respondents are indexed by i ($i = 1, \dots, n$). In this study, $n = 1370$ and $K = 32$. Each respondent is asked how many people he or she knows in each of the K subpopulations. The subpopulations are identified by name (people called Nicole, Michael, Stephanie, etc.), and by certain characteristics (airline pilots, people with HIV, those in prison, etc.).

Without going into the details, we remark that a_i is an individual-level parameter that indicates the propensity of the person i to know people in other groups (we call this a “gregariousness” parameter); it is modeled to be $a_i = e^{\alpha_i}$ where $\alpha_i \sim N(\mu_\alpha, \sigma_\alpha^2)$; similarly, b_k is a group-level prevalence (or group size) parameter modeled as $b_k = e^{\beta_k}$ where $\beta_k \sim N(\mu_\beta, \sigma_\beta^2)$. The overdispersion parameter vector $\omega = (\omega_1, \dots, \omega_K)$ and the hyperparameters are assigned uninformative (or weakly informative) prior distributions.

The model is fitted using a combination of Gibbs and Metropolis algorithms, so our inferences for the modeled parameters (a, b, ω), and the hyperparameters, $(\mu_\alpha, \sigma_\alpha, \mu_\beta, \sigma_\beta)$, are obtained as simulated draws from their joint posterior distribution.

Model-Informed Exploratory Data Analysis

Figure 16.3 displays a small portion of an exploratory data analysis, with histograms of responses for two of the survey questions, along with simulations of what could appear under three fitted models. The last of the models is the one that we used; as the EDA shows, the fit is still not perfect.

A First Look at the Estimates

We could summarize the estimated posterior distributions of the scalar parameters as histograms, but in most cases we find that intervals are a more concise way to display the inferences; our goal here is not just to view estimates, but also to compare the parameters within each batch.

We display the parameter estimates with their 50 % and 95 % posterior intervals as shown in Fig. 16.4. Along with the estimates, the graph summarizes the convergence statistic graphically. Since there are over 1300 quantities in the model, not all of them can be displayed on one sheet. For smaller models, this graph provides a quick summary of the results – but of course this is just a starting point.

We are usually satisfied with the convergence of the algorithm if the values of the \hat{R} convergence statistic (Gelman et al., 2003) are below 1.1 for all scalar parameters.

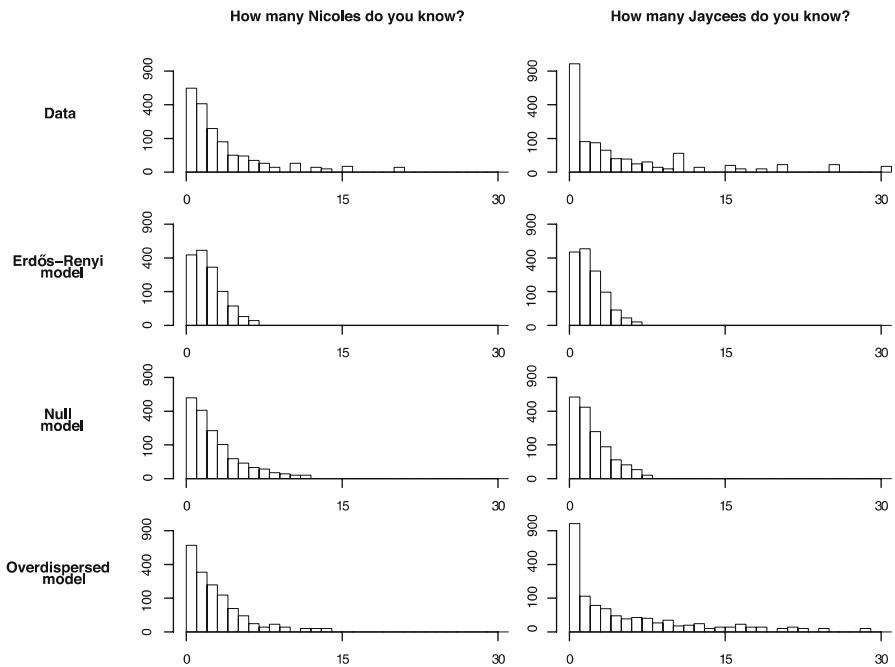


Figure 16.3. Histograms (with a square-root scale) of responses to “How many persons do you know named Nicole?” and “How many Jaycees do you know?” constructed from the data and from random simulations obtained for three fitted models: the Erdős–Renyi model (completely random links), our null model (some people are more gregarious than others, but the propensities of people to form ties to all groups are the same), and our overdispersed model (variation in gregariousness and variation in propensities to form ties to different groups). Each model shows more dispersion than the one above, with the overdispersed model fitting the data reasonably well. The propensities to form ties to Jaycees show much more variation than the propensities to form ties to Nicoles, and hence the Jaycees counts are much more overdispersed. (The data also show minor idiosyncrasies, such as small peaks at the responses 10, 15, 20, and 25. All values that are greater than 30 have been truncated at 30.) We use square-root scales to make tail patterns clearer

A value of \hat{R} that is close to 1.0 implies good chain mixing; if however $\hat{R} > 1.1$ for some of the parameters, we allow the sampler to iterate some more. By using a scalar summary rather than looking at trace plots, we are able to quickly assess the mixing of all of the parameters in the model.

Distribution of Social Network Sizes a_i

We now proceed to summarize the estimates of the 1370 parameters a_i . A table of numbers would be useless unless we want to find the numerical posterior estimates for a certain person in the study; our goal is rather to visualize the posterior distribution of the a_i , so a histogram is a much more appropriate summary. It is interesting to see how men and women differ in their perceived “gregariousness;” we therefore

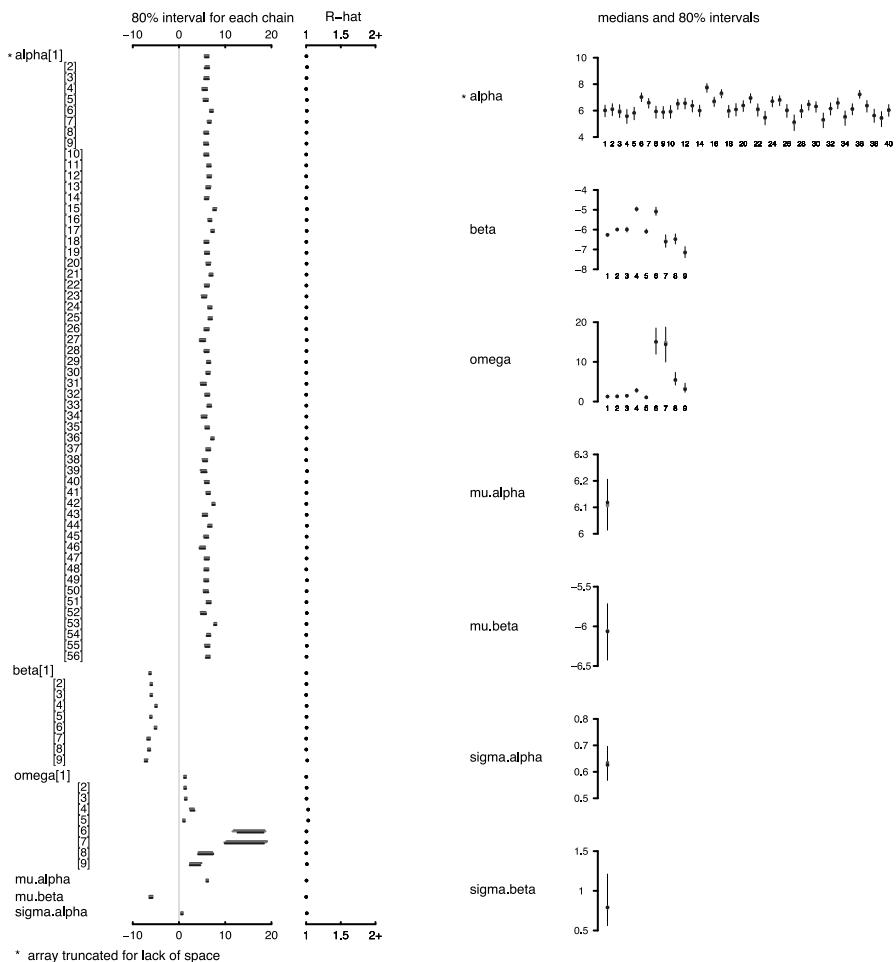


Figure 16.4. A graphical summary (produced automatically by the R package R2WinBUGS Sturtz et al. (2005)) of the estimated scalar parameters in the social networks model. Another alternative is to plot density estimates or histograms of individual scalar parameters; these can be useful, but are difficult to apply to vector parameters. In contrast, the right side of our display here permits an immediate understanding of the inferences for the vectors α, β, ω (another option would be to use parallel coordinate plots for each vector parameter). This graph is an inefficient way of displaying inferences; given that convergence has approximately been reached, only the right side of the display is necessary. However, we include it as an example of a quick inferential summary which, for all its simplicity, is still far more informative than a table of parameter estimates and standard errors

display the posterior mean estimates of a_i as two separate histograms by dividing α_i into men's and women's estimates. See Fig. 16.5.

However, a histogram derived from the posterior means of hundreds of parameters is still only a point estimate; we also want to visualize the posterior *uncertainty*

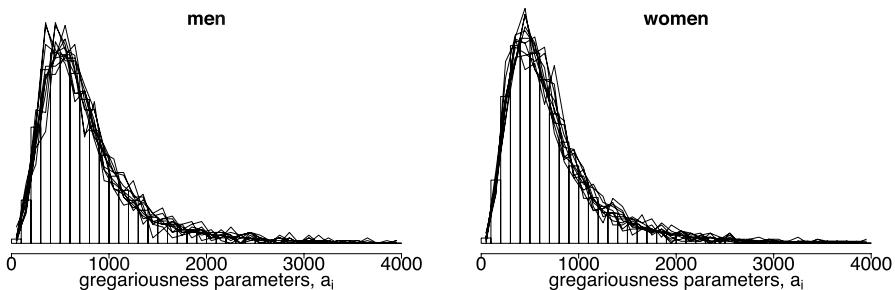


Figure 16.5. Estimated distributions of the “gregariousness parameters” α_i for women and men. The lines drawn over the bars are posterior simulation draws that indicate the inferential uncertainty in the histograms. Our primary goal here is to display inferences for the distribution of gregariousness within each sex, not to compare averages between sexes (which could be done, for example, using parallel boxplots). We compare groups using a regression model as in Fig. 16.6

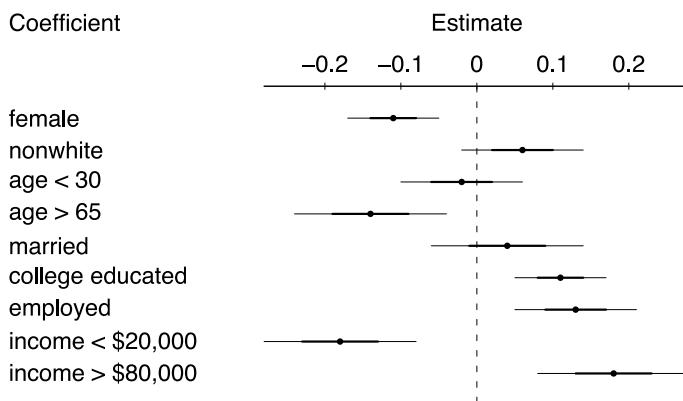


Figure 16.6. Coefficients (and ± 1 and ± 2 standard error intervals) for the regression of estimated log gregariousness parameters α_i on personal characteristics. Because the regression is done on a logarithmic scale, the coefficients (with the exception of the constant term) can be interpreted as proportional differences; thus, with all else held constant, women have social network sizes that are 11 % smaller than men, people that are over 65 years of age have social network sizes that are 14 % lower than others, and so forth

in the histogram. In the case of one scalar, we always draw the posterior intervals that account for the uncertainty in estimation; in the case of a vector shown as a histogram, we similarly want to display the uncertainty in the histogram estimate. To do this, we sample (say, twenty) vectors from the matrix of simulations and overlay the twenty histogram estimates as lines on the average histograms. This gives us a rough idea of how good an estimate the histogram is. As a rule of thumb, we don’t plot the “theta-hats” (point estimates), we plot the “thetas” (posterior draws representing the random variables) themselves.

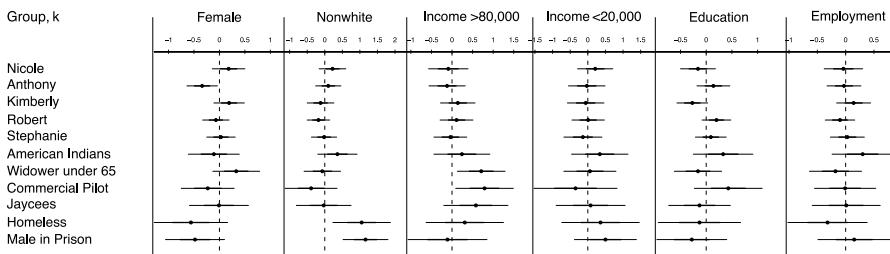
Characteristics of survey respondent, i 

Figure 16.7. A graph comparing the estimates from a more complex version of the social networks model, using individual-level regression predictors: $\log(a_i) \sim N(X_i\psi, \sigma_\alpha^2)$. The rows (Nicole, Anthony, etc.) refer to the groups and the columns refer to the various predictors. Comparisons are efficient when coefficients are rearranged into a “table of graphs” like this

Extending the Model by Imposing a Regression Structure

We also fit an extended model with an individual-level regression model, $\log(a_i) \sim N(X_i\psi, \sigma_\alpha^2)$. The predictors include the indicators of female, nonwhite, income $> \$ 80\,000$, income $< \$ 20\,000$, employment, education (high-school or higher).

Figure 16.4 shows an example of how to visually compare regression coefficients ψ_{ik} on explanatory variables (a characteristic of the survey respondent, i) for different groups (k). Whenever our goal is to compare estimates, we initially think of a graph. Figure 16.7 could have been equivalently summarized by the uncertainty interval endpoints and the posterior median estimate, but it would not have been an efficient tool to use to visualize the coefficient estimates. When a comparison is required, draw a graph; for looking up specific numbers, construct a table.

16.3.1 Posterior Predictive Checks

A natural way to assess the fit of a Bayesian model is to look at how well the predictions from the model agree with the observed data (Gelman et al., 2003). We do this by comparing the posterior predictive simulations with the data. In our social networks example, we create a set of predictive simulations by sampling new data independently from the negative binomial distributions given the parameter vectors a, b, ω drawn from the posterior simulations already calculated. We draw, say, L simulations, $y_{ik}^{(1)}, \dots, y_{ik}^{(L)}$ for each i, k . Each set of simulations $\{y_{ik}^{(\ell)}\}_{\ell=1}^L$ is an approximation of the marginal posterior distribution of y_{ik} , denoted by y_{ik}^{rep} , where “rep” stands for the “replication distribution;” y^{rep} stands for the $n \times K$ replicated observation matrix.

It is possible to find a numerical summary (that is, a test statistic) for some feature of the data (such as standard deviation, mean, etc.) and then compare it to the corresponding summary of y^{rep} , but in addition we prefer to draw *graphical test statistics*, since a few individual numbers rarely illustrate the complexity of the full dataset. In

general notation, for some suitable test function T , which may thus be a graph, we compare $T(y)$ with $T(y^{\text{rep}})$.

Plots of Data Compared with Replicated Data

For the social networks model, we choose to compare the data and the predictions by plotting the observed versus expected proportions of responses y_{ik} . For each subpopulation k , we compute the proportion of the 1370 respondents for which y_{ik} equals 0, 1, 3, 5, 9, 10, and finally those with $y_{ik} \geq 13$. These values are then compared to posterior predictive simulations under the model. Naturally, we plot the uncertainty intervals of $\text{Prob}_k(y_{ik} = m)$ instead of their point estimates.

The bottom row of Fig. 16.8 shows the plots. On the whole, the model fits the aggregate counts fairly well, but tends to underpredict the proportion of respondents who know exactly one person in a category. In addition, the data and predicted values for $y = 9$ and $y = 10$ illustrate that people are more likely to answer with round numbers.

The three first rows of Fig. 16.8 shows the plots for three alternative models (Zheng et al., 2006). This plot illustrates one of our main principles: whenever we need to compare a series of graphs, we plot them side by side on the same page so that visual comparison is efficient (Bertin, 1967/1983; Tufte, 1990). There is no advantage to scattering closely related graphs over several pages.

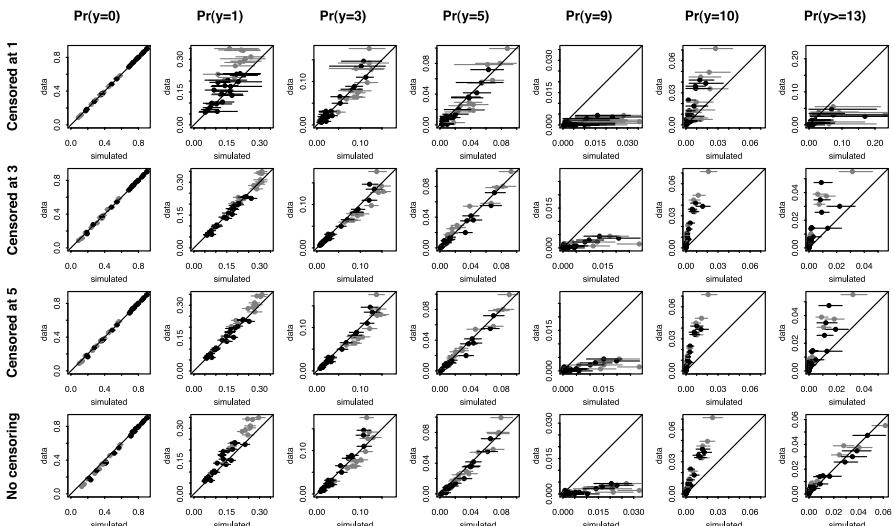


Figure 16.8. Model-checking graphs: observed vs. expected proportions of responses y_{ik} of 0, 1, 3, 5, 9, 10, and ≥ 13 . Each row of plots compares actual data to the estimate from one of four fitted models. The bottom row shows our main model, and the top three rows show the models fitted while censoring the data at 1, 3, and 5, respectively. In each plot, each dot represents a subpopulation, with proper name categories in gray, other categories in black, and 95 % posterior intervals are indicated by horizontal lines

Challenges Associated with the Graphical Display of Bayesian Inferences

We expect that the quality of statistical analyses would benefit greatly if graphs were more routinely used as part of the data analysis. Exploratory data analysis would be more effective if it could be implemented as a part of the software for complex modeling. To some extent this is done with residual plots in regression models, but for complex models there is the potential for much more progress.

As discussed in detail in Gelman (2004), we anticipate four challenges: (1) integrating the automatic generation of replication distributions into the computing environment; (2) choosing the replication distribution – this is not an automatic task, since the task involves selecting which parameters to resample and which to keep constant; (3) choosing the test variables; (4) displaying test variables as graphs. In the near future, automatic features for simulating replication distributions and performing standard model checks should become available.

16.4.1 Integrating Graphics and Bayesian Modeling

We fit Bayesian models routinely with software such as BUGS (BUGS Project, 2004), and move the simulations over to R (R Development Core Team, 2004) using R2WinBUGS (Sturtz et al., 2005). Simulations can also be summarized in R in a more natural way by converting the simulation matrices into vectors of random variable objects (Kerman and Gelman, 2007). BUGS has also its limitations; we also fit complex models in R using Umacs, the “Universal Markov chain sampler,” (Kerman, 2006).

We are currently investigating how to define an integrated Bayesian computing environment where modeling, fitting, and automatic generation of replications for model-checking is possible. This requires further effort to develop standardized graphical displays for Bayesian model-checking and understanding. An integrated computing environment is nevertheless the necessary starting point, since functions that generate such complex graphical displays should have full access to the models, the data, and predictions.

The environment must also take into account the fact that we may fit multiple models with different sets of observations; without a system that can distinguish multiple models (and the inferences associated with them) it is not possible to perform comparisons of them.

16.5 Summary

Exploratory data analysis and modeling can work well together: in our applied research, graphs are used to comprehend and check models. In the initial phase, we create plots that show us how the models work, and then plot data and compare it to the model to see where more work is needed.

We gain insight into the shortcomings of the model by performing graphical model checks. Graphs are most often drawn in order to compare data with an implicit reference distribution (e.g., Poisson model for rootograms, independence-with-mean-zero for residual plots, or normality for quantile–quantile plots), but we would also include more general comparisons; for example, a time series plot is implicitly compared to a constant line. In Bayesian data analysis, the reference distribution can be formally obtained by computing the replication distribution of the observables; the observed quantities can be plotted against draws from the replication distribution to compare the fit of the model.

We aim to make graphical displays an integrated and automatic part of data analysis. Standardized graphical tests must be developed, and these should be routinely generated by the modeling and model-fitting environment.

Acknowledgement. We thank Matt Salganik for collaboration with the social networks project, and the National Science Foundation for financial support.

References

- Bertin, J. (1967/1983). *Semiology of Graphics*. University of Wisconsin Press, Madison, WI. Translation by W.J. Berg.
- BUGS Project (2004) BUGS: Bayesian Inference Using Gibbs Sampling. <http://www.mrc-bsu.cam.ac.uk/bugs/>.
- Buja, A., Asimov, D., Hurley, C. and McDonald, J.A. (1988). Elements of a viewing pipeline for data analysis. In Cleveland, W.S. and McGill, M.E. (eds), *Dynamic Graphics for Statistics*, pp. 277–308, Wadsworth, Belmont, CA.
- Buja, A. and Cook, D. (1999). *Inference for data visualization*. Talk given at Joint Statistical Meetings 1999. Available at <http://www-stat.wharton.upenn.edu/~buja/PAPERS/jsm99.ps.gz>.
- Buja, A., Cook, D. and Swayne, D.F. (1996). Interactive high-dimensional data visualization. *Journal of Computational and Graphical Statistics*, 5:78–99.
- Bush, R.R. and Mosteller, F. (1955). *Stochastic Models for Learning*. Wiley, New York.
- Gelman, A. (2003). A Bayesian formulation of exploratory data analysis and goodness-of-fit testing. *International Statistical Review*, 71:369–382.
- Gelman, A. (2004). Exploratory data analysis for complex models (with discussion). *Journal of Computational and Graphical Statistics*, 13:755–779.
- Gelman, A., Carlin, J.B., Stern, H.S. and Rubin, D.B. (2003). *Bayesian Data Analysis*. Chapman & Hall/CRC, London, 2nd edn.
- Gelman, A., Mechelen, I.V., Verbeke, G., Heitjan, D.F. and Meulders, M. (2005) Multiple imputation for model checking: Completed-data plots with missing and latent data. *Biometrics*, 61:74–85.
- Gelman, A., Shor, B., Bafumi, J. and Park, D. (2005). *Rich state, poor state, red state, blue state: What's the matter with Connecticut?* Technical report, Department of Political Science, Columbia University, New York.
- Kerman, J. (2006). *Umacs: A Universal Markov Chain Sampler*. Technical report, Department of Statistics, Columbia University, New York.

-
- Kerman, J. and Gelman, A. (2007). Manipulating and summarizing posterior simulations using random variable objects. *Statistics and Computing*, 17:3.
- R Development Core Team (2004) *R: A language and environment for statistical computing*. R Foundation for Statistical Computing, Vienna, Austria.
- Ripley, B.D. (1988). *Statistical Inference for Spatial Processes*. Cambridge University Press, New York.
- Sturtz, S., Ligges, U. and Gelman, A. (2005). R2WinBUGS: A package for running WinBUGS from R. *Journal of Statistical Software*, 12(3):1–16.
- Tufte, E.R. (1990). *Envisioning Information*. Graphics, Cheshire, CT.
- Tukey, J.W. (1972). Some graphic and semigraphic displays. In Bancroft, T.A. (ed), *Statistical Papers in Honor of George W. Snedecor*. Iowa State University Press, Ames, IA.
- Wilkinson, L. (2005). *The Grammar of Graphics*. Springer, New York, 2nd edn.
- Zheng, T., Salganik, M.J. and Gelman, A. (2006). How many people do you know in prison?: Using overdispersion in count data to estimate social structure in networks. *Journal of the American Statistical Association*, 101(474):409–423.

Programming Statistical Data Visualization in the Java Language

III.17

Junji Nakano, Yoshikazu Yamamoto, Keisuke Honda

17.1	<i>Introduction</i>	726
17.2	<i>Basics of Statistical Graphics Libraries and Java Programming</i>	727
	Required Functions for Statistical Graphics Libraries	727
	Advantages of Java for Programming Statistical Graphics	729
	Basics of Java Graphics	730
	GoF Design Patterns	731
	MVC Design Pattern	733
	Class Diagrams in UML	733
17.3	<i>Design and Implementation of a Java Graphics Library</i>	735
	Overview of Jasplot	735
	Summary of Basic Interfaces and Classes with an Example	737
	Classes for Original Data.....	740
	Classes for Data about Basic Graphics	742
	Classes for Drawing Basic Graphics	743
	Classes of Panels for Drawing	744
	Classes for Interactive Operations	745
	Classes for Tables of Data	751
	A Class for Building Complicated Graphics	751
17.4	<i>Concluding Remarks</i>	753

When we have to write programs for statistical data visualization using a general-purpose programming language, for example, to achieve new functions or extensibility, the Java language is an appropriate choice. The object-oriented characteristics of Java are suitable for building graphical and interactive programs. Java has well-prepared standard graphical libraries that reduce our programming tasks and increase the portability of software to different platforms. Furthermore, recently developed solutions for object-oriented programming – so called “design patterns” – are useful for building statistical data visualization programs.

This chapter illustrates how to build general-purpose extensible statistical data visualization software in Java by following design patterns. As an example, we use a Java statistical graphics library named Jasplot (JAvA Statistical PLOT). The source code of Jasplot is available from our web site (<http://jasp.ism.ac.jp/Jasplot/>).

17.1

Introduction

Data visualization or statistical graphics is one of the important topics in statistics (Wilkinson, 2004). Statistical graphics are usually drawn by computers these days. Thanks to the development of computer technologies, beautiful and colorful graphics are easily drawn, and interactive and 3-D graphics are even available on personal computers (Symanzik, 2004). Almost all recently developed statistical software includes advanced statistical graphics functions, e.g., R (Murrell, 2005).

Although statistical software products have flexible and well-organized graphics functions, we sometimes need new functions that have not been implemented and are also difficult to realize in such software. For example, it is difficult to realize complicated interactive operations for graphics using only the traditional R language at this stage. We note that the iPlot project (Urbanek and Theus, 2003) expands the capabilities of R to provide interactive statistical graphics using comprehensive Java programs.

Before computers became available, it was difficult to draw statistical graphics for data that had many observations and variables. Now, such graphics are easily drawn by computers. However, we know that the numbers of observations and variables being recorded have also increased with the development of automatic data collection systems that use computers in various fields. Traditional statistical graphics sometimes fail to show the characteristics of such data clearly. For example, if we plot all data on a screen using simple graphics such as a scatterplot or a parallel coordinate plot (Inselberg, 1985), it is difficult to distinguish each observation from the others because of overlapping points. To solve problems like this, several interactive techniques such as focusing, brushing, zooming and linked views have been developed (Unwin et al., 2006). These techniques are useful for viewing particular observations clearly and for grasping characteristics of the data structure.

It is convenient to use general-purpose object-oriented programming (OOP) languages such as C++ or Java to program statistical data visualization. OOP is a paradigm where programs are composed by utilizing various “objects” in which related

data and procedures are encapsulated. This paradigm originally appeared in a programming language intended for simulation: Simula. The Smalltalk system is a pure OOP environment and shows that OOP is very useful for building window-type graphical user interfaces (GUI) by providing window frames, pull-down menus, sliders and buttons. In addition, GUI programming must be event-driven; that is, operations by users are not sequential, and the program needs to decide on its operations according to events caused by users; for example, the clicking of a mouse button or the selection of an item from a pull-down menu. Statistical data visualization is thought to be an advanced form of GUI. In addition to buttons and pull-down menus, it shows data points or lines on windows to express data or results from analysis.

Java is appropriate for modern data visualization for several reasons. First, it is an OOP language. Second, it has well-designed standard graphics libraries, such as Java 2D and Swing. These libraries can work as useful components of statistical graphics and are incorporated by applying so-called “design patterns.”

Design patterns are suggested solutions to common problems that often appear in OOP work (Gamma et al., 1995). They are derived from the experiences of professional programmers using OOP languages. As they aim for generality and reusability, they can appear cumbersome for simple programming. However, when designing reusable libraries that include statistical graphs, design patterns provide powerful guidelines.

In this chapter, we consider how to build a general-purpose statistical graphics library in Java. We explain the basic ideas associated with a statistical graphics library, the Java language, and design patterns in the next section. These ideas are actually used to build a Java statistical graphics library, Jasplot, in Sect. 3. Jasplot (JAvA Statistical PLOT) has been developed by adopting design patterns. Sect. 4 contains our concluding remarks. In the Appendix, we briefly explain how to install and use the Jasplot library.

Basics of Statistical Graphics Libraries and Java Programming

17.2

We recommend the use of Java to write statistical graphics programs. We briefly explain the reasons for this here. First, we consider the requirements of statistical graphics libraries. Then we state why Java is suited to achieving these requirements. Introductions to design patterns and UML class diagrams are also given, because they are used in the next section.

Required Functions for Statistical Graphics Libraries

17.2.1

Statistical graphics libraries must include at least the following functions. We note that basic statistical calculation functions are preferable for statistical graphics libraries. However, we can leave them to the software from which the library is used, and we do not consider them here.

Data Import and Output Export

As a statistical graphics library handles statistical data, it has to import data from files of other software products. There are many file formats for storing data and many software products with which the library is used. Thus, the data structure used in the library should be general and relatively simple to facilitate communication with many different formats, such as the Comma Separated Value (CSV) format and the Microsoft Excel format.

Graphics drawn by the library on screen should also be output on paper or web-pages. Several formats are used these days for this purpose, including the JPEG format and the Scalable Vector Graphics (SVG) format. The library is expected to support them.

Drawing Basic Statistical Graphics

One simple two-dimensional plot is a scatterplot. It expresses observations with two continuous variables as points on a two-dimensional Cartesian coordinate plane. We sometimes connect observation points using lines or fill a particular region with a special color. If we include such functions in a scatterplot, all graphics drawn on a plane are thought to be variations of them, because points, lines and regions are sufficient components to draw any two-dimensional graphics.

Histograms are traditionally used for single continuous variable data, and boxplots express different characteristics of them.

Scatterplot matrices and parallel coordinate plots are powerful and basic graphics used to represent continuous multivariate data. Mosaic plots are available for visualizing categorical data (Hofmann, 2000).

A statistical graphics library should include these basic graphics at the very least. They should be easy to use without complicated programming work.

Building New Graphics

We often need to build new statistical graphics by combining basic graphics. A good example is a scatterplot matrix, which consists of scatterplots placed in a square matrix. A histogram and a density estimation graphic are often overlaid together. A statistical graphics library should aid this composition work efficiently using basic statistical graphics.

Interactive Operations

We often wish to focus and highlight particular observations to see the characteristics of them while ignoring other observations. We sometimes use a brushing technique to focus on a group of observations that are neighbors. In addition to brushing, several interactive operations have been proposed and implemented by many authors; see Symanzik (2004). Such operations are usually executed using mouse operations in current computer systems. Therefore, mouse operations (or mouse event handling) should be prepared as basic functions in a graphics library in order to implement interactive operations easily.

Linked Views

Interactive operations are especially powerful if we link several statistical graphics at the same time. It is usual to draw several graphics for one dataset to detect its characteristics. A simple example is again a scatterplot matrix. If we highlight a group of observations in one scatterplot component, it is useful to highlight the same observations in other scatterplot components. This is an example of so-called “linked views” or “linked highlighting” and “linked brushing.” A modern statistical graphics library should support programming for linked views.

Advantages of Java for Programming Statistical Graphics

17.2.2

Object-Oriented Programming Language

Java is an object-oriented programming (OOP) language: almost all things in Java are written as objects (except primitive types). An object is a set of code and data. As related code and data are encapsulated in an object, it is available as a component for other programs. OOP is especially useful for realizing graphical user interfaces and graphics programming, because components in these programs are easily implemented by intuitive objects. For example, a point on a window is directly realized by an object that has a location, size and color as data, and procedures for changing values of these components as code.

Java adopts a class-instance mechanism, where a class is a kind of a type definition and instances are realizations of classes. Classes have an inheritance mechanism that is used for so-called “difference” programming. This means that a new class can be defined by adding information to an existing class.

Platform Independence

Programs written in the Java language run similarly on many platforms. This is achieved through the Java virtual machine mechanism. Although this approach has overhead for executing Java programs, recent Java virtual machines have mechanisms to reduce it.

This mechanism also permits the provision of standard libraries for accessing features of host platforms, such as graphics, threading and networking, in unified ways.

Abundant Standard Libraries

As Java is not dependent on any specific platform, it provides a set of standard class libraries that contain abundant common reusable functions for various modern operating systems. They support a wide range of tasks from basic ones such as network access and file access to advanced tasks such as interactive GUI building. They are indispensable for statistical graphics programming.

It is important that standard libraries of Java are all written and controlled by the original developer. It often happened that many similar libraries were provided for one language and users then found it difficult to choose between them. As Java has

established standard libraries, users can use them without any future compatibility problems.

Because of its OOP design, it is easy in Java to use other nonstandard libraries written by many authors. There are several reliable libraries that are essential for a statistical graphics library, such as a library for data import and export using the Excel format (Jakarta POI), and a library for outputting graphics using the SVG format (Batik SVG Toolkit).

Design Patterns

OOP is an efficient programming paradigm for achieving extensibility and reusability of programs. Recently, several approaches to writing good object-oriented programs have been proposed as “design patterns.” Design patterns describe how objects communicate without being entangled in each other’s data and code.

Java developers use them extensively, and many standard libraries have been developed by following them. When we use Java standard libraries, we can see and learn what design patterns advocate.

17.2.3

Basics of Java Graphics

Recent Java graphics programming mainly uses the Java 2D and Swing libraries. We note that Java libraries are documented by the Java Application Programming Interface (API). API is a list of Java packages, classes, and interfaces with all of their methods, fields and constructors, and also with information on how to use them.

The first API for graphics programming was the Abstract Window Toolkit (AWT). Java 2D API provides the `Graphics2D` class (a subclass of the `Graphics` class in AWT), which contains a much richer set of drawing operations.

Swing is a package that provides a set of lightweight and enhanced components that replace the ones available from AWT. For example, the `JButton` class enhances the `Button` class in AWT in order to allow not just text but images too on a button.

Java 2D and Swing are included in Java Foundation Classes (JFC), which is a set of APIs intended to be a supplement to AWT. The Accessibility API and the Drag and Drop API are also included in JFC.

An Example of Java Graphics Programming

As an example of using Java 2D and Swing API, we now list a short program that draws line segments and points in a window on the screen.

```
import java.awt.Color;
import java.awt.Graphics2D;
import java.awt.Graphics;
import java.awt.geom.Ellipse2D;
import java.awt.geom.Line2D;
import javax.swing.JFrame;

public class GraphicExample extends JFrame {
    public static void main(String[] args){
        GraphicExample ge = new GraphicExample();
        ge.setDefaultCloseOperation(EXIT_ON_CLOSE);
```

```

        ge.setTitle("Java2D Graphic Example");
        ge.setBounds( 0, 0, 300, 300);
        ge.setVisible(true);
    }
    public void paint(Graphics g){
        Graphics2D g2d = (Graphics2D)g;
        g2d.clearRect(0, 0, getWidth(), getHeight());
        g2d.draw(new Line2D.Double(30, 250, 140, 100));
        g2d.draw(new Line2D.Double(140, 100, 240, 50));
        g2d.fill(new Ellipse2D.Double(30-3, 250-3, 7, 7));
        g2d.fill(new Ellipse2D.Double(140-3, 100-3, 7, 7));
        g2d.fill(new Ellipse2D.Double(240-3, 50-3, 7, 7));
    }
}

```

When we draw statistical graphics using Java 2D, the `java.awt.geom` package is useful. It includes the `Line2D.Double` class in order to specify a line segment by double x-y coordinates, and the `Ellipse2D.Double` class to specify an ellipse by several double values. These objects are drawn on a `java.awt.Container` object by the `draw` or the `fill` methods in the `java.awt.Graphics2D` class.

We first define a `GraphicExample` class by extending the `JFrame` class in Swing. In the `main` method, we create an instance of this class and set all required information, such as a title and the size of the window.

Drawing tasks are defined in the `paint` method. This method is automatically invoked when a window is created or the `repaint` method is required. We draw two line segments and three points. A point is defined as an ellipse specified by the location (the x-y coordinates of the upper-left corner of the framing rectangle) and the size (the width and the height of the framing rectangle). Figure 17.1 is the result of this program.

GoF Design Patterns

17.2.4

It is true that OOP is useful for realizing extensibility and reusability of programs if we design class structure properly. However, it is not an easy task to design software based on good class structure.

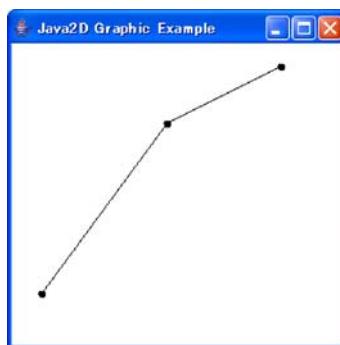


Figure 17.1. Java 2D graphic example

Gamma et al. (1995) discusses 23 “design patterns” as a practical catalog for writing good programs using basic ideas of OOP such as inheritance, polymorphism, and encapsulation.

As the book was written by four authors, they are called the “Gang of Four” (GoF), and their 23 design patterns are sometimes called the “GoF design patterns.” The patterns are divided into three categories: creational, structural, and behavioral.

Creational patterns propose the best way to create instances of objects. Using simply the new method to create an instance is not appropriate in many cases. Creational patterns abstract the creation process into a special creator class.

Structural patterns describe how classes and objects can be combined to form larger structures. Patterns for classes describe how inheritance can be used to provide more useful program interfaces. Patterns for objects describe how objects can be composed into larger structures using object composition.

Behavioral patterns are concerned with communication between objects.

Some of these 23 design patterns are illustrated individually in the next section with statistical graphics examples.

Roughly speaking, these design patterns do not recommend the use of inheritance. Instead, they recommend the use of an interface to realize polymorphism, and to use composition to reuse existing classes. These are quite different from the historical OOP approaches. These ideas are mainly based on several basic principles (Martin, 2003).

The Open-Closed Principle

This principle states that classes should be open for extension, but closed for modification. In other words, it should be possible to extend the functions of a class, but it should not be possible to modify source code.

If we can extend or change the functions of a class by just adding new source code without checking old source code, the class is designed correctly according to this principle.

The Dependency Inversion Principle

This principle insists that high-level modules should not depend upon low-level modules, and both should depend upon abstractions. It also insists that abstractions should not depend upon details and details should depend upon abstractions.

In Java, abstractions are realized by interfaces and abstract classes. Thus, this principle recommends that we write programs by using the methods defined in them.

Liskov Substitution Principle

This principle insists that every method that operates on a superclass should be able to operate on subclasses without knowing them. In other words, any method that uses a superclass must not be confused when a subclass is substituted for the superclass.

If this principle is violated, a method that operates on a variable that points to an object must first check the type of the actual object in order to work correctly.

Inheritance Versus Composition

When class-based OOP appeared, the inheritance mechanism was used to reuse code and to realize dynamic binding and polymorphism. A subclass can use all code in its superclass just by inheritance. In other words, the code in the superclass is reused by the subclass easily.

Dynamic binding means that the method implementation that is actually invoked is decided at runtime. Polymorphism means that one method can handle objects belonging to different types in order to respond differently according to the right type-specific behavior. Polymorphism is implemented at runtime using dynamic binding. The advantage of dynamic binding and polymorphism is that they make the interface of a method simple and make the code easier to change.

We can also realize code reuse, polymorphism and dynamic binding by using composition, which means the use of instance variables that are references to other objects.

Generally speaking, composition is better than inheritance for realizing reusability and polymorphism. In an inheritance relationship, superclasses are often fragile, because a change to an interface of superclass can require changes in many other code segments that use the superclass or any of its subclasses. Thus, inheritance should be used only when a subclass “is-a” superclass. If we want to reuse code or to realize polymorphism, and there is no natural is-a relationship between objects, we should use composition.

MVC Design Pattern

17.2.5

The Model–View–Controller (MVC) design pattern is not included in GoF design patterns. It is, however, an important design pattern (or framework), especially for graphics programming. It was originally developed for the GUI of the Smalltalk system (Krasner and Pope, 1988) and has been widely used in many systems, including Java. The MVC pattern breaks an application into three parts: a model, a view and a controller.

A model is used to manage information and notify observers when information changes. It contains only data and functionality that are related by a common purpose. A model is meant to serve as a computational approximation or abstraction of some real world process or system. It captures not only the state of a process or system, but how the system works.

A view is responsible for mapping graphics of a model onto a device. A view has a one-to-one correspondence between a region of a display surface and a model, and knows how to render the contents of a model to the display surface. When the model changes, the view automatically redraws the affected part of the image to reflect those changes.

A controller is the means by which the user interacts with the application. A controller accepts input from the user and instructs the model and view to perform actions based on that input. In effect, the controller is responsible for mapping end-user action to application response. For example, if the user clicks the mouse button or

chooses a menu item, the controller is responsible for determining how the application should respond.

17.2.6

Class Diagrams in UML

We explain class diagrams in Unified Modeling Language (UML), a tool for displaying class structure intuitively, for each design pattern described in the next section. UML is a standard language for specifying, visualizing, constructing, and documenting software systems, as well as for business modeling and other non-software systems (Hunt, 2003; Virius, 2004). UML uses mostly graphical notations to express the design of software projects. UML 2.0 defines 13 diagrams. They are designed to let developers and customers view a software system from a different perspective and in varying degrees of abstraction. We use only class diagrams among them to illustrate design patterns in our Java graphics library.

Class diagrams describe types of objects in a system and their relationships. The fundamental element of the class diagram is an icon that represents a class. This icon is simply a rectangle divided into three compartments. The topmost compartment contains the name of the class. The middle compartment contains a list of fields, and the bottom compartment contains a list of methods. We usually show selected fields and methods. In many cases, the bottom two compartments are empty.

We can add visibility markers to signify who can access the information contained within a class. For example, private visibility (marked by `-`) hides information from anything outside the class partition, and public visibility (marked by `+`) allows all other classes to view the marked information.

If the operations are shown in italics, it indicates that they are purely virtual and have no implementation there.

The inheritance relationship in a class diagram is depicted by a triangular arrowhead. This arrowhead points to a superclass icon and the base of the arrowhead comes from the subclass(es) (see Fig. 17.2).

An interface is depicted by a triangular arrowhead with a broken line segment. We add `<<interface>>` above the name of the interface (see Fig. 17.3). `<<interface>>` is a stereotype in UML. Stereotypes provide a way of extending UML and produce new kinds of model elements. A stereotype name is written above the class name and is enclosed in guillemets.

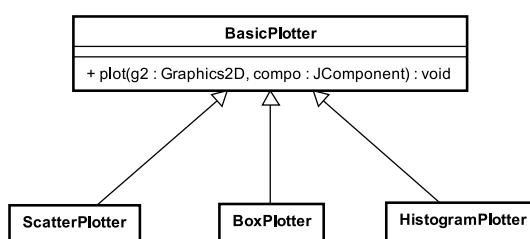


Figure 17.2. Inheritance in a class diagram

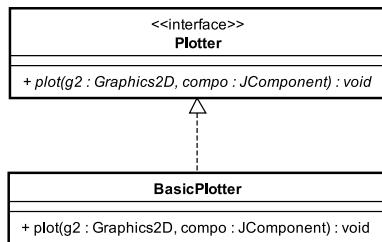


Figure 17.3. Interface in a class diagram

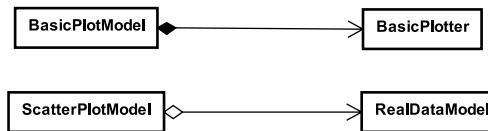


Figure 17.4. Composition and aggregation in class diagrams

In UML, composition indicates that one class B belongs to another class A. Class B is the “part” class of the “whole” class A. If “whole” class A is destroyed, so is “part” class B. In a class diagram, class A and class B are connected by a line segment which ends at the “whole” class A with a filled diamond.

Aggregation is similar to composition, but is a less rigorous way of grouping things. Let class A aggregate class B. At this time, “part” class B continues to exist even if “whole” class A is destroyed. In a class diagram, “whole” class A is pointed to by an empty diamond (see Fig. 17.4).

Design and Implementation of a Java Graphics Library

17.3

In this section, we illustrate the details of the techniques described above using a statistical graphics library, Jasplot (JAvA Statistical PLOT), which was originally written for our general-purpose statistical software Jasp (JAvA-based Statistical Processor) (Nakano et al., 2000; Yamamoto et al., 2002; Nakano et al., 2004). Jasplot has since been improved from the original version so that it is now independent of Jasp.

Overview of Jasplot

17.3.1

Jasplot is a Java library for drawing interactive statistical graphs. It is supposed to be used from programs written by users and to be used from other software like Jasp. In the Appendix, we briefly explain how to use Jasplot from Java programs. At this stage, Jasplot is integrated into Jasp, which has its own language (Kobayashi et al., 2002). All of the functions of Jasplot can easily be used by writing Jasp programs in Jasp. The Jasp language is an extension of Pnuts, a script language. As script languages (such as Pnuts, Python and Ruby) are designed to write programs easily, Jasp is the easy way to utilize Jasplot.

Jasplot has basic graphics classes such as scatterplots (Fig. 17.5), histograms (Fig. 17.6) and boxplots (Fig. 17.7). It implements scatterplot matrices (Fig. 17.8) and parallel coordinate plots (Fig. 17.9) as examples of combining basic classes to build new graphics. It also implements mosaicplots (Fig. 17.10). These graphics have functions for focusing, zooming and linked views.

Jasplot was written by adopting the MVC design pattern. Each basic statistical graphic is written as three main classes that implement the DataModel, PlotModel and Plotter interfaces. DataModel realizes the model of the MVC pattern for a graphic. The view of the MVC pattern consists of PlotModel and Plotter. The controller of the MVC is realized by the JasplotPanel class, in which graphics are drawn and mouse events are handled.

Jasplot also uses several GoF design patterns to realize reusability and extensibility. We illustrate them by building several new graphics such as scatterplot matrices or parallel coordinate plots.

Jasplot realizes useful interactive operations. Jasplot has three different and exchangeable selectors for specifying particular observations. When the numbers of data and variables are large, interactive operations can become slow. To reduce computational burdens and to increase response speed, Jasplot adopts several mechanisms such as control over repaint timing, multilayers and double buffering.

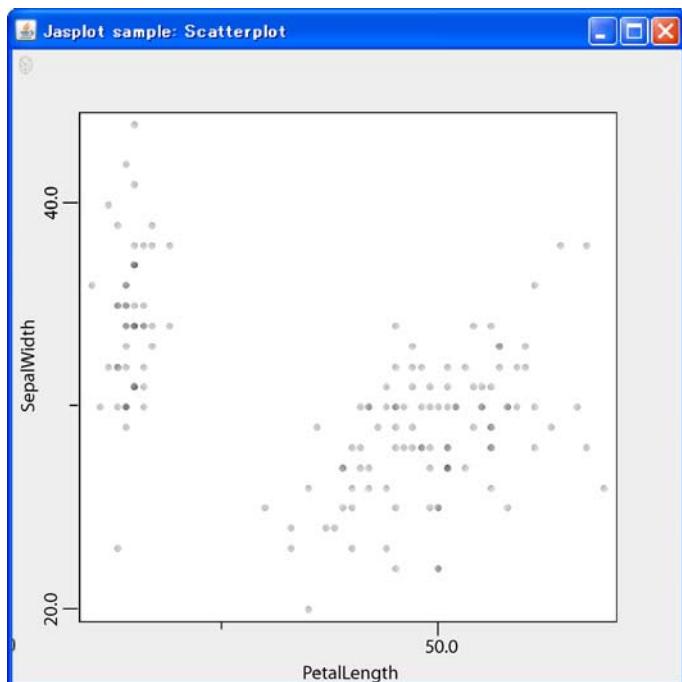


Figure 17.5. Scatterplot example

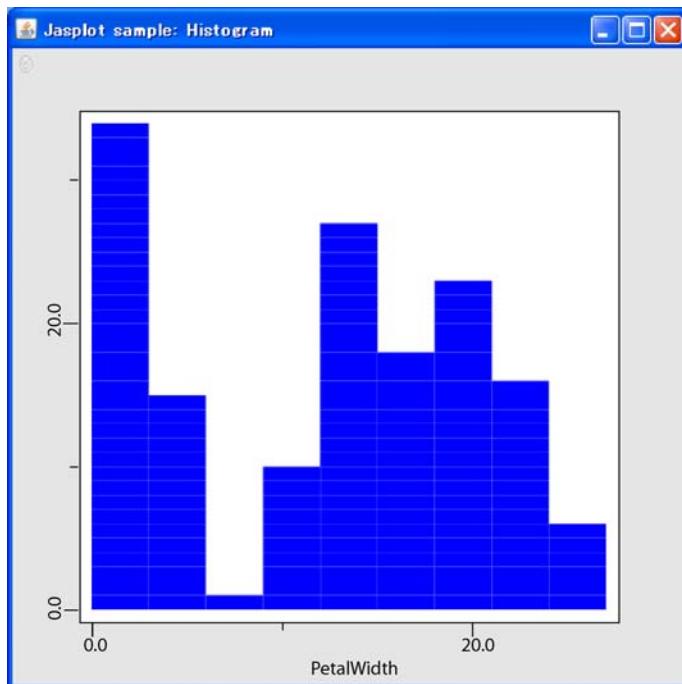


Figure 17.6. Histogram example

All objects in Jasplot are serializable. In other words, they can be converted into a byte stream and transmitted over a network.

Java supports printing as standard functions, so Jasplot graphics are printable.

Summary of Basic Interfaces and Classes with an Example

17.3.2

We show the names of basic interfaces and classes of Jasplot. Some of them are explained in detail later.

- Classes for original data
DataModel, RealDataModel, CSVDataModel, ExcelDataModel
- Classes for data about graphics
PlotModel, BasicPlotModel, ScatterPlotModel, HistogramPlotModel, BoxPlotModel, MultiPlotModel, MosaicPlotModel
- Classes for drawing graphics
Plotter, BasicPlotter, ScatterPlotter, HistogramPlotter, BoxPlotter, MultiPlotter, MosaicPlotter
- Classes of panels for drawing
JasplotPanel, JasplotPanelPaletteLayer, JasplotPanelDragLayer, JasplotPanelPopupLayer, JasplotPanelPalette-

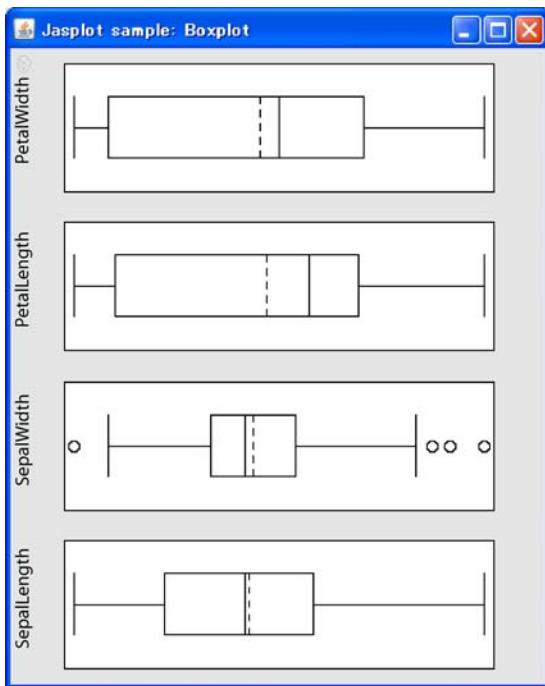


Figure 17.7. Boxplot example

LayerUI, JasplotPanelDragLayerUI, JasplotPanelPopupLayerUI

- Classes for interactive operations
 - Classes for interpreting mouse events

BrushingState, DragGestureState, DragLayerState,
DraggingState, ManipulatingState, MultiPlot-
ManipulatingState, SelectingState, ToolTipState,
ZoomingState
 - Classes for selecting observations

Selector, RectangleSelector, PointSelector,
LineSelector
 - Classes for linked views

PlotModelEvent, PlotModelHandler, PlotModelListener,
DataModelEvent, DataModelHandler, DataModelListener
- Classes for tables of data

TableModel, TableDisplayer

We show a simple example program that uses some of these classes. This program displays a scatterplot for the first two variables of the famous Iris data and produces Fig. 17.5.

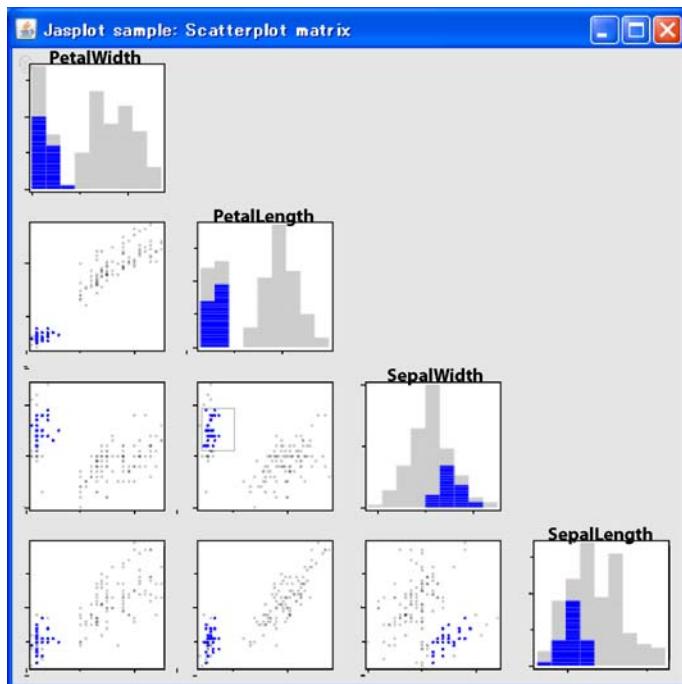


Figure 17.8. Scatterplot matrix example

```
import javax.swing.JFrame;
import jp.jasp.jasplot.CSVDataModel;
import jp.jasp.jasplot.DataModel;
import jp.jasp.jasplot.MatrixDataModel;
import jp.jasp.jasplot.JasplotPanel;
import jp.jasp.jasplot.ScatterPlotModel;
import jp.jasp.jasplot.PlotModel;
public class ScatterPlotSample {
    public ScatterPlotSample() {
        DataModel data =
            new CSVDataModel(new MatrixDataModel(),"../data/iris.csv");
        PlotModel model = new ScatterPlotModel();
        model.setDataModel(data);
        JasplotPanel jasplot = new JasplotPanel(model);
        JFrame jFrame = new JFrame("Jasplot Scatterplot Sample");
        jFrame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        jFrame.getContentPane().add(jasplot);
        jFrame.setSize(500, 500);
        jFrame.setVisible(true);
    }
    public static void main(String[] args) {
        ScatterPlotSample sample = new ScatterPlotSample();
    }
}
```

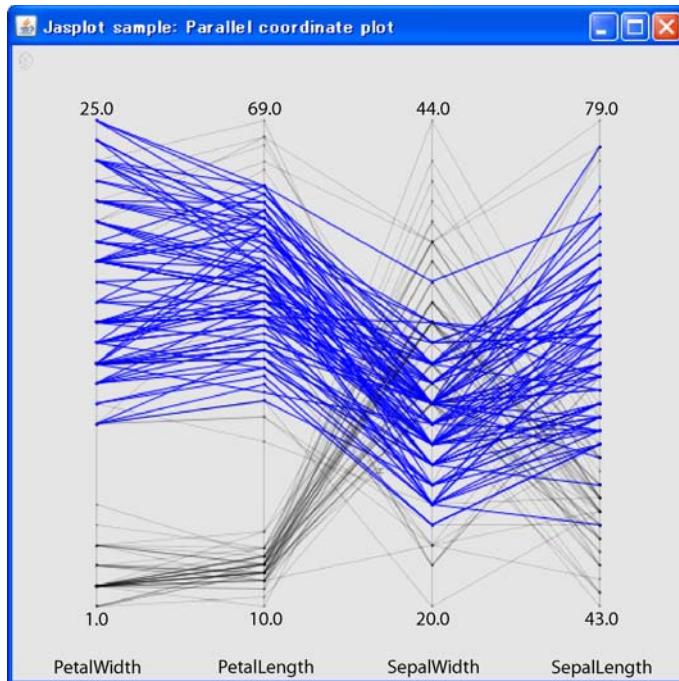


Figure 17.9. Parallel coordinate plot example

The program is not so different from the Java 2D example above. We first create a `DataModel` object. The real object is `CSVDataModel` that implements `DataModel`. Then we create a `PlotModel` object. Note again that the real object is `ScatterPlotModel`. We draw this object on the `JasplotPanel` object. Finally, we show it in a `JFrame` object. We can select some observations by a rectangle selector using mouse operations.

17.3.3 Classes for Original Data

Jasplot provides a general model for describing statistical data by `DataModel`, from which statistical graphics are drawn. `DataModel` is designed as an interface to follow the principles of design patterns, and plays a role as a model in the MVC pattern. `DataModel` expresses a simple two-dimensional table whose rows and columns represent observations and variables, respectively. We define basic methods for getting the values of data (as real values and as string values) and the sizes of observations and variables. Methods for getting the minimum, maximum and median of data values are declared. We also define a method to get a `PlotModelHandler` object, which is used to record selection status for each observation.

`RealDataModel` implements `DataModel` and handles data whose values are all real numbers. The `setReal` method of this class sets real values from a given two-dimensional array.

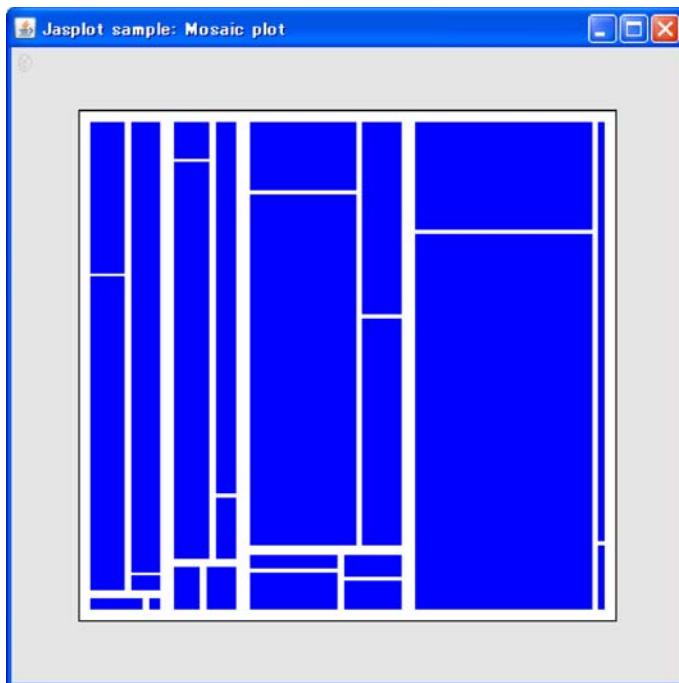


Figure 17.10. Mosaic plot example

We have classes that implement `DataModel` and have components of `RealDataModel` to read data from files. `CSVDataModel` can read data from a file that stores data in the CSV format. `ExcelDataModel` can read a Microsoft Excel data format file. They are almost the same as `RealDataModel` except for the behavior of the `setReal` method. The traditional way of realizing them is to use an inheritance mechanism. However, we use the “Decorator” design pattern to realize these classes using a composition technique.

The decorator pattern attaches additional responsibilities to an object dynamically. Decorators provide a flexible alternative to subclassing for extending functionality (Gamma et al., 1995).

In other words, the decorator pattern provides a class to add new capabilities to an original class, and passes all the unchanged methods of the underlying class to the new class.

We realize `CSVDataModel` by decorating `RealDataModel`. `CSVDataModel` has the `RealDataModel` object in it. `setReal` of `CSVDataModel` reads data from a CSV file and sets them to a two-dimensional real array. Then `setReal` of the `RealDataModel` object is used and data are stored to it. Other methods of `CSVDataModel` are all realized by delegation to the `RealDataModel` object. This is a typical decorator pattern and a typical use of the composition technique. See Fig. 17.11.

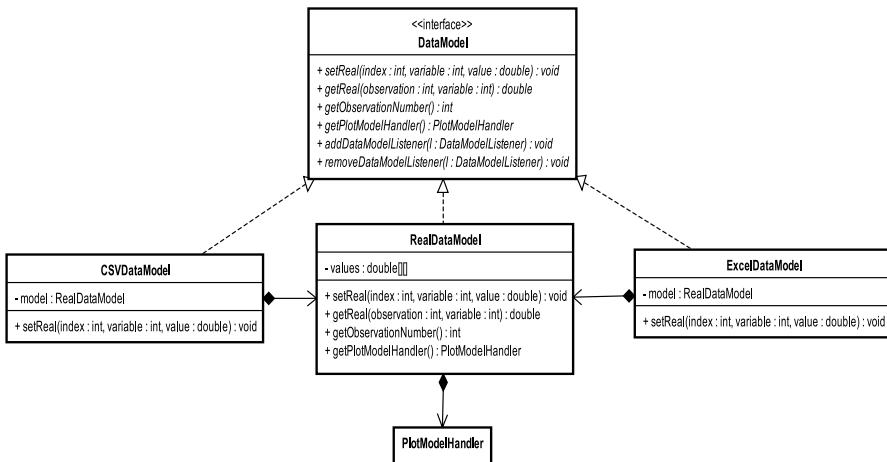


Figure 17.11. Classes for original data

We explain the difference between this realization and the traditional inheritance realization. If we use an inheritance mechanism, all methods of a superclass are automatically available from subclasses. Instead, this realization requires explicit delegation statements, which are additional work. However, explicit declarations for delegation are useful to make the responsibilities of classes that execute methods clear and to realize the rigid software structure.

17.3.4 Classes for Data about Basic Graphics

PlotModel is an interface to set or get information for a particular graphic. As each graphic requires data to be visualized, it declares methods for handling a **DataModel** object: **setDataModel** and **getDataModel**. In order to draw a graphic, detailed drawing procedures are required; they are then declared in a **Plotter** interface separately. **PlotModel** has methods for handling them: **createPlotter** and **getPlotter**.

PlotModel has several methods for interactive operations, such as **getPlotModelHandler**, **setSelector**, and **getSelector**. **PlotModel** also has a method for drawing strings at any location of a graphics: **drawString**. As **PlotModel** does not maintain the size information of graphics, we specify the location of strings by the ratio of **graphicsize** in the **drawString** method.

BasicPlotModel is an abstract class that implements **PlotModel** and gives simple implementations of many of the methods. It also defines several fields to store important objects such as **plotter** for **Plotter**, **selector** for **Selector**, and **title** for the title of the graphics.

To draw a two-dimensional scatterplot, **ScatterPlotModel** is defined by extending **BasicPlotModel**. It has specific information for drawing a scatterplot such as **connect** for indicating whether points are connected or not as a boolean

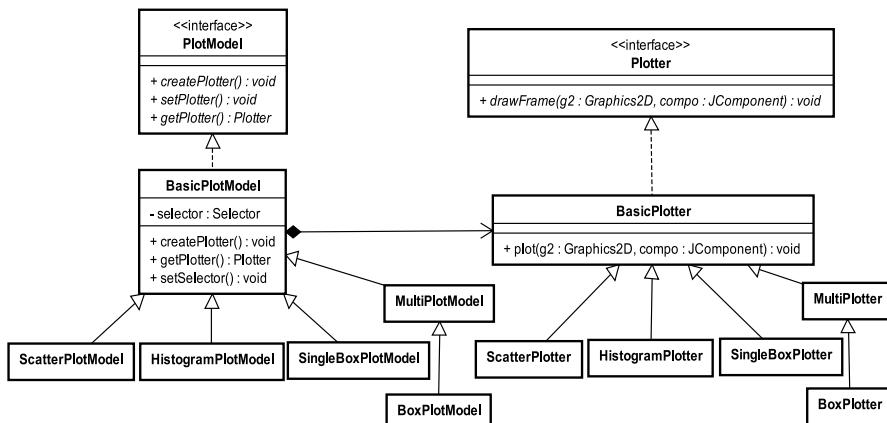


Figure 17.12. Classes for basic graphics

value, lineWidth for storing the width of the connected line segments by a double value, and connectionGroups for recording the group of connected points given by an ArrayList class.

We have similar classes that extend BasicPlotModel and are specific to particular statistical graphics: HistogramPlotModel and SingleBoxPlotModel. See Fig. 17.12.

Classes for Drawing Basic Graphics

17.3.5

Plotter is an interface for plotting and interactive operations. It declares methods such as draw, drawFrame, drawRulers, and plot.

Classes that implement the Plotter interface draw graphics of the corresponding PlotModel in the Java graphics context. For example, ScatterPlotModel accompanies ScatterPlotter, and HistogramPlotModel accompanies HistogramPlotter. They realize methods for drawing particular graphics. The objects of these classes are automatically created by createPlotter methods of the corresponding PlotModel classes. This mechanism is an example of the “Factory method” design pattern.

Factory method defines an interface for creating an object, but let the subclasses decide which class to instance. Factory method lets a class defer instantiation to subclasses (Gamma et al., 1995).

It is often not clear what kind of components we will use at the first stage of programming, although we have a general idea of the operations of certain components. We want to implement the components later. We can achieve this functionality by using interfaces for these components. As an interface has no function to create an object, we use a method for creating an object called the factory method. This technique is useful for creating two or more tightly connected objects.

In Jasplot, a `PlotModel` object always works with a `Plotter` object. However, detailed implementations are not defined at the first stage. We then declare a `createPlotter` method in `PlotModel` and use it in the method

```
public Plotter getPlotter() {
    if (plotter == null) { plotter = createPlotter(); }
    return plotter;
}
```

in `BasicPlotModel`. The concrete implementation of `createPlotter` is given in each specific class. For example, in `ScatterPlotModel`, it is defined as

```
public Plotter createPlotter() {
    return new ScatterPlotter(this);
}
```

Thus, when we use the `getPlotter` method of a `ScatterPlotModel` object, a `ScatterPlotter` object is automatically created. The same things happen with other `PlotModel` objects and `Plotter` objects, such as `HistogramPlotModel` and `HistogramPlotter`. Figure 17.12 shows some of these relations.

The combination of `PlotModel` and `Plotter` plays the role of the view in the MVC pattern. Data expressed by a `DataModel` object can be “viewed” by various graphics that are conceptually described by `PlotModel` objects and are drawn by methods implemented concretely in `Plotter` objects.

17.3.6

Classes of Panels for Drawing

`JasplotPanel` is a base class for displaying graphics defined by a `PlotModel` object. `JasplotPanel` extends the `JLayeredPane` class of Swing components. We can set a `PlotModel` object by using the `setModel` method of a `JasplotPanel` object.

`JasplotPanel` has multilayer functionality, like almost all recent interactive graphical applications. Multilayering means that the visible graphics display shows stacks of virtual transparent layers. Each layer draws some graphics or handles mouse events. This is a useful technology for realizing interactive dynamic graphics.

`JasplotPanel` has three layers. The `JasplotPanelPaletteLayer` displays points, lines and strings to build the main graphics. `JasplotPanelDragLayer` implements mouse event handling. `JasplotPanelPopupLayer` handles pop-up menus. These classes are accompanied by the user interface classes `JasplotPanelPaletteLayerUI`, `JasplotPanelDragLayerUI` and `JasplotPanelPopupLayerUI`, respectively, and they play a controller role in the MVC patterns as a whole. These classes are depicted in Fig. 17.13.

It may take considerable time to redraw `JasplotPanelPaletteLayer` in order to realize dynamic graphics or animations on a screen when the number of data points is large. In this case we need to draw slightly different images promptly and repeatedly. If we draw such images directly to the screen, it probably takes a notable period of time and we see some flickering.

One technique to eliminate these difficulties is double-buffering. This is a technique that repeats two steps: create an offscreen image, then draw it to the screen. Swing uses this technique in many of its components using the `setDouble-`

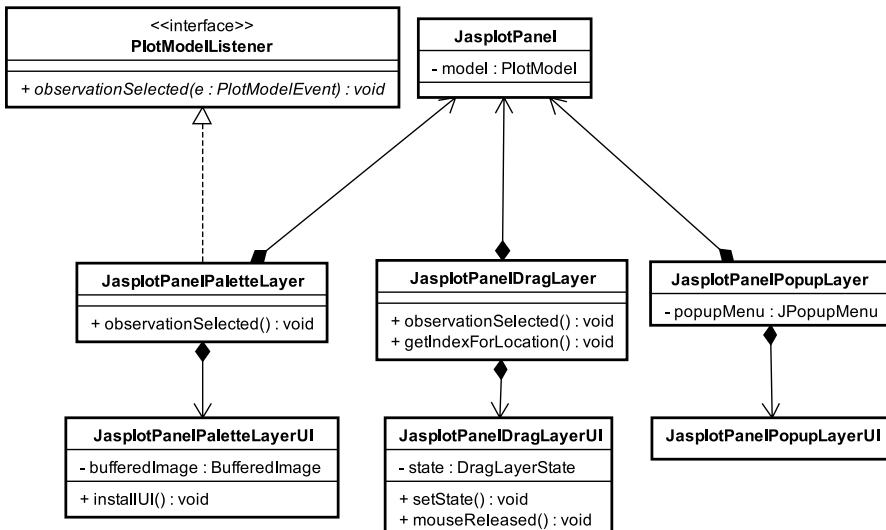


Figure 17.13. Classes of panels for drawing

Buffered method. We can implement them ourselves using thread programming and the `BufferedImage` class in AWT. In the `JasplotPanelPaletteLayerUI` class, we define `PlotImage` to create the buffered image using a thread, and then draw it using the `drawImage` method of the `Graphics2D` object.

Classes for Interactive Operations

17.3.7

We currently mainly use a mouse to indicate graphics operations to the system. Mouse events have to be detected, for example, to check whether the mouse button is pressed or released. The system should then interpret what the user wishes to achieve with that mouse operation. As a mouse can be operated in a limited number of ways, and we wish to use them to send many commands to the system, one mouse operation will have several meanings depending on the context. For example, when we specify a rectangular area on a graphics, we may wish to select observations inside the area, or we may wish to zoom inside the area.

We require different selection methods for different graphics. When selecting observations in a scatterplot, a rectangular selector may be the most natural. It is desirable to draw a rectangular selector on a scatterplot and to select observations inside the selector, and to be able to drag the selector to change the selected observations interactively. Clicking a mouse button on the bin of a histogram may be the natural way to select it. A rectangular selector is also useful to select several neighboring bins in a histogram.

We also know that linked views or linked highlighting and linked brushing are important in interactive operations for statistical graphics.

Classes for Capturing Mouse Events

Interactive mouse or key operations can be implemented effectively by event-driven programming. We call operations such as pressing a mouse button or a key events. Event-driven programming should constantly check for the occurrence of events and perform prescribed executions when a particular event happens.

In the `JasplotPanelDragLayerUI` and `JasplotPanelPopupLayerUI` classes, we define a `MouseInputHandler` class that implements a `MouseListener` defined in Swing to detect mouse events. When this class creates its object, the system starts to check for the defined mouse events.

Classes for Interpreting Mouse Events

We wish to interpret mouse events differently according to context. This is efficiently realized by applying the “State” design pattern.

State pattern allows an object to alter its behavior when its internal state changes. The object will appear to change its class (Gamma et al., 1995).

The State pattern defines a context class to present a single interface to the outside world. It defines an abstract state base class and represents different states as subclasses of it. We then define state-specific behavior in the appropriate state subclasses. We maintain a private reference to the current state in the context class, and change the state by changing the current state reference.

In Jasplot, a `DragLayerState` interface characterizes state classes. It declares methods to decide the behavior of mouse events, such as `mousePressed`, `mouseMoved`, `mouseDragged` and `mouseReleased`. They are implemented by classes such as `BrushingState`, `DragGestureState`, `DragLayerState`, `DraggingState`, etc. One of these classes is set to state field in the `JasplotPanelDragLayerUI` class and executed by, for example, the call `state.mousePressed(MouseEvent e)`. Thus, the `mousePressed` method causes differ-

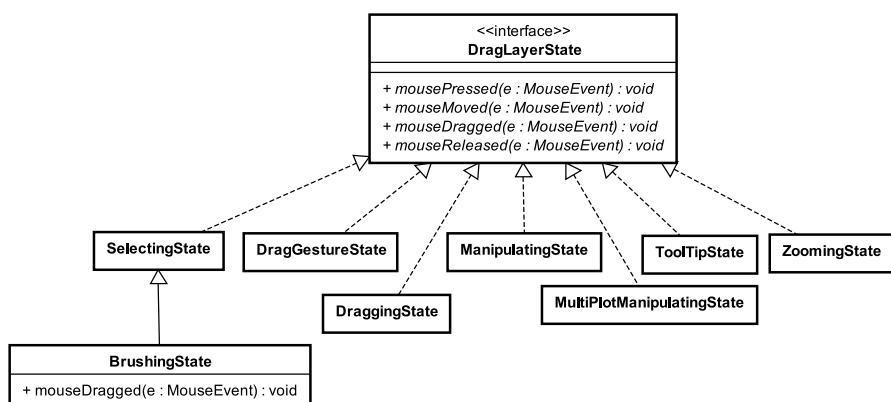


Figure 17.14. Classes for interpreting mouse events

ent behavior depending on the object selected by the state field. Figure 17.13 and 17.14 partly show these relations.

The SelectingState class can select observations. It draws a selector at the position specified by a mousePressed event. mouseDragged events can change the size of the selector. A mouseReleased event selects the observations inside the selector; more precisely, a mouseReleased event generates a PlotModelEvent object, sets the information of selected observations to it, and sends the object to the PlotModelHandler object to repaint the graphics.

The BrushingState class inherits the SelectingState class, and has a different implementation for handling a mouseDragged event. In the BrushingState class, mouseDragged events cause the same behavior as the mouseReleased event in the SelectingState class.

The DraggingState class enables the movement of selected observations by dragging the selector. This operation changes the values of selected observations temporarily. The resetDragging method can recover their original values again.

The ManipulatingState class can reverse the directions of the *x* and *y* axes. When this class is set to the state field in the JasplotPanelDragLayerUI class, a frame appears around the graphic and this can be used to specify the directions of the axes.

The MultiPlotManipulatingState class is available for graphics placed by the MultiPlotModel class that is explained in Sect. 17.3.9 later. It works like the ManipulatingState class and can swap the locations of graphics by dragging an accompanying frame.

The ZoomingState class is used to zoom the region that is specified by the mousePressed and mouseReleased events. The resetZooming method can reset the operation to the original size.

In Jasplot, strings drawn on graphics can show tooltips, which are small pop-up windows to show strings. They are useful, for example, for showing each variable name clearly when several variable names overlap and are difficult to identify on the original graphics. We can see each variable name by placing the mouse cursor on the variable name strings and presenting a tooltip on which the specified variable name is displayed. This function is realized by the ToolTipState class. The BasicPlotter class has several methods to draw strings on graphics such as drawStringCenter and drawStringLeftUpper. When we use them, the BasicPlotter class records the position of the strings as a pair consisting of a Rectangle2D object and a String object. The mouseMoved method in the ToolTipState class can check whether the position of the mouse cursor is inside that Rectangle2D object or not. If the mouse cursor is inside one of these Rectangle2D objects, the corresponding String object is displayed by the setToolTipText method in the JasplotPanel object.

Drag & drop operations are also available for strings on graphics using the DragGestureState class. Strings can be dragged and dropped onto graphics components defined by the DroppableTextField and DroppableComboBox classes. This function can be used to specify a variable name on a graphic by performing a drag & drop operation for a variable name on another graphic.

Classes for Selecting Observations

In order to select observations on graphics, we need to select objects that represent observations such as points in a scatterplot, rectangular bins in a histogram or line segments in a parallel coordinate plot.

A draggable selector is one tool for selecting such objects. We first specify a selector such as a rectangular box, then select objects inside the selector. We may wish to move the selector by mouse dragging and to change selected objects dynamically. Such a selector is realized by the `RectangleSelector` class in Jasplot. It may be preferable to specify the shape of selector region freely by a mouse cursor, but this is not yet realized in Jasplot.

We wish to select one bin of a histogram by pointing at it with the mouse cursor and clicking a mouse button. This is realized by the `PointSelector` class. This class can also select objects that are located inside a rectangular area specified by pressing a mouse button, dragging the mouse cursor and releasing the mouse button. The rectangle area disappears after we release the mouse button.

To select line segments, for example in a parallel coordinate plot, we can use the `LineSelector` class, which is a vertical line segment that selects line segments representing observations that intersect the selector.

We sometimes wish to replace these selectors according to the purpose of our analysis. Such replacement is easily realized by following the “Strategy” design pattern.

The Strategy pattern defines a family of algorithms, encapsulates each one, and makes them interchangeable. Strategy lets the algorithm vary independently from the clients that use it (Gamma et al., 1995).

The strategy pattern consists of decoupling an algorithm from its host, and encapsulating the algorithm into a separate class. When we have several objects that are basically the same, and differ only in their behavior, the strategy pattern is useful. We can reduce these several objects to one class that uses several strategies.

We first implement a strategy interface for strategy objects. We then implement concrete strategy classes that implement the strategy interface. In the context class, we maintain a private reference to a strategy object. Then context class implements public “setter” and “getter” methods for the strategy object. We can see that the strategy pattern satisfies the open–closed principle. The context class can be extended by adding new strategy objects without changing any code inside the context class.

The strategy and state patterns can be confused. The strategy pattern is better if the context will contain only one of several possible state/strategy objects. On the other hand, the state pattern is better if the context may contain many different state/strategy objects. An object is usually put into a state by an external client, while it will choose a strategy on its own.

In Jasplot, we use the `Selector` interface to define common methods for all selectors such as `create`, `getRegion`, and `mouseReleased`. `RectangleSelector`, `PointSelector` and `LineSelector` implement `Selector`. Figure 17.15 shows a class diagram for these classes.

For example, `HistogramPlotModel` uses `PointSelector` as default. This is realized by a `setSelector(new PointSelector)` method call in the con-

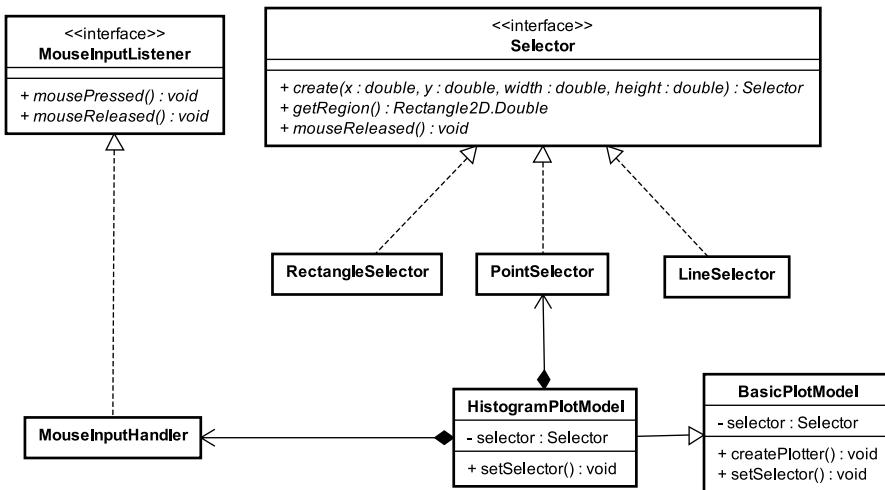


Figure 17.15. Classes for selecting observations

structor. We can easily replace a selector by using `RectangleSelector` instead of `PointSelector`. This is the advantage of the strategy pattern. The context (abstract) class `BasicPlotModel` has a `selector` field in order to reference a concrete strategy class such as `PointSelector`. We always use methods of the strategy class through the `selector` field, for example, `selector.create`.

We now explain how the `PointSelector` object is used in `HistogramPlotModel`. When we press the mouse button on the histogram graphic, the `mousePressed` method in the `MouseInputHandler` class that implements the `MouseInputListener` interface defined in `JasplotPanelDragLayerUI` detects the event. The `mousePressed` method calls the `state.mousePressed` method, and then it calls the `createSelector` method and executes the `selector.create` method to create a `PointSelector` object. When we release the mouse button (after dragging), the `mouseReleased` method in the `MouseInputHandler` class detects the event, and the `mouseReleased` method calls `state.mouseReleased`, and it executes the `mouseReleased` method in the `SelectingState` class. This method performs the `getIndexForLocation` method that checks whether representations of all observations are included in the selector box just defined or not. Selected observations are drawn in different colors by the `repaint` method. See Fig. 17.15.

As a selection operation is dynamic, the graphic must change dynamically. We must then be careful about the time taken to repaint. If the number of observations is huge, repainting will require a considerable period of time. In Jasplot, we can choose between two approaches to repainting. One is real-time repainting; that is, repainting many times during mouse operation. This may be natural if we use `RectangularSelector` and drag it. We wish to select observations in the selector box at each moment of dragging. If we drag the selector box and one selected ob-

servation moves outside the selector box, we expect that the observation is deselected as soon as possible. This type of repainting requires much computation and works for only a limited number of observations. Another natural approach is to repaint when the mouse button is released. If we use `PointSelector` and specify a rectangular area, it is reasonable to repaint at the last moment of specifying the selector box; that is, when the mouse button is released. This approach is computationally light and appropriate when the number of observations is huge. We can choose these timings by using the `setDynamicChange` method defined in the `JasplotPanel` class. If we execute `setDynamicChange(true)`, real time repainting is executed. If we execute `setDynamicChange(false)`, repainting is performed when we release the mouse button.

Classes for Linked Views

We have explained mechanisms of selecting or brushing observations in a single statistical graphic. It is also important, however, to connect such operations across several graphics; that is, to realize linked views or linked highlighting and linked brushing. To do this, we must inform other graphics of changes in selected observations in one graphic. This can be realized by the “Observer” and “Mediator” design patterns.

The Observer pattern defines a one-to-many dependency between objects so that when one object changes state, all its dependants are notified and updated automatically (Gamma et al., 1995).

The observer pattern allows one object (the observer) to watch another (the subject). It forms a publish–subscribe relationship between the subject and the observer. Observers can register to receive events from the subject. When the subject needs to inform its observers of an event, it simply sends the event to each observer.

In Jasplot, the `PlotModelHandler` object, accompanied by the `DataModel` object, behaves as a subject. Observers are other graphics’ `JasplotPanel-PaletteLayer` objects that implement the `PlotModelListener` interface. Observers can be registered by using the `addPlotModelListener` method in the `PlotModelHandler` class. See Fig. 17.15.

Consider that selected observations are changed by a selector operation. The `SelectingState` object creates a `PlotModelEvent` object in which the selected/unselected information for each observation is stored, and sets it by using the `setSelectedObservation` method in the `PlotModelHandler` class. It informs registered `JasplotPanelPaletteLayer` objects of the new `PlotModelEvent` object. It executes the `observationSelected(PlotModelEvent e)` method in the `JasplotPanelPaletteLayer`. As this method executes the `repaint` method, each `JasplotPanelPaletteLayer` object is repainted according to the new information.

We know that `PlotModelEvent`, `PlotModelHandler` and `PlotModelListener` are associated with changes in the `PlotModel` object. There are similar classes, such as `DataModelEvent`, `DataModelHandler` and `DataModelListener`, which can handle changes in `DataModel` objects; that is, changes in data values, the addition and deletion of observations and variables. A `DataModel-`

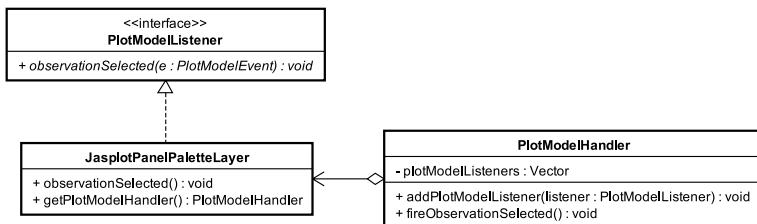


Figure 17.16. Classes for linked views

Handler object is generated together with a DataModel object that is a subject. PlotModel objects that implement the DataModelListener interface are observers. Observers are registered by a addDataModelListener method in the DataModelHandler class.

PlotModelHandler and DataModelHandler are also examples of the “Mediator” in the “Mediator” design pattern.

The Mediator pattern defines an object that encapsulates how a set of objects interact. A mediator promotes loose coupling by keeping objects from referring to each other explicitly, and it lets you vary their interaction independently (Gamma et al., 1995).

We sometimes notice that many objects need to communicate with each other. Such mutual interactions may prevent an object from working without the support of many other objects. The mediator pattern is a useful approach to use to solve such a mess. The mediator object provides a common connection point, centralized behavior, and behavioral management. The mediator pattern promotes loose coupling by keeping objects from referring to each other explicitly. Objects do not need to know about each other and only need to know their mediator. Several JasplotPanel-PaletteLayer objects communicate with a PlotModelHandler mediator object. Figure 17.16 shows these classes.

Classes for Tables of Data

17.3.8

We sometimes wish to display a table of data that can be linked with graphics. Jasplot has two classes for this purpose: TableModel and TableDisplayer. The TableModel class is similar to PlotModel, and the constructor method requires a DataModel object as an argument. The TableDisplayer class is similar to Plotter and displays an TableModel object on the screen by using the JTable class in Swing. As it implements a PlotModelListener interface, the rows specified by the observationSelected method are highlighted and linked with other graphics of Jasplot via a PlotModelHandler.

A Class for Building Complicated Graphics

17.3.9

Jasplot provides a MultiPlotModel class for drawing several graphics on one panel. This class extends the BasicPlotModel class and is inherited by the Box-

PlotModel class that is composed of several SingleBoxPlotModel objects. Scatterplot matrices and parallel coordinate plots are implemented using this class.

The `MultiPlotModel` class defines a simple matrix whose elements are graphics. It has a `setRowColumn(row, column)` method to specify the numbers of rows and columns. The window defined by a `MultiPlotModel` object is divided into `row×column` small components of the same size. Graphics are placed on each component by the `setPlotModel(model, row, column)` method in `MultiPlotModel`, where `model` specifies a `PlotModel` object, and `row` and `column` specify the location of the component. It is possible to overlay several graphics on the same component. However, mouse operations are effective only for the graphics overlaid last.

Our scatterplot matrix is realized by placing scatterplots on off-diagonal components and histograms on diagonal components. The main part of the program is straightforward:

```

for (int i = 0; i < dataModel.getVariableNumber(); i++) {
    for (int j = 0; j < i; j++) {
        PlotModel pModel =
            new ScatterPlotModel(dataModel, j, i);
        multiPlotModel.setPlotModel(pModel, i, j);
    }
}

HistogramPlotModel hModel =
    new HistogramPlotModel(dataModel, i);
hModel.setTitle(dataModel.getVariableName(i));
multiPlotModel.setPlotModel(hModel, i, i);
}

```

Another example is a parallel coordinate plot. The main part of the program is the following:

```

plotModel.drawString(String.valueOf(dataModel.getMax(i)),
                    0.5, 0.9, DrawString.CENTER, null);

plotModel.setXAxisLabel(plotModel.getYAxisLabel());
plotModel.setDrawYAxisLabel(false);
plotModel.setDrawFrame(false);
plotModel.setDrawXZeroLine(true);
multiPlotModel.setPlotModel(plotModel);
}

```

We generate a `MultiPlotModel` object for a `DataModel` object specified by `dataModel`. We divide it into $1 \times (\text{number of variables})$ regions. The `setConnect(true)` method of the `MultiPlotModel` class enables us to connect the points for the same observation. `setSelector(new LineSelector())` specifies the use of a `LineSelector` object. In the next `for` command, we place scatterplots generated by the new `ScatterPlotModel(dataModel, PlotModel.NULL, i)` method for each variable. We specify that no data are available for the x -axis, and the i -th variable of `dataModel` is used as data for the y -axis. The methods `setDrawXTick(false)` and `setDrawYTick(false)` specify that the x - and y -axes are not drawn. Next, several lines set and draw the maximum values of the variables at the top of the y -axis and the minimum values at the bottom. The `setXAxisLabel(plotModel.getYAxisLabel())` method draws a variable name given for the y -axis at the place where the label for the x -axis is drawn. `setDrawYAxisLabel(false)` does not draw a variable name for the y -axis, `setDrawFrame(false)` does not draw a frame for the scatterplot, and `setDrawXZeroLine(true)` draws a line at the 0 position of the x -axis. We do not set the location in the `setPlotModel(plotModel)` method, because graphics are automatically placed from left to right when they are not given.

This type of parallel coordinate plot usually becomes long horizontally. In this case, it may be useful to attach a `JasplotPanel` object to a `JScrollPane` object of Swing, which has a scroll bar and can show a part of the whole graphic in the window.

We can attach these `MultiPlotModel` classes to a `MultiPlotModel` class recursively. For example, Fig. 17.17 shows a scatterplot matrix and a parallel coordinate plot on a `MultiPlotModel` object.

Concluding Remarks

17.4

By following design patterns, Java statistical graphics programming is made extensible and reusable. Such patterns are useful for building new graphics from existing components in the library. We illustrate the usefulness of these solutions with `Jasplot`, a Java statistical graphics library.

Note that we did not use all of the 23 GoF design patterns to build `Jasplot`. We should not use design patterns excessively in a project; this would impact negatively on the simplicity and execution speed of the program. However, it is a good idea to use design patterns appropriately. If it is possible to use design patterns with a little

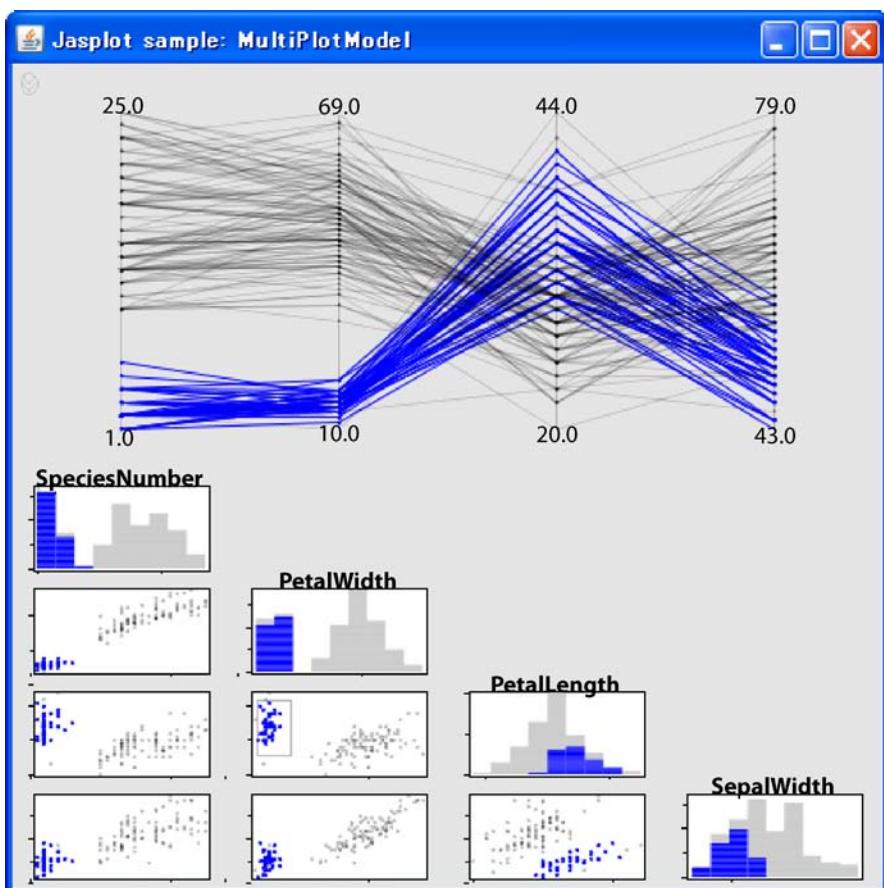


Figure 17.17. MultiPlotModel example

additional programming effort and a little decrease in execution speed in some parts of the project, it is worth adopting them.

Jasplot still lacks several important functions associated with data visualization. For example, the `MultiPlotModel` class should be able to divide the whole window into components of unequal size. Color handling in Jasplot is not fully realized.

We note that three-dimensional graphical functions are often used in modern data visualization (Symanzik, 2004). The Java 3D API seems a promising approach to realizing them. We plan to include them in a future version of Jasplot.

Appendix: Using the Jasplot Library

Jasplot is a Java library developed using Java2 SDK, Standard Edition (Version 1.5 or later), which is freely available from <http://java.sun.com/products/>. It is required to use Jasplot.

Jasplot is available from our site <http://jasp.ism.ac.jp/Jasplot/>. All source code, class files and jar files are provided as one zip file Jasplot-1_0_0.zip (the version number may vary). Unpack it by using an unpack tool and go into the Jasplot-1_0_0 directory. You can see demo by executing the jasplot-demos.jar file by, for example, double-clicking it. The menu that appears enables you to choose graphics from Figs. 17.6 – 17.10 and 17.17.

We use a Java-based build tool, Apache Ant (Version 1.6.5 or later), which is freely available from <http://ant.apache.org/> for developing Jasplot. If you install it, you can rebuild the jar file by executing the commands

```
ant  
ant demo
```

in the Jasplot-1_0_0 directory.

We now explain how to use the Jasplot library from your own Java program. Consider, say, MyClass.java, which uses the Jasplot library, jasplot.jar, stored at /xxx/jasplot.jar. We first import the jasplot library at the top of the program:

```
import jp.ac.ism.jasplot.*;
```

MyClass.java can then be compiled and executed with the following commands:

```
javac -classpath .:/xxx/jasplot.jar MyClass.java
```

```
java -classpath .:/xxx/jasplot.jar MyClass
```

References

- Apache Software Foundation (2007). *Batik SVG Toolkit*. <http://xmlgraphics.apache.org/batik/>.
- Apache Software Foundation (2007). *Jakarta POI – Java API To Access Microsoft Format Files*. <http://jakarta.apache.org/poi/index.html>.
- Gamma, E., Helm, R., Johnson, R. and Vlissides, J. (1995). *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley, Reading, MA.
- Hofmann, H. (2000). Exploring Categorical Data: Interactive Mosaic Plots. *Metrika*, 51:11–26.
- Hunt, J. (2003). *Guide to the Unified Process featuring UML, Java and Design Patterns*. Springer, Berlin.
- Inselberg, A. (1985). The Plane With Parallel Coordinates. *Visual Computer*, 1:69–97.
- Kobayashi, I., Fujiwara, T., Nakano, J. and Yamamoto, Y. (2002). A Procedural and Object-Oriented Statistical Language. *Computational Statistics*, 17:395–410.
- Krasner, G. and Pope, S. (1988). A Description of the Model–View–Controller User Interface Paradigm in the Smalltalk-80 System. *Journal of Object Oriented Programming*, 1:26–49.
- Martin, R. (2003). *Agile Software Development: Principles, Patterns, and Practices*. Prentice Hall, Upper Saddle River, NJ.
- Murrell, P. (2005). *R Graphics*. Chapman & Hall/CRC, Boca Raton, FL.
- Nakano, J., Fujiwara, T., Yamamoto, Y. and Kobayashi, I. (2000). A Statistical Package Based on Pnnts. In: Bethlehem, J.G., van der Heijden, P.G.M. (eds) *COMPSTAT2000 Proceedings in Computational Statistics*. Physica, Heidelberg, pp. 361–366.

- Nakano, J., Huh, M.Y., Yamamoto, Y., Fujiwara, T., Kobayashi, I. (2004) Adding Visualization Functions of DAVIS to Jasp: Mixing Two Java-Based Statistical Systems. *Computational Statistics*, 19:137–146.
- Symanzik, J. (2004). Interactive and Dynamic Graphics. In: Gentle, J.E., Härdle, W., Mori, Y. (eds) *Handbook of Computational Statistics: Concepts and Methods*. Springer, Berlin, pp. 293–336.
- Unwin, A., Theus, M. and Hofmann, H. (2006). *Graphics of Large Datasets: Visualizing a Million*. Springer, Berlin.
- Urbanek, S. and Theus, M. (2003). iPlots – High Interaction Graphics for R. In: Hornik, K., Leisch, F., Zeileis, A. (eds) *Proceedings of the 3rd International Workshop on Distributed Statistical Computing (DSC 2003)*. <http://www.ci.tuwien.ac.at/Conferences/DSC-2003/Proceedings/UrbanekTheus.pdf>.
- Virius, M. (2004). Object Oriented Computing. In: Gentle, J.E., Härdle, W., Mori, Y. (eds) *Handbook of Computational Statistics: Concepts and Methods*. Springer, Berlin, pp. 403–434.
- Wilkinson, L. (2004). The Grammar of Graphics. In: Gentle, J.E., Härdle, W., Mori, Y. (eds). *Handbook of Computational Statistics: Concepts and Methods*. Springer, Berlin, pp. 337–377.
- Yamamoto, Y., Nakano, J., Fujiwara, T. and Kobayashi, I. (2002). A Mixed User Interface for a Statistical System. *Computational Statistics*, 17:379–393.

Web-Based Statistical Graphics III.18 using XML Technologies

Yoshiro Yamamoto, Masaya Iizuka, Tomokazu Fujino

18.1	<i>Introduction</i>	758
	The Web, Statistics and Statistical Graphics	758
	XML and Statistical Graphics.....	759
18.2	<i>XML-Based Vector Graphics Formats</i>	759
	What is XML?	759
18.3	<i>SVG</i>	765
	Overview of SVG.....	765
	Basic Structure.....	765
	Implementation of Interactive Functionality via JavaScript	768
18.4	<i>X3D</i>	771
	Overview of X3D.....	771
	Basic Structure.....	774
	X3D Scatter Plot Function of R	777
18.5	<i>Applications</i>	777
	SVG Application as Teachware	777
	Application to Three-Dimensional Representations	778
	GIS Applications	779
	Authoring Tool for SVG Statistical Graphics in R	784

Introduction

Most statistical graphics on the Web are static, noninteractive and undynamic, even though other statistical analysis systems usually provide various interactive statistical graphics. Interactive and dynamic graphics, see Symanzik (2004), can be implemented using Internet technologies such as Java or Flash (Adobe, 2007). Scalable Vector Graphics (SVG) and Extensible 3D (X3D) offer alternative means of realizing an XML-based graphics format. One advantage of using XML is that data from a wide range of research topics are easy to deal with, because they are all presented in the XML format. Another advantage is that XML is a text-based graphics format, i.e., it is scriptable, meaning that it can be generated dynamically by a statistical analysis system or web application. Before introducing XML-based graphics, we introduce the relationship between the Web, XML, and statistical graphics.

18.1.1 The Web, Statistics and Statistical Graphics

Although the Internet is a powerful communication tool, at present, its core function is webpage access, which has brought about dramatic changes to the way that statistics are made available.

In addition to the publication of official statistical data and the results of company research, members of the general public have begun to make data available through their own websites. Huge databases that can be accessed by anyone via the Internet have made information available to the public. As a result, statistical databases (see Boyens et al., 2004) have been developed to store such data, and statistical analysis methods such as data mining (see Wilhelm, 2004) have been developed to help the public access such data. Moreover, new target areas in statistical analysis, such as network intrusion detection (see Marchette, 2004), have been developed.

The popularization of the Web has brought about significant changes to the field of statistical analysis and statistics education. Early in the history of the Internet individuals made textbooks and data available to the public, so sites such as StatLib, which gathered statistical information, were important. At present, various services and applications that use multimedia and multiplatform characteristics are available for statistical analysis and for statistics education. Client–server type systems, such as XploRe (MD*Tech, 2007) and Jasp (Project Jasp, 2005), are also available to the statistical analysis systems. Moreover, server-type commercial software, such as SPSS and S-PLUS, is also available.

Numerous data sets, tutorials and analysis tools have been made available for statistics education, for example by the UCLA Department of Statistics and the Web Interface for Statistics Education (WISE, WISE Project (2007)) program at Claremont University. MD*Base (MD*Tech, 2007) and DASL (DASL Project, 1996) are databases of case studies. The EMILeA Stat (e-stat) project (BMBF, 2007) and the @d project (@d Project, 2006) enable analysis to be performed on the Web using statistical engines. In addition, e-learning systems, such as New Statistics (University of Hagen, 2007), use multimedia teaching materials that include video and interactive applications.

These types of content require statistical graphics in order to visualize statistical data. Early in the history of web publication, static (noninteractive) and raster (nonvector) graphics formats such as JPEG or GIF were used. However, using the Java mechanism, it became possible to implement interactive and dynamic graphics on the Web. However, such graphics did not become sufficiently popular. With the spread of Flash, the interactive features available on the Web became more popular. As a result, the demand for interactive and dynamic graphics using web technology is rising. In Web-based systems, it is sometimes necessary to create graphics according to user requests. In such cases, it is impossible to create graphs and prepare information beforehand. Moreover, a feature to display detailed information according to user requests is also necessary. Therefore, Web-based statistical graphics packages require interactive features.

XML and Statistical Graphics

18.1.2

HTML can provide functions on webpages by cooperating with other technologies, such as CGI or JavaScript, by linking to other pages, and by arranging information well. However, when the information described in HTML is reused, it is difficult to automate these tasks, because the accompanying information consists solely of tags that control the display of information on the webpage. This causes databases on the Web to become enormous. Therefore, the concept of the *semantic web* (W3C, 2007) was devised in order to allow information to be used efficiently and effectively.

Semantic web uses the metadata that accompanies all web contents to interpret exchanges between information devices, without mediation by a human operator, by conveying the meaning (semantics) of the information to the computer. The basic technologies for realizing the semantic web are XML and its associated technologies, which are standardized by the World Wide Web Consortium (W3C, 2007).

Standards based on XML have been developed for various kinds of data in order to realize the semantic web. StatDataML and DandD have been developed for statistical data, and GML has been developed for geographical information data. To realize statistical graphics in the semantic web framework, XML graphics such as SVG and X3D are necessary.

XML-Based Vector Graphics Formats

18.2

What is XML?

18.2.1

As we noted above, standards based on XML have been developed for various forms of data in order to create the semantic web, including StatDataML (Meyer et al., 2004) and DandD (DandD Project, 2007) for statistical data and GML (Open Geospatial Consortium, 2004) for geographical information data. But what is XML?

Just like HTML, XML uses tags to specify the meaning of data. The grammar associated with XML tags can easily enable computational processing while simultaneously expressing various types of data flexibly, enabling the data type to be set freely.

One of the advantages of SVG and X3D being XML-based is that documents written in these languages are easily interconverted using XSL Transformations (XSLT). XSLT is a language to convert XML document into other format, and itself is described by XML. Suppose that a user wants to get a scatter plot by SVG format when the data is given by XML format such as StatDataML. An XSLT that is possible to convert can be applied to any data of StatDataML format. The other advantage includes that an interactive function can be realized by scripting source code of the programming language which supports Document Object Model (DOM), such as JavaScript, in the XML document. DOM defines Application Programming Interface (API) which specifies the manipulating methods for XML elements and attributes by programming language and the tree-structured object model referenced by the interface. Some examples will be shown in a later section.

XML Files as Text Files

While conventional raster graphics such as JPEG, GIF, PNG and BMP are binary files, XML graphics are text files. Thus, we can confirm or modify the content of an XML file by opening it a simple text editor. Therefore, it is easy to reuse the contents of an XML file. This also allows us to develop systems more flexibly, because graphics can be output simply by displaying a text file, no matter what kind of programming language is used. Moreover, if the graphics (e.g., a statistical graph, a map or a CAD) are closely related to the outside resource, we include the related information between the element of the graphics and the outside resource within the graphics themselves. For example, a circle element which represents the point on a scatter plot can include not only its coordinates on the SVG canvas but also its data value and data label. In the case of GIS, graphic elements of SVG which represent geographical objects can include its latitude and longitude. On the other hand, raster graphics use binary images, so it is difficult to link outside resources and images because we have no information about figure elements. Moreover, in order to generate raster graphics, we need a library corresponding to the particular programming language we are using, and for different programming languages, the graphics must be output by different grammars.

Vector Graphics

There are two methods of displaying graphics by a computer: raster graphics and vector graphics. Raster graphics, which are expressed as enumerations of points (pixels) and colors (of the pixels), do not reproduce all of the information contained in the original image. In raster graphics, outlines appear jagged (notches appear) when the image is examined closely, and information is lost when we move away from the image. Therefore, raster graphics are not suitable for zooming or transformation. On the other hand, vector graphics contain drawing information, such as position, size and shape. Therefore, image quality deterioration can be prevented by using this in-

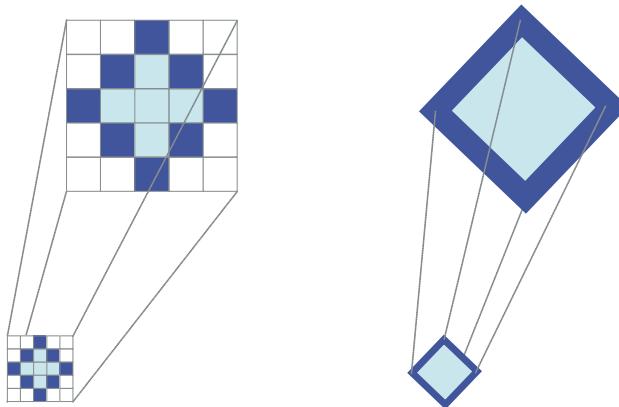


Figure 18.1. Vector and raster graphics

formation to redraw the image via software when the image is expanded, reduced, or transformed. In other words, the advantage of vector graphics over raster graphics is that the degree of freedom is high for vector graphics with high-quality images. In addition, vector graphics have the advantage that the file size relies on the quantity of information in the diagram, and the size of the image is unrelated to the file size.

Implementation of Interactive Functionality and Animation

The software or plug-ins that display most XML graphics have functions that allow actions similar to zooming and graphic movement. In addition, we can independently add a new interactive function to the XML graphics, as written in JavaScript. In addition, in SVG, we can change the attribute of an SVG element into SVG with a change in time using Synchronized Multimedia Integration Language (SMIL, W3C (2007)), which can be built onto SVG. This is referred to as animation. We will present examples of the implementation of interactive functionality using JavaScript later.

Interactive Capabilities of XML Statistical Graphics

The following interactive functions are particularly required for statistical graphs in exploratory data analysis (EDA), not only to visualize the data and the results from the analysis, but also to support the interpretation of the graph:

- zoom and pan (Fig. 18.2)
- tooltip, layer and highlight (Fig. 18.3)
- cooperation with other diagrams and tables (Fig. 18.4)

Zoom and pan helps us to observe local properties of the original graphics. It is especially useful for visualizing datasets with large amounts of information, such as scatter plots that include lots of data points and detailed map data. Figure 18.2 shows a plot of the number of live births per 10,000 23-year-old women in the United States between 1917 and 1975, and its smoothing curve, as realized by SVG. The right-hand side of Fig. 18.2 shows a magnified view of part of the plot. Zoom and pan can be

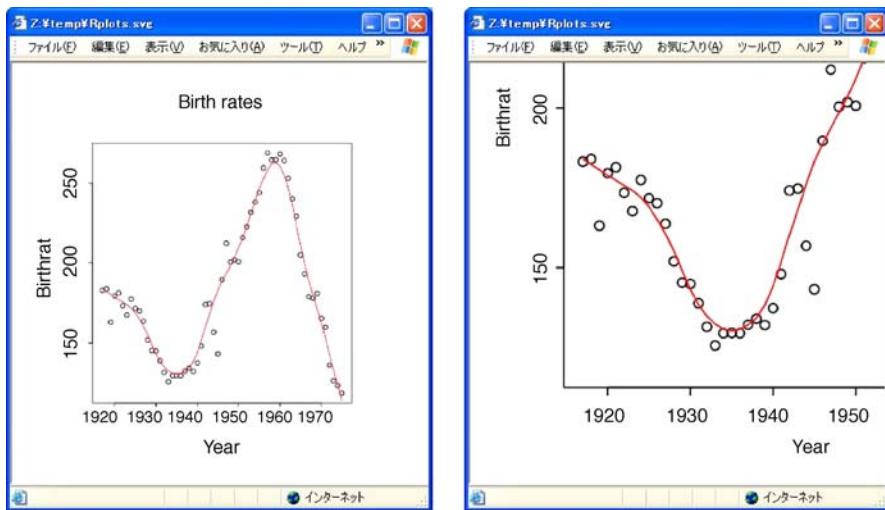


Figure 18.2. Zoom and pan

achieved without special programming in XML graphics because its rendering applications and plug-ins contain zoom and pan functions.

The tooltip function is used to display a variety of information types, such as data labels and values in relation to a particular location or point indicated by the mouse cursor. Detailed information is hidden in order to help users to intuitively understand the data through visualization. Tooltip is useful when users want to obtain information on demand.

The layer function overlays various kinds of information in a single display area. In this case, each type of information is called a “layer,” and users can select which layers to display. In the case of geostatistical data, there are often several types of information – such as maps, locations of data points, roads, railways, institutions, and results of statistical analyses – that need to be displayed. However, it is usually desirable to display only selected layers at any one time, because displaying all of them at once will hinder comprehensibility. Examples of the use of the tooltip and layer functions in SVG are shown in Fig. 18.3. These data, the *Cities* data of DASL (DASL Project, 1996), were collected by the Economic Research Department of the Union Bank of Switzerland. They represent the economic conditions in 48 cities around in world in 1991. The *Cities* data contain three variables: Work, Price and Salary. Work is the weighted average of the number of working hours for 12 occupations. Price is the index of the cost 112 goods and services, excluding rent (Zurich = 100). Salary is the index of hourly earnings for 12 occupations after deductions (Zurich = 100). Figure 18.3 shows a scatter plot of the first two principal component scores and cluster regions. Each state label can be obtained by the tooltip by pointing with the mouse. Furthermore, the elliptical regions show the two clusters obtained from the *k*-means method. Checkboxes on the html forms can be used to specify whether the cluster layer and the city labels layer should be visible or invisible.

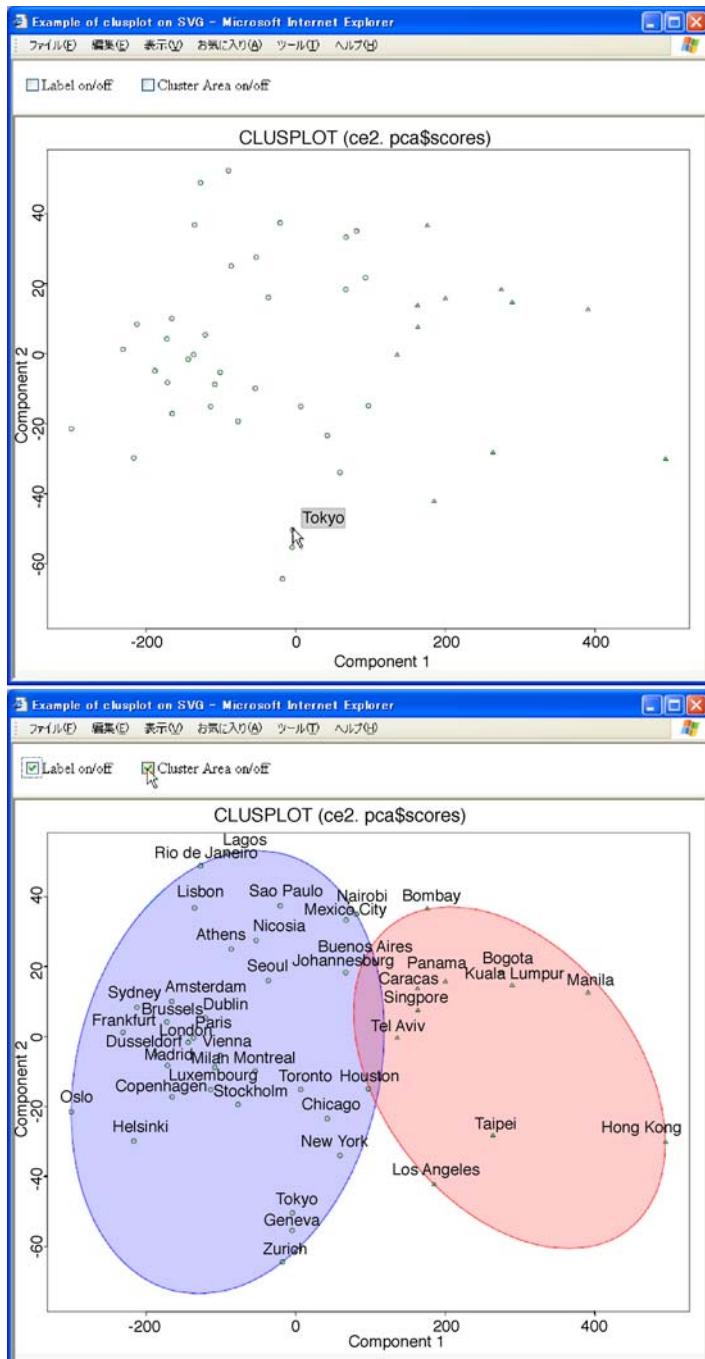


Figure 18.3. Examples of using a tooltip and switching layers

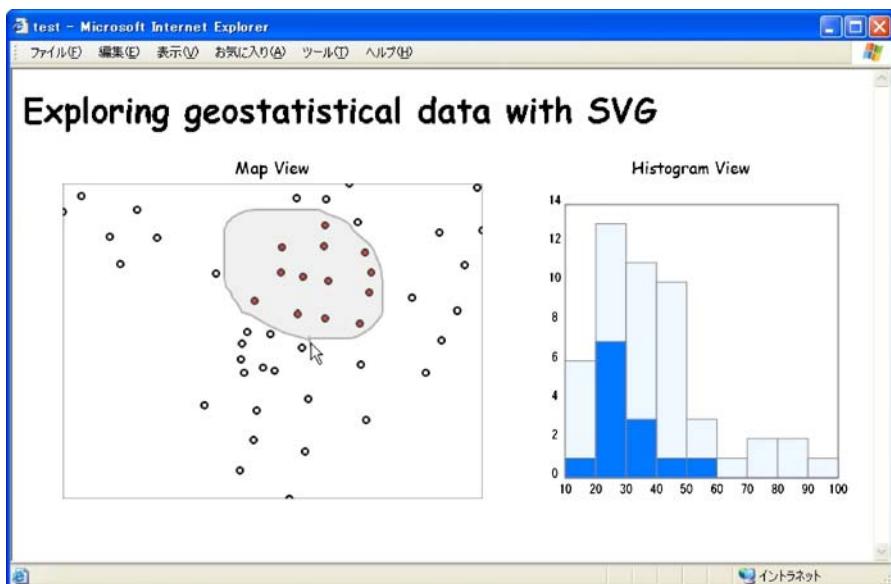


Figure 18.4. Cooperation between map view and histogram view

Figure 18.4 shows a tool for exploring spatial data using dynamic graphics that was proposed by Haselett et al. (1991) as an example of the application of cooperation between two statistical graphs. This tool was also realized via SVG. The locations of observation points are plotted in the “Map View” on the left, and a histogram of the observed data is displayed in the “Histogram View” on the right. An arbitrary region can be specified by dragging the cursor over the “Map View.” The data in the histogram that corresponds to the locations included in the specified region will then be overwritten on the “Histogram View.” This tool makes it possible to explore the local variability in spatial data intuitively.

Although it is possible to realize these functions by raster graphics in a web application, we encounter some problems if we do this using the conventional method. It is difficult to update part of the image dynamically with raster graphics because the color data are associated with the individual pixels. Therefore, it is necessary to regenerate the entire image from the server upon a user request, and then display the image on the client side. That is, the server and the client must communicate upon each individual user request, which leads to a decline in the operability and performance of the web application. Because the amount of information in a raster graphic increases with the quality and the image size, this decline is significant. In addition, there are portability problems in that functions such as (1) and (2) work only within the web application; they cannot be executed in other environments. XML graphics can solve such problems and they offer many advantages to the user and the developer. We will discuss this in more detail in the following section.

SVG

18.3

Overview of SVG

18.3.1

SVG is an XML format for describing two-dimensional vector graphics. In the days before SVG, the Microsoft-led Vector Markup Language (VML) and the Adobe-led Precision Graphics Markup Language (PGML) were proposed to W3C (W3C, 2007). SVG1.0 was released in September 2001 by W3C as an integrated format. The current version, SVG1.1, was recommended in January 2003.

SVG Viewers

SVG consists of a plain text file as well as HTML. Thus, exclusive software that displays SVG as a graphic is required. Functions mentioned in the previous section, such as zooming, are implemented in the viewer. Table 18.1 lists some SVG viewers; they be classified into the following three categories:

- **Browser plug-in:** The browser plug-in is the most common form of SVG display environment. Adobe SVG Viewer 3.0 (ASV3, Adobe (2007)), which supports SVG 1.1, is the de facto standard for SVG browser plug-ins. ASV6, which supports the next version of SVG (SVG1.2), is currently at the beta testing stage and is set to be released after the SVG1.2 recommendation is announced.
- **Web browsers that support native rendering of SVG:** SVG-enabled builds of Firefox and Mozilla (the official binary packages do not support native rendering) and Opera 8.0 support native rendering of SVG.
- **Stand-alone application:** Batik (Apache Software Foundation, 2007) is a Java technology-based toolkit for SVG. One application of Batik is Batik Squiggle, which is a full-fledged SVG browser.

We will briefly illustrate the SVG language specification through simple examples. For more detail, refer to the W3C website W3C (2007). Note that all SVG examples were obtained with Internet Explorer 6.0 + ASV3.0.

Basic Structure

18.3.2

The first line of a SVG document is the XML declaration, and the second line is the Document Type Definition (DTD) declaration. The root element of SVG is the

Table 18.1. SVG Viewers

Type	Products	Version	SVG	Developer
Plug-in for IE	SVG Viewer	3.03	1.1	Adobe Systems, Inc.
	SVG Viewer	6.0 beta	1.2	Adobe Systems, Inc.
Web browser	Firefox	1.5	1.1	Mozilla Corporation
	Opera	8.5	1.1	Opera Software
JAVA application	Batik Squiggle	1.6	1.1	Apache Software Foundation

<svg> element, which has width and height attributes that set the canvas size of SVG. When the attribute unit is not specified, the units are assumed to be pixels. The <title> and <desc> elements provide a title and detailed information about the SVG document, respectively. The text node content of the <title> element is shown in the title bar of the window.

Coordinate System

SVG operates on an infinite plane on which the coordinate system is oriented such that the positive x -axis is to the right and the positive y -axis is downward from the coordinate origin. However, in many statistical graphs, the positive y -axis is directed upward. Therefore, when coding a statistical graph in SVG, developers must take this quirk into account in order to make sure that the graphs are displayed correctly. An SVG viewport is a physical area that displays graphics elements on the screen; its size is set via the width and height attributes in the <svg> element. When no units are specified, the units of the width and height attributes are assumed to be pixels. Although other units, including cm and pt, can also be designated, no units are specified in the cases presented herein. When only the width and height attributes are designated, a domain of size width \times height extending from the origin of the coordinate plane is assigned to a viewport of size width \times height. When a specific domain is required in the coordinate plane (coordinates of the upper left quadrant of the domain: (originX, originY), size: width \times height) in this viewport, the viewBox attribute is designated as follows:

```
viewBox="originX originY width height"
```

When the aspect ratios of the domain and the viewport are different, the default action is to shrink or enlarge the x - and y -axes appropriately, but this behavior can also be controlled by designating the preserveAspectRatio attribute.

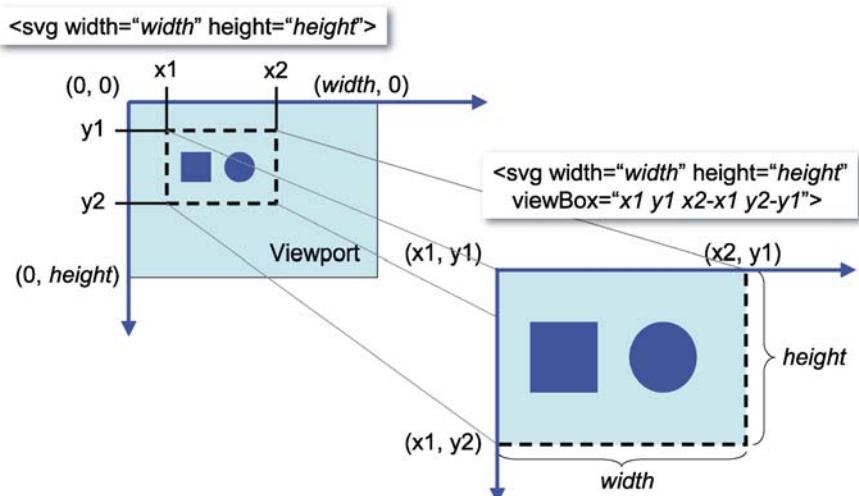


Figure 18.5. Viewport and viewBox

Basic Shapes

Basic shapes available using SVG are illustrated in Fig. 18.6, along with the corresponding SVG codes. The minimum information required to display the shapes, such as position and size, are specified as attributes of each element. Additional information, such as the line and fill colors and the line width, are specified via the `style` attribute in Cascading Style Sheet (CSS) format.

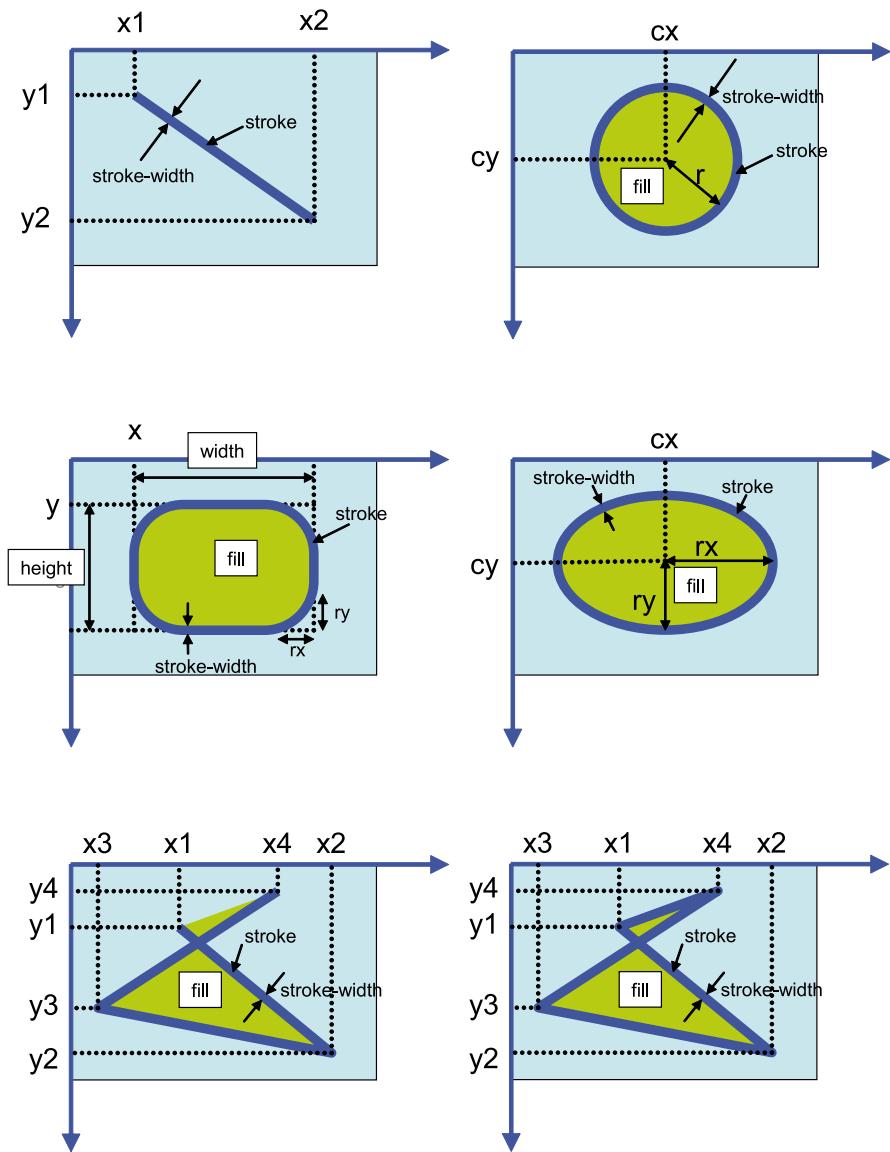


Figure 18.6. Basic shapes

Text

Text can be realized in SVG using the `<text>` element, which has the attributes `x` and `y` that determine the position at which the string in the text node enclosed by the `<text>` element is displayed. Figure 18.7 shows a simple example of text in SVG.

Various properties of text, such as font size, family, weight, and color, are specified using the `style` attribute. The shape of the text in SVG is controlled by the font border and the area inside the border, which can be specified in various ways, as shown by the third string in the example. The `text-anchor` property is used to determine how the string is positioned with respect to the text position specified by the `x` and `y` attributes (in other words, whether the text position corresponds to the start, the middle, or the end of the string).

Group

One of the advantages of vector graphics is that layers can be included. In SVG, layer functionality is realized using the group element `<g>`. This means that elements that are grouped by the `g` element are considered to be one layer. By using `<g>` elements appropriately, it is possible to turn a specific layer on and off, and apply styles and interactive functions (as described in the next section) to all of the elements in the layer collectively. Figure 18.8 shows a simple scatter plot obtained using the `<g>` element.

The data points (circle elements) are collected into one layer using one `<g>` element that has a `style` attribute which is reflected in all of the data points. The size of an SVG file that includes several elements, such as a scatter plot with several data points, is significantly increased when the `style` attribute is specified for each `<circle>` element. Therefore, it is desirable to minimize the size of the SVG file by using the `<g>` element.

Implementation of Interactive Functionality via JavaScript

18.3.3

As stated earlier in this section, interactive functions other than those supported by default in SVG browsers (such as zooming), can be implemented using JavaScript. VBScript is also available in some SVG browsers. All of these languages are Document Object Model (DOM)-compliant. In other words, all of these languages define



Figure 18.7. Text elements

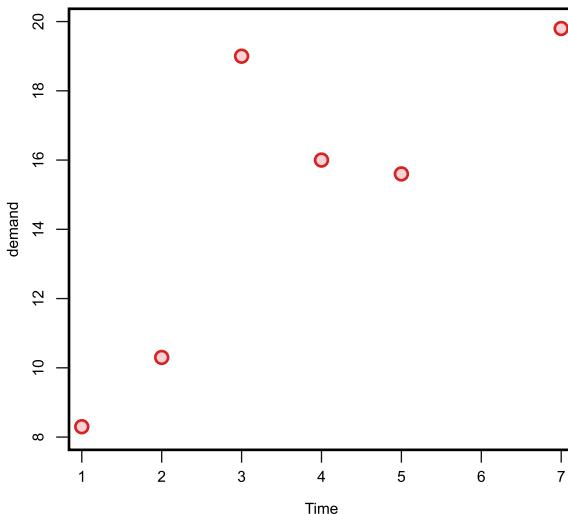


Figure 18.8. Grouped elements

an API and a corresponding object model for manipulating XML documents. Each SVG element can include event handlers that implement the appropriate script when events are captured (such as keyboard and mouse operations and SVG document loading). In this example, the `onclick` attribute has an `alert` method (which displays messages in a pop-up window) that is set as the event handler in the group element that has text and circle elements. The most frequently used event handlers in SVG are those associated with mouse events, such as `onmouse {down, move, out, over, up}`, SVG-specific events such as `onload`, `onresize` and `onzoom`, and keyboard events such as `onkeydown`, `onkeypress` and `onkeyup`.

Tooltip

We will now briefly illustrate how to realize tooltip and layer functionality by JavaScript in SVG. Three functions, `ShowTooltip`, `HideTooltip`, and `ZoomControl`, are defined in the script of Fig. 18.9. The `ShowTooltip` function is called from the `onmousemove` attribute in the `<g>` element, which includes `<circle>` elements for data points. When the mouse cursor is moved over one of the data points, the `ShowTooltip` function is executed. In this function, the `text` and `rectangle` element grouped together as a tooltip (hidden in the initial state) are updated by JavaScript. At first, the id of the `<circle>` element indicated by the mouse cursor is acquired, and the tooltip text is updated. The position of the mouse cursor is then acquired, and the positions of the text and the rectangle are updated. Finally, the `visibility` property of the `style` attribute of the tooltip is set to `visible`.

The `HideTooltip` function is called from the `onmouseout` attribute of the `<g>` element. The process that is executed when the mouse cursor moves out of the region associated with the element is described in the `onmouseout` attribute. This

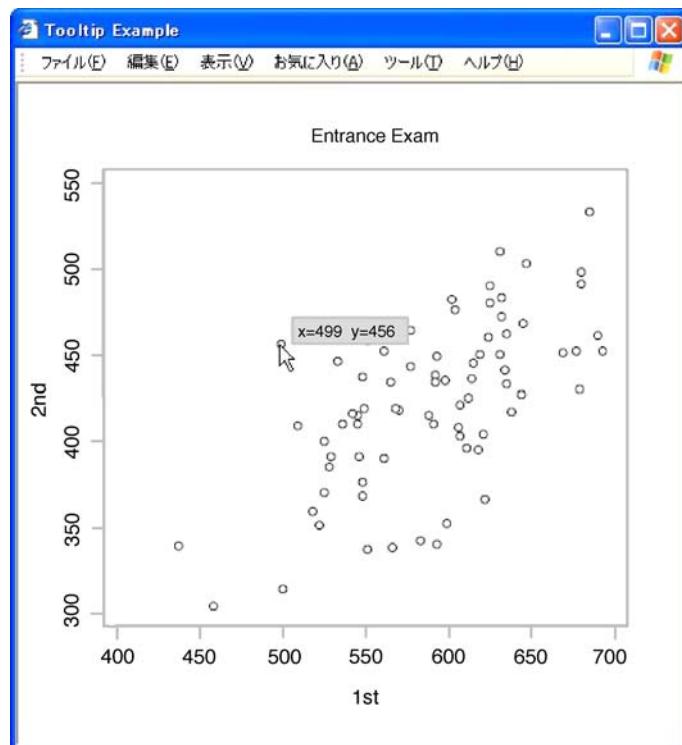


Figure 18.9. Tooltip example

function causes the tooltip to disappear (the `visibility` property of the `style` attribute of the tooltip is set to `hidden`).

The `ZoomControl` function is called from the `onzoom` attribute of the `<svg>` element; this function executes when zooming is required in the SVG viewer. The script-revising `scale` property (which depends on the degree of zoom) of the `transform` attribute of the tooltip is specified, so that the tooltip is not magnified excessively when zooming is requested.

Switching Layer

Displaying related information on statistical graphs and maps is an effective means to perform visual analysis. However, this approach may actually be disadvantageous at times, since there may be too much information displayed at any one time. Therefore, functions that control whether information is displayed or not are necessary in visual analysis. Such a function can easily be realized in SVG using JavaScript. Objects that may or may not be displayed are first gathered into one `<g>` element, and the `visibility` property is set in the `style` attribute. Code to set whether the item is visible or hidden is provided in a script called from an event.

In the example shown in Fig. 18.3, the SVG component (`cluster.svg`) that displays the graph and the HTML form (`control.html`) that provides an interface for changing the visibility of items in the graph are separated by a frame (`index.html`). The `control.html` form contains check boxes that specify whether the data point labels and/or ellipse domains are visible; the function that is executed when the check boxes are selected is designated in the `onclick` attribute.

In `cluster.svg`, the element for the ellipse domain is described as a child element of the `<g>` element in which the `id` attribute is “`clustArea`,” and the `<text>` element for the individual label is described as a child element of the `<g>` element in which the `id` attribute is “`label`.” When `cluster.svg` is loaded, the `setVisibility` function defined in `cluster.svg` is linked to `index.html`. As a result, this function can be called by `control.html`. The `setVisibility` function receives the contents of the `id` attribute of the check box clicked in `control.html` as an argument, and the `visibility` property of the `<g>` element corresponding to the `id` is changed.

Figure 18.10 shows another example of layer functionality. In this example, the number of clusters displayed can be changed interactively. This plot has several layers containing cluster polygons for cluster numbers of 2–6. If the Overlap check box is checked, the cluster polygons are displayed such that they overlap the previous view. Thus the plot can show the hierarchical structure of clustering results obtained using a hierarchical clustering method.

X3D

18.4

Extensible 3D (X3D) is an open-standard XML-enabled three-dimensional modeling language that enables the real-time communication of 3D data across all applications, including network applications. X3D is neither a programming API nor a simple file format for geometrical data interchange. Instead, X3D combines both geometry and runtime behavioral descriptions into a single file. X3D is the next revision of the VRML97 ISO specification, referred to as VRML-NG (Next Generation). In this section we will introduce some features of X3D; a full description of X3D can be found at the Web3D website (Web3D Consortium, 2007) and in Geroimenko and Chen (2004).

Overview of X3D

18.4.1

Specifications

The first draft of the VRML 1.0 specification was published in 1994. In 1996, the first version of the VRML 2.0 specification was released, and the JTC1/SC24 committee of the International Standards Organization (ISO) agreed to publish VRML 2.0 as Committee Draft (CD) 14772. This specification is known as VRML97. X3D, a new version of VRML, has been designated International Standard ISO/IEC 19775, and was published in 2004 by the Web3D Consortium. As of September of 2005, there are six X3D International Specification Standards, including X3D encodings, which

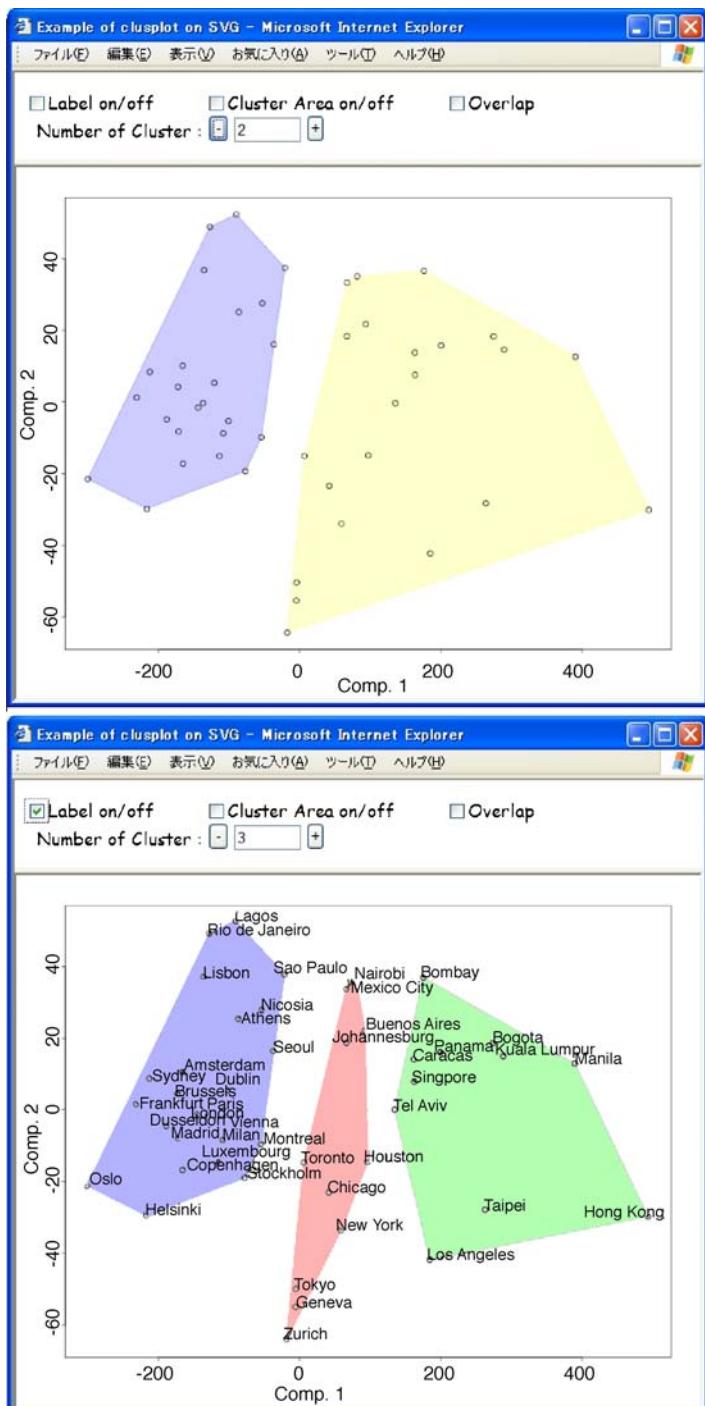


Figure 18.10. Scatter plot with cluster polygons

specify XML and Classic VRML encoding. The latest specifications are described at the Web3D website.

The Web3D Consortium organizes several working groups to deal with various problems regarding Web3D. Several working groups, such as GeoSpatial (X3D Geo-Spatial Working Group), H-Anim (Humanoid Animation), and a number of focus market working groups, such as CAD, are researching and proposing solutions to specific technical problems related to X3D.

Component and Profile

Component and profile are new X3D methods of defining both extensibility and the set of services required by user content. A component defines a specific collection of nodes, and a profile is a collection of components at specific levels of support. X3D allows developers to support subsets of the specification (profiles) composed of modular blocks of functionality (components).

A component-based architecture supports the creation of different profiles that can be individually supported. Components can be individually extended or modified by adding new levels, or new components can be added to introduce new features, such as streaming. Through this mechanism, the specification can be rapidly advanced because development in one area does not slow down the specification as a whole.

The following are X3D baseline profiles:

- The **Interchange** profile is the basic profile for communication between applications. **Interchange** supports geometry, texturing, basic lighting, and animation.
- The **Interactive** profile enables basic interaction with a 3D environment by adding various sensor nodes for user navigation and interaction (e.g., **Planse-Sensor**, **TouchSensor**, etc.), enhanced timing, and additional lighting (**Spotlight**, **PointLight**).
- The **Immersive** profile enables full 3D graphics and interaction, including audio support, collision, fog, and scripting.
- The **Full** profile includes all defined nodes, including NURBS, H-Anim and GeoSpatial components.

X3D viewers

X3D requires a viewer, a X3D browser or a plug-in for a Web browser, in order to parse and realize a 3D world. It is possible to move and rotate this world using the functions of the viewer. Details of the viewer are provided at the X3D Documentation website (Web3D Consortium, 2007).

Octaga Player is the first 3D player for both VRML and X3D. Octaga Player supports the entire profile of X3D and is freely available for personal noncommercial use Octaga AS (2007). Octaga Player is a high-performance, standards-compliant 3D player that can run as a standalone application or as a plug-in in any Internet browser. In this section, all X3D objects are shown using Octaga Player.

Table 18.2. X3D Viewers

Product	Type	OS	License
Octaga Player	Viewer	Linux, Windows	Commercial
Octaga Professional	Viewer	Windows	Commercial
Flux Player	Plug-in	Windows	Commercial
FreeWRL	Viewer, Plug-in	Java, Linux, MacOS X	GPL-style
Xj3D	Viewer	Linux, MacOS X, Windows	GPL-style

FreeWRL is an open-source X3D/VRML browser and plug-in. Platforms that support FreeWRL include MacOS X, Linux, Unix, IRIX, and Java. Other plug-ins for MS Windows are Flux Player, OpenWorlds and Vcom3D Venues. Xj3D is a project of the Web3D Consortium (Web3D Consortium, 2007) that focuses on the creation of a toolkit for VRML97 and X3D content written completely in Java. A standalone viewer is included.

18.4.2

Basic Structure

The first line is the XML declaration, and the second and third lines are the DTD declaration. The root element of X3D is the `<X3D>` tag (called a node in X3D), with a `version` attribute that specifies the version of X3D and a `profile` attribute that specifies the profile. The X3D world is described with the `<Scene>` node, by arranging its contents appropriately. The default background color is black.

Node and Field

Since X3D uses an object-description format, the world or components consist of 3D, multimedia and interactive objects. An object is described by a nest of nodes, and the parameters of an object are described as fields. New components can reuse the grouping and prototype of an existing node.

Standard Units and Coordinate System

X3D defines the unit of measurement of the global coordinate system as the meter. All other coordinate systems are generated from transformations based on the global coordinate system. The unit of linear distance is the meter. Angles are given in radians, and time is given in seconds. The color space is specified by three real numbers (RGB) between 0 and 1, e.g., '1 0 0' for red.

X3D uses a Cartesian, right-handed and three-dimensional coordinate system. By default, the viewer is positioned along the positive z -axis, looking in the z -direction with the $+y$ -axis directed upwards. A modeling transformation (`<Transform>`) and `<Billboard>` or viewing transformation (`<Viewpoint>`) can be used to alter this default projection.

Basic Objects

Objects that construct a X3D world are described within the `<Shape>` node. The basic nodes for figures are `<Box>`, `<Sphere>`, `<Cylinder>` and `<Cone>`, and parameters (such as size and radius) are specified using fields. The material of an object is specified within the `<Appearance>` node using the `<Material>` node or the texture node. The color and the degree of transparency of a object is also specified within the `<Material>` node using the `diffuseColor` and the `transparency` fields. The position of an object is specified by nesting with the `<transform>` node and setting the `translation` field. When using the `<Text>` node, it is possible to show strings with the `string` field and set the font type and size with the `fontStyle` field. When using the `<Billboard>` node, it is possible to orient the text face to the front view. Figure 18.11, left, shows the coordinate axes generated by using `<Cylinder>` as the axis, `<Cone>` as the arrows for the coordinate axes, and `<Text>` as the axis labels. By arranging points in the same way via X3D objects such like `Sphere` or `Box`, a scatter plot can be composed. The right-hand side of Fig. 18.11 shows a 3D scatter plot of the *Cities* data. This viewpoint gives almost the same 2D scatter plot as Fig. 18.3.

Using the Octaga player, the mode can be set by choosing from the fourth to ninth buttons from the left of the tool bar. The modes are Walk, Fly, Examine, Slide, Pan and Look-at from the left. Various views can be realized using these modes. It is possible to specify a mode that describes the `type` attribute of `<Navigation-Info>` node. If a particular mode is specified, then all of the other mode buttons that are not specified become disabled.

Moreover, it is possible to create a two-dimensional plane with `IndexFaceSet` and `IndexLineSet`. Thus, a statistical map can be constructed using these nodes. Figure 18.12 displays a 3D bar chart of ward populations in the city of Sapporo (in

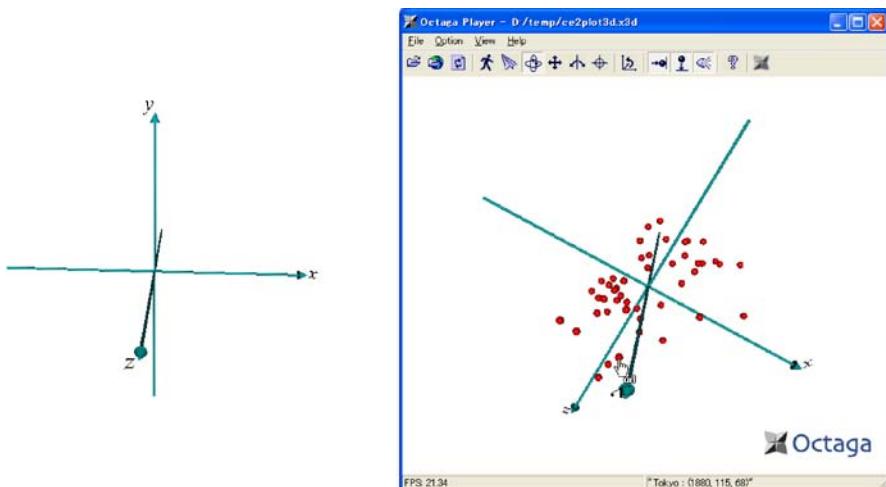


Figure 18.11. Coordinate axes (left) and 3D scatter plot (right)

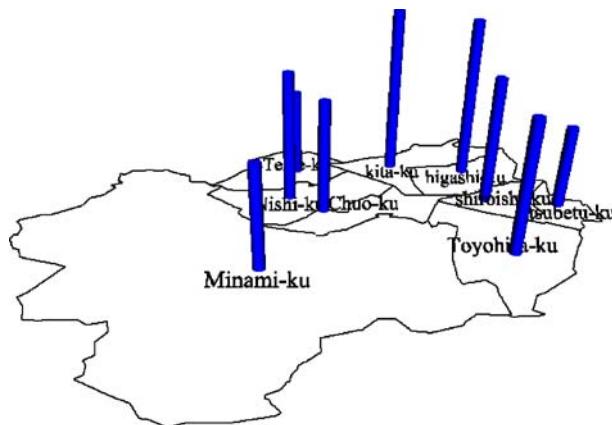


Figure 18.12. Bar chart placed on a city map

northern Japan) plotted on the city map construct by X3D. The city map is made as an SVG file using `IndexLineSet`. The statistical map is created by utilizing the SVG file of the city map and by placing `<Cylinder>` nodes with suitable lengths on the map. This approach is convenient when creating a statistical map on top of a physical map.

Grouping and Prototype

The `Group` node can be used to group objects in order to move or copy all of the components. The `DEF` keyword can be used to define an object that can be used by the `USE` keyword. In the above example involving coordinate axes, a cylinder and a cone can be grouped together in order to create an arrow. The arrow is used to indicate another axis with the `USE` keyword, and it is rotated to point in the appropriate direction. The labels of axes are composed of different strings, so we define the prototype of the label with the `<ProtoDeclare>` node. It is easy to create individual labels that can be used to make instances of the prototype using the `<ProtoInstance>` node.

Interactive Functionality

X3D has various sensors. The touch sensor, for example, generates an event when the mouse is clicked or the item of interest is approached. This event requires animation, during which the colors and positions of objects change. If the example of Fig. 18.11, all of the points have an `<Anchor>` node with a `description` field that describes the case name and point coordinates. By positioning the mouse cursor over a point, the case name and the coordinates of the point are shown at the status bar on the bottom right-hand corner of the viewer. Moreover, it is possible to implement more interactive functions in order to use Java and JavaScript (ECMAScript) within the `<Script>` node.

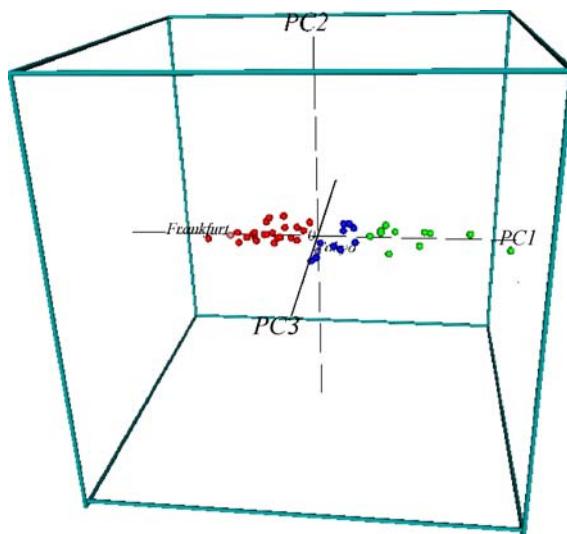


Figure 18.13. Colored scatterplot with frame and coordinate axes

X3D Scatter Plot Function of R

18.4.3

Using the 3D coordinate system introduced above as a template, it is easy to create a 3D scatter plot by simply transforming the coordinates and plotting points using a text editor. However, this is not easy when there are numerous data points to visualize.

Therefore, we provide a function that enables the statistical package R (R Project, 2007) to make a 3D scatter plot. The options available for the 3D scatter plot are rescaling and presentation of the coordinate axes, coordinate planes, frames, and case labels. An interactive function to show the case name and the coordinates on the bottom right-hand status bar when a point is selected is also implemented. The function `x3dplot3d()` can create a grouped scatter plot with colors (see Fig. 18.13).

Applications

18.5

SVG Application as Teachware

18.5.1

As we mentioned in Sect. 18.1, there is a great deal of software on the Web that can be used to teach statistical thinking or concepts. Since most students are familiar with TV or games, it is most productive to teach statistics in an interactive and visual way. This is the aim of statistical teachware such as MM*Stat (MD*Tech, 2007) and interactive textbooks like e-books (MD*Tech, 2005), including those provided by XploRe (MD*Tech, 2007).

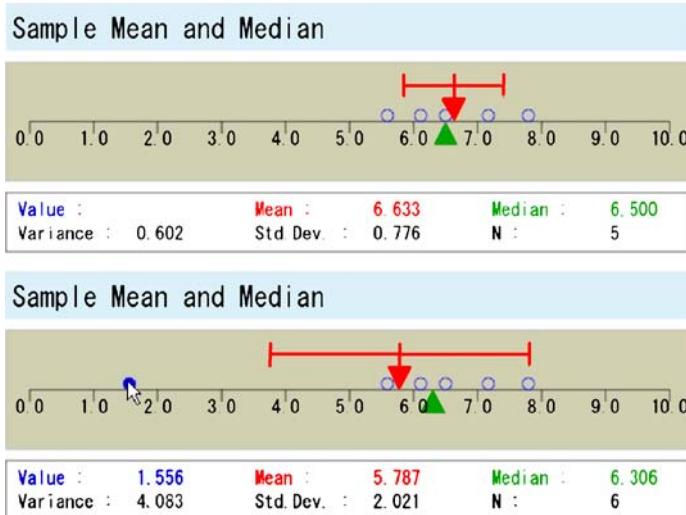


Figure 18.14. Teachware that illustrates the concepts of the mean and the median

Such interactive applications are developed using Java or Flash, but it is possible to develop similar applications by utilizing the interactive characteristics of SVG. One advantage of SVG teachware is that it is possible to use the interactive teachware without a network connection or statistical engine.

The application introduced here is an SVG version of teachware developed in Java for the Case project (Project CASE, 2007). This application was written to teach the difference between the concepts of the mean and the median. The application presents a coordinate axis. When a point on the axis is clicked, this data point is added to the axis, and the mean and median of all data points are calculated and presented. This application therefore shows the difference between the mean and the median visually.

In Fig. 18.14, the plot at the top shows five data points and displays the mean and the median as triangles. The plot below shows that adding a data point that is much less than the others changes the mean a lot, but changes the median only slightly. It is easy to edit this SVG application, since the SVG is in a text file. If someone wanted to change the language presented by the applications, they would only need to change the corresponding parts; a special authoring tool would not be needed.

18.5.2 Application to Three-Dimensional Representations

By rearranging a X3D scatter plot, it is possible to generate a new graphical representation. Figure 18.13 shows the results of a principal components analysis (PCA), including the first three principal components, of the *Cities* data, colored according to k -means clustering for six clusters. Figure 18.15 shows a prototype of a 3D dendrogram.

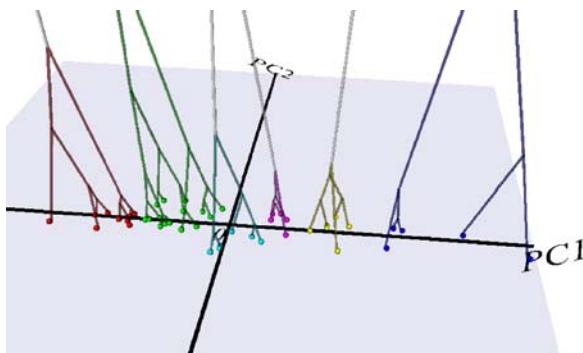


Figure 18.15. 3D dendrogram for the first two principal components

The first two principal component scores are plotted and the dendrogram is constructed according to the hierarchical clustering method. X3D graphics make it easy to realize 3D graphical representations via statistical software like R or XploRe.

Another application of X3D graphics is to visualize association rules for a market basket analysis. For more detail on association rules, see Wilhelm (2004). In general, an association rule is described in the form $X \Rightarrow Y$, in which X and Y are referred to as the rule head and rule body, respectively. The method of obtaining the association rules depends on the specification parameters, support and confidence. In addition, it is difficult to visualize association rules, because there are too many parameters. Figure 18.16 shows a prototype of a representation of association rules. Items are arranged according to item score by Hayashi's quantification method, type III. The height and color of each bar indicate the confidence and the support, respectively. The translucent plane represents a confidence level of 0.55. Meaningful rules are therefore represented by the tall red bars. The status bar at the bottom on the right shows information about the rule selected by the mouse cursor (pointing finger); the rule is $21 \rightarrow 8$; the support is 0.522; and the confidence is 0.756757.

GIS Applications

18.5.3

One of the most important fields that can benefit from the use of SVG is that of GIS (geographic information system) web. GIS applications do not require simply a function that displays a map. Functions that enable the enlargement/reduction/movement of the map and those that display information, such as on roads, railroads or facilities, are required depending on the demands of the user. Furthermore, statistical information on domains and points retrieved from databases associated with statistical surveys and location-dependent statistical data is presented on the map. Although many such systems have been designed, most have been built as web applications using raster graphics or as Java applications, which is a somewhat problematic approach. Applications that use raster graphics encounter issues related to the cost of preparing large numbers of image files (all of the maps required for the system and maps that

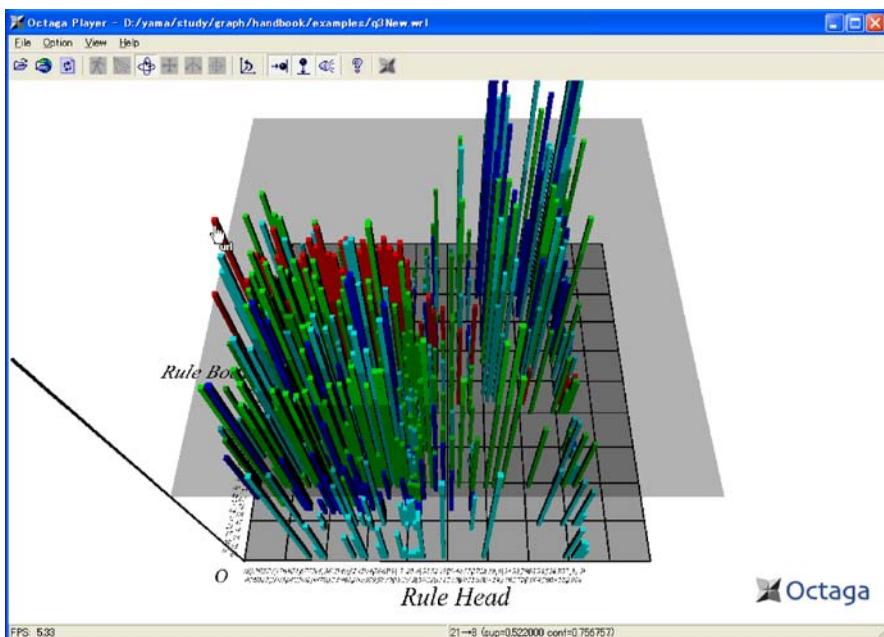


Figure 18.16. [This figure also appears in the color insert.] A representation of association rules

have layered information) and operability, because extensive communication with a server is necessary upon every user request. Moreover, scalability problems (extra functionality cannot be added to the application) and picture quality issues (as described in the “Introduction”) are also troublesome. In the case of Java applications, developmental costs increase because many components (including the interface and drawing components) are required in order to develop a useful and convenient system. The adoption of SVG is expected to enhance productivity and convenience when applied to GIS applications on the Web. The advantage of using SVG in GIS applications is easy operability, because SVG enables only part of a graphic to be changed, and it allows graphics layering without necessitating the entire graphic to be reread upon a user request.

An example of a SVG-based GIS web application is described below.

Okayama Trade Area Analysis System

This system was developed in order to visualize data for a trade area obtained in ten behavioral area surveys performed in Okayama prefecture, Japan, from 1979. In each survey, 7,000–8,000 replies were collected, and a number of samples were obtained for each of 78 municipalities in proportion to the population of the city, town or village in the area in order to examine the change in the population of each local municipality. The survey consisted of questionnaire items regarding the city, town or village in which the respondent most frequently purchased or used 15 items or services.

Degree of Dependence and Outflow Ratio

The degree of dependence and the outflow ratio, which are important characteristics in the analysis of trade areas, can be calculated using these data. We let n_{ab} be the number of inhabitants of city A that purchase a particular item in city B. $n_{ab} / \sum_b n_{ab}$ is the proportion of people who purchase the item in city B out of the total number of people in city A who purchased the item. When city A is fixed and this proportion is calculated for each city, town and village B in the prefecture, the results are collectively referred to as the outflow ratio from city A to each of the municipalities. In contrast, when one city B is fixed, and the proportions are calculated for each city, town and village A in the prefecture, the results are collectively referred to as the degree of dependence on city B of each of the municipalities. The behavioral tendency of inhabitants of a particular municipality can be determined by examining the outflow rate of the municipality, and when the degree of dependence is examined, the trade area of the municipality can be determined. This system can visualize the outflow rate with a spider plot (Fig. 18.18) and the degree of dependence with a choropleth map (Fig. 18.17). The annual change in each characteristic can also be visualized (Fig. 18.19). Furthermore, because the system has a function that displays/hides information (Fig. 18.20) such as the locations of government offices, railways, stations, roads and borders of cities, towns and villages on the map, the relationship between the change in the trade area and these factors can be determined. When a position of public office is selected for the cities, towns, and villages, corresponding time series plots for “Degree of dependence” and “Outflow ratio” are displayed (Fig. 18.21).

Ajax Technology

As well as using SVG as a tool for visualization, another feature of this system is that it is implemented in the form of a web application written in Asynchronous JavaScript + XML (Ajax). Ajax is implemented in the form of a web application that executes processing while transmitting and receiving XML or plain text data and without updating an entire web page using the HTTP communication function of JavaScript implemented in a web browser. Because HTML is output whenever a user request is processed in a CGI-based web application, sufficient operability is still not achieved. However, a seamless Web application that does not make a user conscious of the server can be realized because only the minimum information required to update the web page is obtained from the server using Ajax, and the HTML and SVG descriptions can be updated through the DOM API. Because SVG consists of XML and a text file, SVG can be treated in the same way as HTML in websites that are constructed using Ajax technology.

Naturally, the user interface is a web browser, and the HTML file, which is a frame consisting of three files (menu.html, layercontrol.html, okayama.svg), is loaded first from a web server. The back end of the server side is a RDBMS. A CGI script that receives requests from clients, sends queries to the RDBMS, receives the results, and sends the required data to the client is installed in the web server. A map and related information regarding Okayama prefecture are contained in okayama.svg. The map is described by polyline elements, which are grouped together as `g` elements with



Figure 18.17. Choropleth map

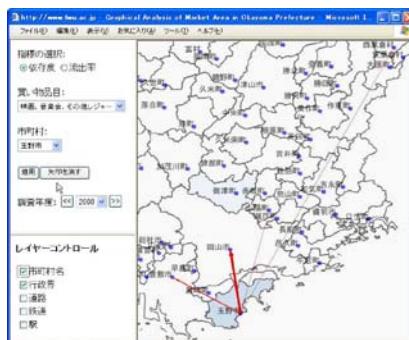


Figure 18.18. Spider diagram

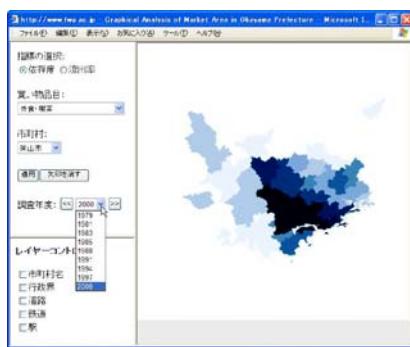


Figure 18.19. Time variation

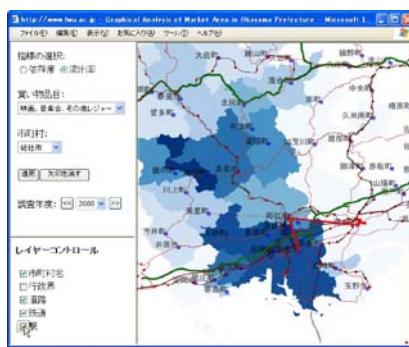


Figure 18.20. Displaying other layers

an id property for each municipality. The following five functions are defined in the script and they can be called from menu.html and layercontrol.html by associating them with index.html upon loading.

- **setBorder(col)**: Boundaries of cities, towns and villages are set by colors designated with col. Display or non-display of the boundaries is realized by changing the color of the boundary to black or white, respectively.
- **fillColor(id,col)**: Fills the domains of the cities, towns and villages specified by id with the color specified by col. Called when a choropleth map is drawn.
- **setVisibility(id,visibility)**: Sets the visibility property of the layer for information associated with the specified id that can be made visible or hidden.
- **setArrow(id1,id2,sw)**: Draws an arrow with a stroke width sw from government office id1 to government office id2. Called when drawing a spider diagram.
- **eraseArrows()**: Erases every arrow.

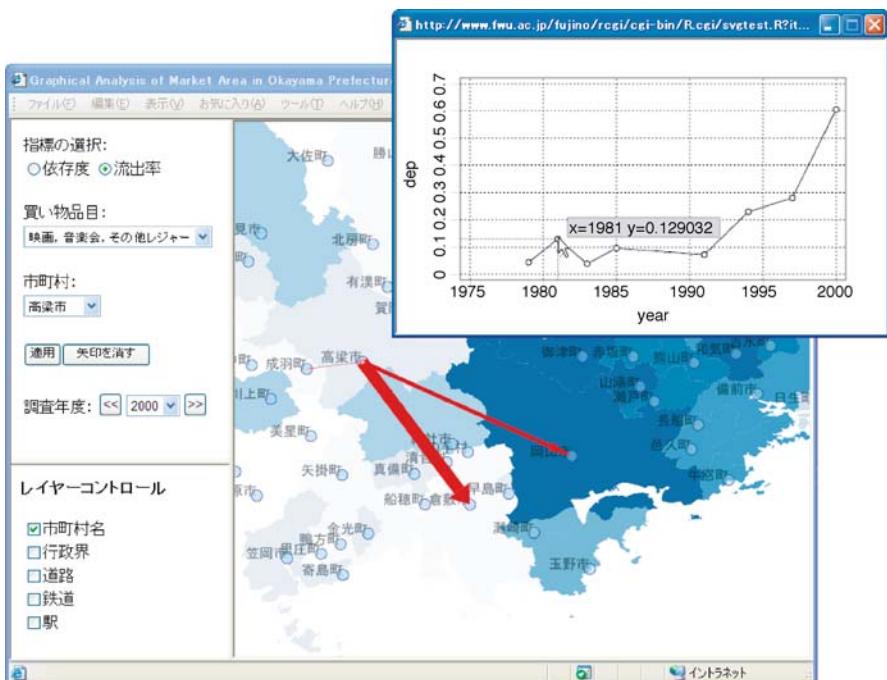


Figure 18.21. Time series plot of the Okayama trade area analysis system

`layercontrol.html` consists of check boxes that specify whether to display a layer of related information, and the `setVisibility` function is called when an `onclick` event occurs for a check box. `menu.html` provides an interface such as a drop-down menu that can be used to select the city, town or village, shopping item, and survey year in order to obtain the required degree of dependence and the outflow rate. Functions called by events from this interface are defined in `index.html`. First, in the script part of `index.html`, an `XMLHttpRequest` object is created, as follows:

```
xmlhttp = new ActiveXObject("Microsoft.XMLHTTP");
```

This object has the ability to publish HTTP requests, obtain resources and to analyze them as XML, as well as to build DOM trees, but only the first of these abilities is used here. In other words, the object is utilized as a simple HTTP client component. The `getResultSet` function receives the URL of the resource as an argument and initiates communication between the web browser and the server that has the resource using the `open` method of `XMLHttpRequest`. The process that occurs when the server communication status changes is specified in the `onreadystatechange` method as a function. When the data have been received in the correct way they are stored in `res`, and `res` is returned as the value of this function.

The `getDepend` function passes the URL of the CGI script (`getDepend.cgi`) of the server to the `getResultSet` function along with a parameter that is necessary for calculation, and then receives the IDs of the cities, towns and villages and any color

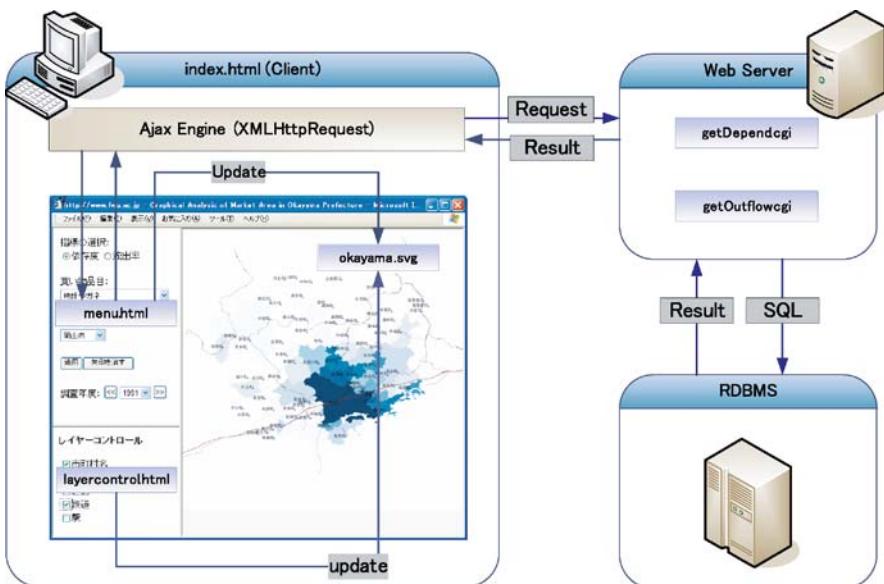


Figure 18.22. Structure of the Okayama market area analysis system

data that are needed to draw a choropleth map. A choropleth map is then created by calling the `fillColor` function with these data as its argument.

18.5.4 Authoring Tool for SVG Statistical Graphics in R

RSvgDevice

R (R Project, 2007) is one of the most popular open source software packages and is supported by numerous statisticians. R can output various statistical graphs. The library RSvgDevice (Luciani, 2005), developed for R by T. Jake Luciani, can treat SVG as a graphic device (as well as devices such as postscript and pdf) in R. Using this library, the user can generate a statistical graph in SVG format using the R commands. Even if the user has no knowledge of the SVG format, the user can benefit from the advantages of SVG by presenting the generated SVG files on the Web or by investigating the effects of small modifications made using a text editor. For example, it may not be realistic to describe all of the parts of a scatter plot using SVG in a text editor, but using RSvgDevice, we can create the plot by simply changing the graphic device to SVG using R command `devSVG`.

In this way, we can obtain the SVG output as shown in Fig. 18.23. There is little change in the labor required to input the command that outputs the graphics in other devices. In addition, we can immediately use functions such as expansion, reduction and movement, as shown in Fig. 18.24.

The data is the *Cars* data from DASL (DASL Project, 1996), which are measurements from 38 1978–1979 model automobiles, specifically the gas mileage in miles

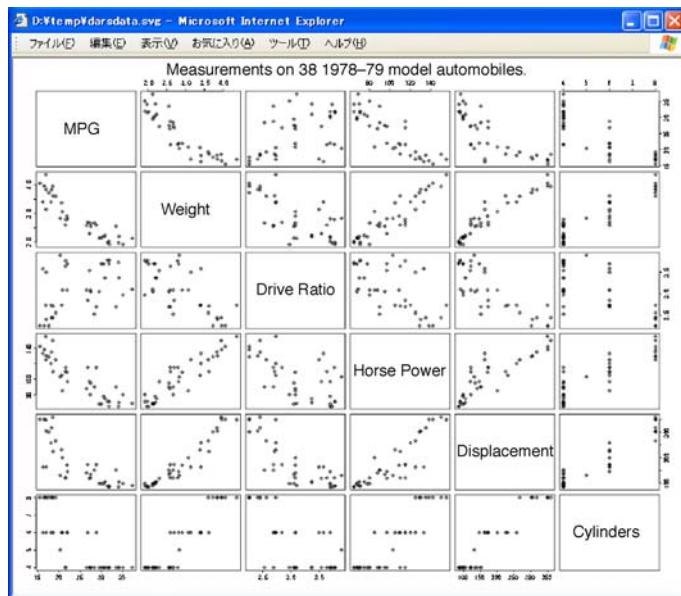


Figure 18.23. A scatter plot matrix created by RSvgDevice

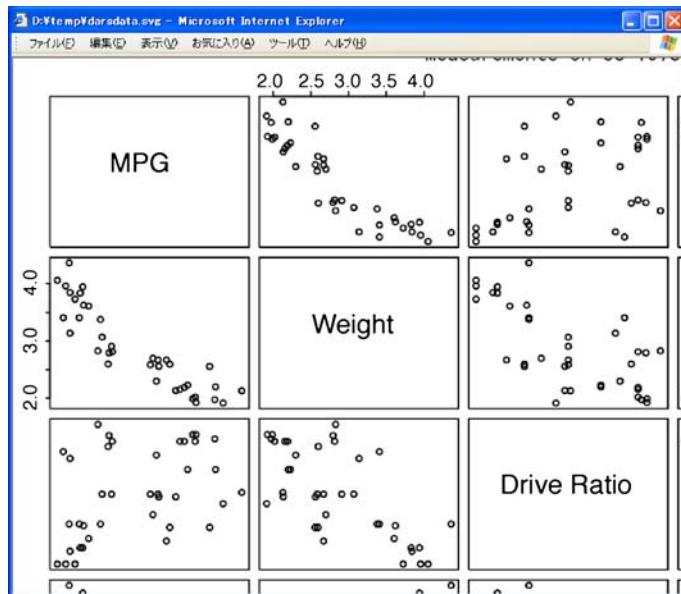


Figure 18.24. Close-up of the scatter plot matrix

per gallon as measured by a consumers' union on a test track and other values reported by the automobile manufacturer. The measurements comprise six variables in all, making it difficult to comprehend Fig. 18.23, but Fig. 18.24 gives clearer information.

RInG Library

If we use techniques such as tooltip display or layer changing, as described in the section on SVG scripting, we can develop interactive statistical graphs. However, it is difficult for users who have no knowledge of JavaScript or DOM to apply these techniques. In addition, RSvgDevice does not have a function that adds interactive functionality, and can only output drawing information in SVG form. While interactive functionality can be obtained by editing the SVG graphs that are output using RSvgDevice, such functionality requires a great deal of work, including adding a script, grouping the elements, and numbering the ID. To solve such problems, we developed the R Interactive Graphics (RInG) library. The RInG library provides an R function that outputs fundamental statistical graphs, including an interactive function, in SVG form. At present, the source code and binary package of the Windows version can be downloaded from our website. After installing this package, the user can use the RInG library by invoking the command `library(ringlib)`.

In the RInG library, we provide three functions that output fundamental interactive statistical graphs. `iplot` is a function that outputs a two-dimensional scatter plot with interactive features using SVG. The interactive statistical graphs provided by this function have the following features. When the mouse cursor is positioned over a data point, the value of the data point is displayed by a tooltip, and additional

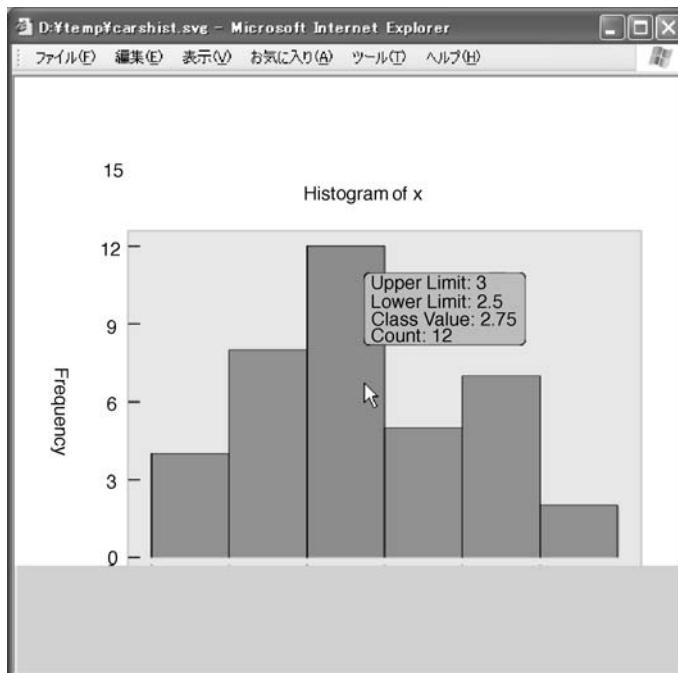


Figure 18.25. Histogram obtained using the `iplot` function of the RInG library

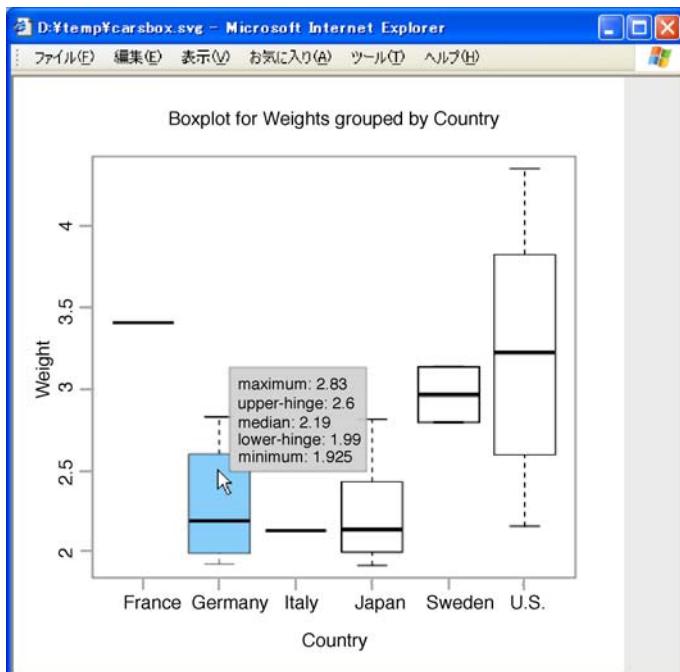


Figure 18.26. Boxplot obtained using the `iplot` function of the RInG library

lines running from the data point to both the x - and y -axes of the graph are also displayed.

`ihist` is a function that outputs the histogram in SVG form with interactive features. For example, when the mouse cursor is positioned over a bar in the histogram, information for this bar such as its upper and lower values, its class, and the number of observations included in it are displayed as a tooltip. The histogram shown in Fig. 18.25 is the output of the command `ihist(rnorm(1000), breaks="FD")`.

`iboxplot` is a function that outputs a boxplot in SVG form with interactive features. If the mouse cursor is positioned over a box in the boxplot, Tukey's five number summary is displayed as a tooltip. In addition, the values of outlying data points are displayed as tooltips when the mouse cursor is positioned over them.

Typing the following command yields a boxplot like that shown in Fig. 18.26:

```
> iboxplot(split(Weight,Country),file="carsbox.svg",
  xlab="Country", ylab="Weight",
  main="Boxplot for Weights grouped by Country")
```

These functions are available in server applications such as the “Okayama trade area analysis system” described in the previous section. Figure 18.21 is an example of dynamic graphics. This dynamic functionality is implemented using the RInG library, which can output the SVG code directly to the standard output as well as to a file. Therefore, an interactive plot that is dynamically generated without a temporary file according to the request of the user can be provided through the Web using a combi-

nation of the RInG and CGIwithR libraries. While many CGI-based systems use R, most of them invoke R as a child process of an interpreter of a script language such as Perl. However, by using that combination it has been possible to achieve from communication with RDBMS to statistical computing and graphics only by R.

References

- Boyens, C., Günther, O. and Lenz, H.-J. (2004). Statistical Databases. In: Gentle, J.E., Härdle, W., Mori, Y. (eds) *Handbook of Computational Statistics – Concepts and Methods*. Springer, Berlin.
- Eisenberg, J.D. (2002). *SVG Essentials*. O'Reilly, Sebastopol, CA.
- Fitzgerald, M. (2004). *XML Hacks*. O'Reilly, Sebastopol, CA.
- Fujino, T., Yamamoto, Y. and Tarumi, T. (2004). Possibilities and Problems of the XML-Based Graphics in Statistics. In: Antoch, J. (ed) *COMPSTAT2004 Proceedings in Computational Statistics*. Physica, Heidelberg, pp. 1043–1052.
- Geroimenko, V. and Chen, C. (eds) (2004). *Visualizing Information Using SVG and X3D*. Springer, Berlin.
- Geroimenko, V. and Chen, C. (eds) (2003). *Visualizing the Semantic Web*. Springer, Berlin.
- Härdle, W., Klinke, S. and Müller, M. (2000). *XploRe Learning Guide*. Springer, Berlin.
- Haselett, J., Bradley, R., Craig, P., Unwin, A. and Wills, G. (1991). Dynamic Graphics for Exploring Spatial Data With Application to Locating Global and Local Anomalies. *The American Statistician*, 45(3):234–242.
- Klinke, S. (2004). Statistical User Interfaces. In: Gentle, J.E., Härdle, W., Mori, Y. (eds) *Handbook of Computational Statistics – Concepts and Methods*. Springer, Berlin.
- Marchette, D.J. (2004). Network Intrusion Detection. In: Gentle, J.E., Härdle, W., Mori, Y. (eds) *Handbook of Computational Statistics – Concepts and Methods*. Springer, Berlin.
- Meyer, D., Leisch, F., Hothorn, T. and Hornik, K. (2004). StatDataML: An XML Format for Statistical Data. *Computational Statistics*, 19(3):493–509.
- Mori, Y., Fujino, T., Yamamoto, Y. and Tarumi, T. (2004). XML-based Applications in Statistical Analysis. In: *Proceedings of Interface 2004: Computational Biology and Bioinformatics*, 36th Symposium on the Interface, 26–29 May 2004, Baltimore, MD.
- Mori, Y., Fujino, T., Yamamoto, Y., Kubota, T. and Tarumi, T. (2004). XML-based Applications in Statistical Analysis. In: *Proceedings of Interface 2004: Computational Biology and Bioinformatics*, 36th Symposium on the Interface (CD-ROM), 26–29 May 2004, Baltimore, MD.
- Symanzik, J. (2004). Interactive and Dynamic Graphics. In: Gentle, J.E., Härdle, W., Mori, Y. (eds) *Handbook of Computational Statistics – Concepts and Methods*. Springer, Berlin.
- Wilhelm, A. (2004). Data and Knowledge Mining. In: Gentle, J.E., Härdle, W., Mori, Y. (eds) *Handbook of Computational Statistics – Concepts and Methods*. Springer, Berlin.

Yamamoto, Y., Iizuka, M. and Fujino, T. (2005). *Consideration for Developing Environments of Web-based Interactive Statistical Graphics*. 55th Session, International Statistical Institute, Sydney, Australia.

- @d Project (2006). @d – Data Oriented Statistical System. <http://mo161.soci.ous.ac.jp/@d/>
- Adobe (2007). Flash. <http://www.macromedia.com/flash/>
- Adobe (2007). SVG Viewer. <http://www.adobe.com/svg/>
- Apache Software Foundation (2007). Batik SVG Toolkit. <http://xml.apache.org/batik/>
- BMBF (2007). EMILeA Stat. <http://www.emilea.de/>
- DandD Project (2007). <http://stat.math.keio.ac.jp/DandDIV/>
- DASL Project (1996). DASL – The Data and Story Library.
<http://lib.stat.cmu.edu/DASL/>
- ISR, Univ. Michigan (2007). ICPSR (The Interuniversity Consortium for Political and Social Research). <http://www.icpsr.umich.edu/>
- Luciani, T.J. (2005). Scalable Vector Graphics for R: RSvgDevice. <http://www.darkridge.com/~jake/RSvg/>
- MD*Tech (2005). Xplore e-books. <http://www.xplore-stat.de/ebooks/ebooks.html>
- MD*Tech (2007). Xplore homepage. <http://www.xplore-stat.de/>
- MD*Tech (2007). MD*Base. <http://www.quantlet.org/mdbase/>
- MD*Tech (2007). MM*Stat. <http://www.quantlet.com/mdstat/mmstat.html>
- Meyer, D. (2004). The StatDataML Package. <http://www.omegahat.org/StatDataML/>
- Octaga AS (2007). Homepage. <http://www.octaga.com/>
- Open Geospatial Consortium (2004). GML – the Geography Markup Language. <http://opengis.net/gml/>
- Project CASE (2007). Homepage (in Japanese). <http://case.f7.ems.okayama-u.ac.jp/>
- Project Jasp (2005). Jasp – Java-Based Statistical Processor. <http://jasp.ism.ac.jp/index-e.html>
- R Project (2007). Homepage. <http://www.r-project.org/>
- UCLA Department of Statistics (2007). Homepage. <http://www.stat.ucla.edu/>
- University of Hagen (2007). Multimedia Project: New Statistics. <http://www.fernuni-hagen.de/newstatistics/>
- Vlachos, P. (2007). Statlib – Data, Software and News from the Statistics Community. <http://lib.stat.cmu.edu/>
- W3C (World Wide Web Consortium). <http://www.w3.org/>
- W3C (2007). Semantic Web. <http://www.w3.org/2001/sw/>
- W3C (2007). SMIL, the Synchronized Multimedia Integration Language. [http://www.w3.org/](http://www.w3.org/<AudioVideo/)
[AudioVideo/](#)
- W3C (2007). SVG (Scalable Vector Graphics). <http://www.w3.org/Graphics/SVG/>
- Web3D Consortium (2007). Homepage. <http://www.web3d.org/>
- Web3D Consortium (2007). X3D: Extensible 3D. <http://www.web3d.org/x3d/>
- WISE Project (2007). WISE – Web Interface for Statistical Education. <http://wise.cgu.edu/>

Part IV

Selected Applications

Visualization for Genetic Network Reconstruction

IV.1

Grace S. Shieh, Chin-Yuan Guo

1.1	<i>Introduction</i>	794
1.2	<i>Visualization for Data Preprocessing</i>	794
	Outlier Detection	794
	Data Augmentation	795
1.3	<i>Visualization for Genetic Network Reconstruction</i>	796
	Clustering and Graphical Models	797
	A Time-lagged Correlation Approach.....	799
	A Smooth Response Surface Approach	801
	A Regression Approach.....	802
	A Pattern Recognition Approach	806

Introduction

This chapter reviews data visualization techniques that are used to reconstruct genetic networks from genomics data. Reconstructed genetic networks are predicted interactions among genes of interest and these interactions are inferred from genomics data, e.g., microarray data or/and DNA sequence data. Genomics data are generally contaminated and high-dimensional. The dimensionality of a microarray is the number of genes in it, and it usually numbers in the thousands at least. It is important to examine and clean data carefully to attain meaningful inferences. Thus visualization tools that are used in the preprocessing of data associated with genetic network reconstruction are also reviewed.

With the advent of high-throughput genomics and proteomics data, simultaneous interrogation of the status of each component in a cell is now possible. One challenging issue in the post-genomics era is to understand the biological interaction networks that are relevant to cell function (Barabási and Oltvai, 2004). Biological networks include protein–protein interaction pathways and genetic networks. The latter include direct interactions or those in which the biological principle is known, such as transcriptional networks and metabolic networks, as well as subtle ones like synthetic genetic interaction networks (Tong et al., 2001; Tong et al., 2004), transcriptional compensation interactions (Wong and Roth, 2005), among others. Although protein chips and semiautomatic yeast two-hybrid screens are available, they are not yet as widely used as microarray gene expression data. Hence, in this article we focus on visualization for genetic network reconstruction from microarray data.

With the abundant information produced by microarray technology, various approaches to inferring genetic networks have been proposed. Most of them can be grouped into three classes: discrete variable models, continuous variable models, and graph models. The discrete variable models discretize gene expression into a few states. The dynamics of gene expression may be perceived as transitions of finite states. Typical discrete variable models are Boolean networks (Liang et al., 1998; Akutsua et al., 2003) and discrete Bayesian networks (for example, Friedman et al., 2000). In general, continuous variable models characterize the expression of a gene or changes in it as a linear or nonlinear function of other genes (for instance, Beal et al., 2005). Graph models, for example Schäfer and Strimmer (2005), depict genetic interactions through directed graphs (“digraphs”) instead of characterizing the interactions quantitatively. For exhaustive literature reviews of both static and dynamic models used to reconstruct genetic networks, we refer the reader to De Jong (2002), van Someren et al. (2002), and Shieh et al. (2005).

Visualization for Data Preprocessing

Outlier Detection

Although many microarray data sets in yeast have been made available, microarray experiments conducted under similar treatments are still sparse compared to the

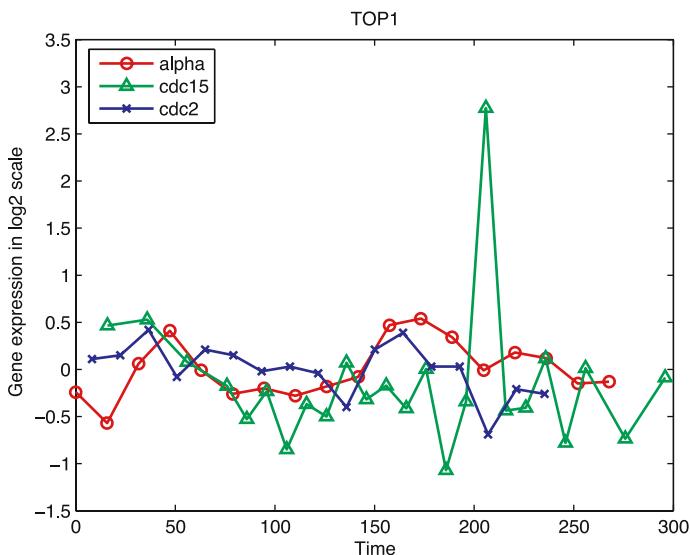


Figure 1.1. Aligned expression of the gene *TOP1* in the alpha, cdc15 and cdc28 data sets

complex genetic networks to be reconstructed. Therefore, the integration of microarray data sets from homogeneous or similar, but not identical, experimental conditions is of interest. Bar-Joseph et al. (2002) developed an algorithm to align data sets; for example, they aligned alpha, cdc15 and cdc28 in Spellman et al. (1998). The cDNA microarray gene expression data were from synchronized yeast cells (red-channel intensities) versus nonsynchronized ones (green-channel intensities that served as background signals). Note that all data sets except for cdc28 were processed by normalization procedures in Yang et al. (2001); cdc28 data were provided in log ratios only, so normalization could not be applied. Figure 1.1 depicts the curves of a given gene's expression (in log ratios) from the three aligned data sets. Consistency across all three curves (data sets) supports the validity of the data, whereas any inconsistency in one curve with respect to the other curves suggests a potential outlier. For example, in Fig. 1.1 the gene expression of *TOP1* at 205 minutes in the cdc15 data set is likely to be an outlier, since it is very different to its corresponding points in the alpha and cdc28 data sets. In addition, the small ups and downs at the fifth and later points of the cdc15 data set indicate that the data quality of the cdc15 is worse than that of the alpha and the cdc28 data sets. Hence, aligning data sets with similar experimental conditions provides a route to the detection of outliers or noisy data visually; it also helps to exclude patterns suggested by contaminated data.

Data Augmentation

1.2.2

There are 18, 24, 17 and 14 time points without replicates in the alpha, cdc15, cdc28 (originally from Cho et al., 1998), and Elu microarray data sets in Spellman et al. (1998). To augment data for inference, Xie and Bentler (2003) integrated these four

data sets to infer the regulation of latent factors (to model protein, RNA degradation and other factors that cannot be measured by microarrays) on genes by structure equation modeling.

Without augmenting data in the aforementioned four data sets, those time points (sample sizes) are not sufficient to obtain statistical inferences. However, in order to carefully distinguish which data sets among alpha, cdc15, cdc28 and Elu to integrate, Shieh et al. (2005) applied the algorithm in Bar-Joseph et al. (2002) to align the four data sets in the same time domain. Figure 1.2 clearly shows that alpha, cdc15 and cdc28 exhibit similar trends, while Elu does not. This observation is consistent with the following biological argument. The Elu data set was synchronized by elutriation, unlike the other three sets, which were treated by pheromone α factor, cdc15 and cdc28 mutant, respectively. Based on Fig. 1.2, they integrated those three data sets and applied the proposed structural equation modeling algorithm to the integrated data in order to infer genetic interactions. A gene network of five genes that are synthetic

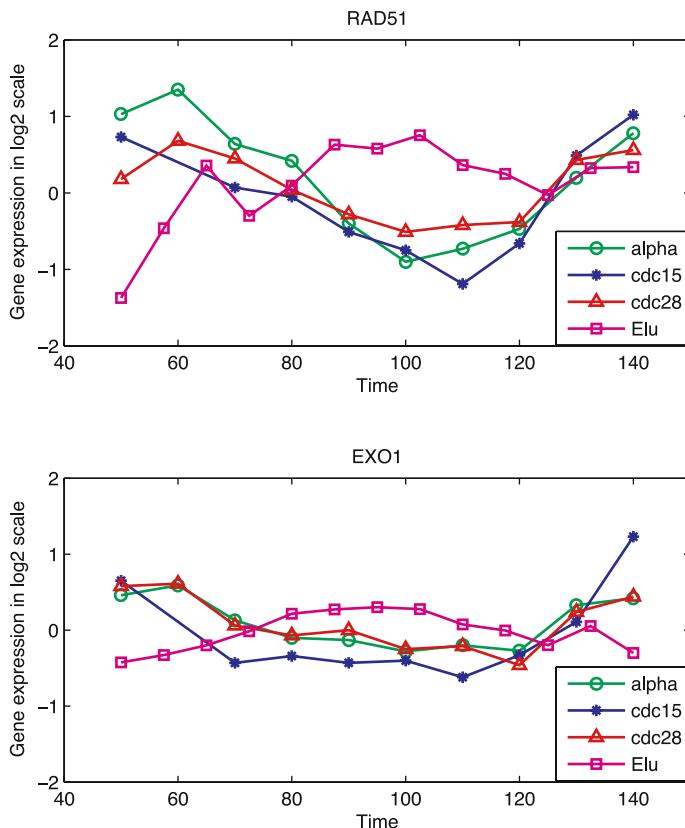


Figure 1.2. The expression levels of two genes in the four aligned data sets; each gene's expression pattern in Elu is clearly different from its corresponding patterns in the other sets

sick or lethal (SSL) to SIS2 (Tong et al., 2001; Tong et al., 2004) was predicted; checked against the quantitative RT-PCR, the inferred results are satisfactory.

Visualization for Genetic Network Reconstruction

1.3

Clustering and Graphical Models

1.3.1

Cell cycle regulated genes display periodicity through two cell cycles in the alpha, cdc15 and cdc28 data sets of Spellman et al. (1998). For genes that show periodic expression across phases, the target gene *B* may be enhanced in the S phase but it may be repressed or there may be no interaction with the gene *B* in other phases. This suggests that the interaction between two given genes may vary from phase to phase over a cell cycle. Due to data limitations, most of the approaches used so far have assumed that genetic interactions are invariant across time or phase, but there are some approaches that have dealt with genetic interactions that vary with phase, e.g., Toh and Horimoto (2002) and Aburatani et al. (2006). These authors utilized clustering and a graphical model to infer interactions of *pseudogenes* that may vary across phases, where a pseudogene denotes a group of genes clustered via similar gene expression patterns.

Modeling genetic interactions that vary with phase requires more (four times more for four phases in a cell cycle) data than those that have interactions that are invariant with phase. To reduce the huge number of interactions among genes in a given genome, graphical model requires that clustering is performed as a preprocessing step. Namely, coexpressed genes are grouped into clusters, and each cluster is treated as a random variable to infer its interaction with other clusters. For instance, a hierarchical clustering analysis was applied in Toh and Horimoto (2002) to aggregate 2467 genes into 34 clusters. The averaged expression profile of each cluster was then analyzed by a graphical Gaussian model (GGM). Inevitably, this makes it more difficult to interpret the meaning of such interactions. This GGM inferred direct interactions between variables based on their partial correlations, which is a function of the elements in Σ^{-1} , where Σ is the covariance matrix.

A graph $G = (V, E)$, where V and E denote a set of vertices (variables) and a set of ordered edges (the associations between pairs of variables), represents the interactions among M clusters. Note that G assumes the Markov property. The chain graph model (Aburatani et al., 2006) consists of the following steps. First, the variables are partitioned into a few ordered blocks, e.g., four blocks for four phases. Second, within each block, the conditional independence of each pair of variables is tested by a likelihood ratio test given the rest of the variables in the block. This likelihood ratio test is based on the inverse of the covariance matrix Σ^{-1} . If the conditional direct association of two given variables is significant, then an undirected edge is drawn between them. Third, one can similarly test for the conditional independence between any

two variables given the rest of the variables in blocks k and l . If the conditional association between variables i and j in blocks k and l is significant, an arrow pointing from k (blocks with lower index) to l (blocks with higher indices) is plotted. Likewise, the fitting continues until all significant direct associations between variables within each block and between any two blocks are determined. Figure 1.3, which is reproduced from Fig. 1.1 of Aburatani et al. (2006), depicts the interactions of pseudogenes within and between phases. The causal relationship between variables in one phase to those in a latter phase is illustrated more clearly by a chain graph, for example Fig. 1.3c, than by formula and tables. Moreover, a graph also clearly demonstrates which gene is the highly connected "hub gene," for example $C_{1,3}$ in G_1 among the four pseudo-genes. Hub genes are likely to be important because random mutations in organisms lacking these genes would likely lead to fitness defects. By comparing graphs of genetic networks among different species, one may infer which hub genes are conserved; this may lead to important applications to complex diseases in humans. Tong et al. (2004) observed that synthetic genetic interactions have the property that the connectivity of genes follows a power law distribution. Namely, many genes have few interactions and a few genes have many interactions. Networks of the World Wide Web (WWW) and protein–protein interactions also share this power law property. Figure 1.3c also shows that graphs can help to identify networks whose connectivity distributions follow the power law. Further, graphical similarity may indicate that analytic methods applied in the area of the WWW can also be applied to the areas of genetic and protein–protein interactions.

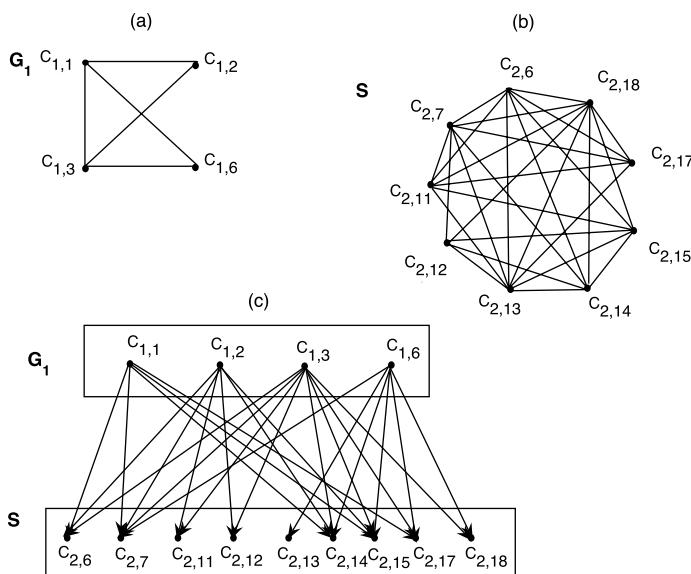


Figure 1.3. Chain graphs used to depict genetic interactions within and across phases

A Time-lagged Correlation Approach

1.3.2

In order to study the mechanism of the physiological response of the photosynthetic cyanobacterium *Synechocystis sp.* to alternating light conditions, Schmitt Jr. and his colleagues conducted time series experiments (Schmitt Jr. and Stephanopoulos, 2003). In these experiments, a culture of *Synechocystis sp.* was exposed to serial perturbations in light intensity; 47 samples were successfully hybridized to DNA microarrays. The time series experiments were set 20 min apart over periods of eight and sixteen hours. The transcriptional networks of *Synechocystis sp.* that responded to various light intensities were of interest. Postulating that the expression level of target gene 1 follows the pattern of the input with a time lag, and similarly gene 2 follows its regulating gene 1 with another time lag, as depicted in Fig. 1.4, Schmitt Jr., Raab and Stephanopoulos (2004) used a time-lagged correlation approach to infer transcriptional interactions. Let $x_i(t)$ denote the expression level of gene i at time t and \bar{x}_i be the average expression level of gene i across the time points. The time-lagged correlation between genes i and j , proposed in Arkin and Ross (1995) and Arkin et al. (1997), is defined as $r_{ij}(\tau) = s_{ij}(\tau)/\sqrt{s_{ii}(\tau)s_{jj}(\tau)}$, where $s_{ij} = \sum_{t=1}^T (x_i(t) - \bar{x}_i)(x_j(t + \tau) - \bar{x}_j)$ and $0 \leq \tau < T$.

Let $\mathbf{R}(\tau)$ denote the $n \times n$ matrix consisting of lagged- τ correlation between any pair from n genes. The matrix $\mathbf{R}(\tau)$ can be used to rank the correlation and anticorrelation between genes via a Euclidean distance metric d_{ij} . The metric D_{ij} assumes the form $d_{ij} = (c_{ii} - 2c_{ij} + c_{jj})^{1/2} = \sqrt{2}(1 - c_{ij})^{1/2}$, where the maximal correlation $c_{ij} = \max_{\tau} |r_{ij}(\tau)|$. The matrix $\mathbf{D} = (d_{ij})$ is a distance-based correlation that maps genes that are the least correlated (for any τ) – the “farthest” apart. By using the matrix \mathbf{D} , Schmitt Jr. et al. (2004) were able to include both highly correlated and anticorrelated genes for further analysis. Using the modified time-lagged correlation method, the input profile was employed as a “seed” to find genes that had time-lagged correlated expression profiles, where the input signal profile consisted of the autoscaled light intensity values at each time point.

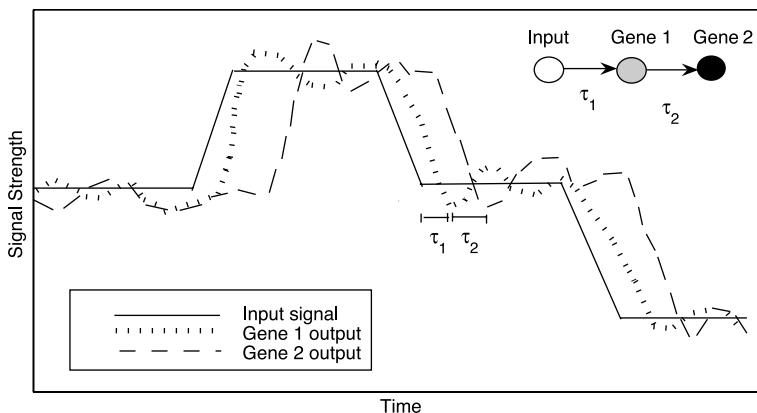


Figure 1.4. The time-lagged correlations between input and gene, and between cascades of genes

Specifically, this time-lagged correlation approach comprised the following steps.

Step 1: Genes with low expression levels or low expression changes were filtered.

Furthermore, coexpressed genes ($|R(0)| > 0.7$), which are located adjacently in DNA sequences, were clustered. All clustered genes (their averaged gene profiles) and nonclustered genes were submitted for further analysis in Step 2.

Step 2: Lag- τ correlations between expression profiles were computed, using the input signal in the first iteration or the averaged profiles of each subgroup resulting from Step 3 in subsequent iterations.

All clusters and genes with at least one $|r(\tau)|$ value that is greater than the pre-specified threshold of 0.7 were retained for Step 3.

Step 3: Genes retained from Step 2 were partitioned by the time-lag τ from their lagged correlations. For instance, all genes that were best correlated with lag-1 were grouped into category 1. Within each category, a nearest neighbor clustering method (Dillon and Goldstein, 1984) was applied to cluster the genes into subgroups using the usual correlation as the similarity metric.

Step 4: The significantly correlated groups in each category were used as “seed” nodes in Step 3 to expand the interaction network. Iterations were then stopped, provided that the network could not be further expanded for a given threshold.

Step 5: The Graphviz program from ATT research labs (<http://www.graphviz.org/>) was adapted in order to minimize the crossovers in a given network, while the software for displaying graphics was written in Matlab.

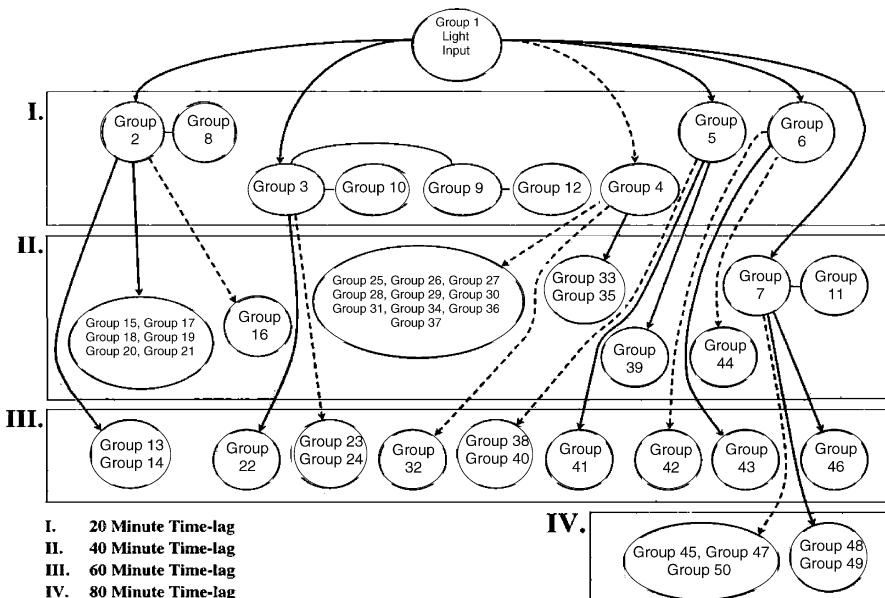


Figure 1.5. Simplified time-lagged correlation network across four time points; second iteration; where →, -→ and -- denote lagged correlation, lagged inverse correlation and zero-lagged correlation between groups

This time-lagged correlation approach resulted in a predicted network comprising 50 groups of 259 genes; a simplified network, obtained at iteration 2, is shown in Fig. 1.5. The graph illustrates zero-lagged highly correlated groups at the same time point. Moreover, it depicts groups that are correlated between different time points, which demonstrates the dynamic interactions between groups across time much more clearly than when any analytical method is used. Among the predicted groups, several have known light-stimulated gene clusters – for example carbon dioxide fixation pathways – while others are novel findings.

A Smooth Response Surface Approach

1.3.3

Let A , R and T denote the activator (enhancer), the repressor and their regulated target gene, respectively. Woolf and Wang (2002) proposed that the interaction relation of A , R and T could be elucidated via a fuzzy logic algorithm, where A and R are inputs and T is the output. The fuzzy logic function used is discrete and has the drawback that it can map two adjacent boundary inputs to two quite different outputs. The fuzzy logic algorithm was improved by the smooth response surface approach (Xu et al., 2002), which provides a continuous regulatory influence that has biological bearing. Xu and his colleagues proposed that the triplets could be fitted to the following response surface:

$$S(A, R) = \begin{cases} 2A(1 - R), & 0 \leq A \leq 0.5, 0.5 \leq R \leq 1 \\ 1 - 2(1 - A)R, & 0.5 \leq A \leq 1, 0 \leq R \leq 0.5 \\ A - R + 0.5, & \text{otherwise.} \end{cases} \quad (1.1)$$

This approach captures the basic idea that an activator increases its regulated target gene's expression level while a repressor decreases its target gene's expression level. To fit gene expression levels of triplets into certain fixed surfaces (Wu and Hamada, 2000 and Xu et al., 2002), gene expression levels (in \log_2 scale) were transformed into the interval $[0, 1]$. The minimum and maximum values of each gene were transformed into 0 and 1, respectively.

If a given triplet fits the (A, R, T) relation specified by the response surface in (1.1) well, the predicted target gene's expression over time should be close to the observed one's, and the mean squared error should be small compared to the target gene's variance. Thus the lack-of-fit and diagnostic functions are defined as follows:

$$RT(A, R, T) = \frac{\sum_{t=1}^{T_0} (T_t - \hat{T}_t)^2}{\sum_{t=1}^{T_0} (T_t - \bar{T})^2} \quad (1.2)$$

and

$$\text{Diag}(A, R, T) = \frac{\left[\frac{1}{T_0} \sum_{t=1}^{T_0} \{ RT_{(t)}(A, R, T) - RT(A, R, T) \}^2 \right]^{1/2}}{RT(A, R, T)}, \quad (1.3)$$

where T_0 is the total number of time points and $RT_{(t)}(A, R, T)$ denotes the lack-of-fit score of (A, R, T) with the t th sample deleted. From (1.3), the diagnostic function can apparently screen out any gene that has one or more time points that deviate greatly from the model in (1.1). A large $Diag$ value suggests that either gene expression levels of the triplet at one or more time points may be problematic or that the triplet does not fit the response surface well. It is clear that the criteria that should be used to check whether a triplet fits the response surface well are $RT(A, R, T) \leq C_1$ and $Diag(A, R, T) \leq C_2$, where the C_i values are constants. The following score function was defined to measure the overall fitting:

$$\text{Score}(A, R, T) = RT(A, R, T)[1 + Diag(A, R, T)]. \quad (1.4)$$

The SRS algorithm was applied to two sets of microarray data: GeneChip data on the yeast cell cycle (Cho et al., 1998) and cDNA microarray data on the yeast cell cycle (Eisen et al., 1998). There were 6457 genes with 17 time points from the affymetrix Ye6000 chip in the former set. After filtering, 1514 genes were retained and processed by the SRS algorithm. Among those triplets formed, 20500 had RT scores of < 0.1 and $Diag < 2.0$. Those best-scoring triplets that have biological meaning are plotted in Fig. 1.6 (Fig. 5 in Xu et al., 2002). A few targets in the network of Fig. 1.6 associated with regulators either carry out similar cellular functions or are involved in the same cellular process. For instance, CDC9 is an essential gene for cell division and DNA recombination; four regulators of CDC9 (SMCI, NIP29, BTT1, NUM1) are functionally related. HAP1 is a transcription factor; the predicted repression of HAP1 on CYC7 agrees with the literature that HAP1 represses CYC7 under anaerobic growth and activates CYC7 under aerobic growth. FAA1 and HES1, which are related to cellular lipid metabolism and ergosterol biosynthesis, have been implicated in HAP1 regulation in the literature.

Besides those described in Xu et al. (2002) and Shieh et al. (2004), there is a wide range of software available for the visualization of gene networks; for instance, for visualizing biomedical networks (BioMiner, Graphviz, Ospray, among others), for integrating genomics and proteomics data from external databases to visualize pathways or genetic networks (Dynamic Signal Maps, KnowledgeEditor, PathFinder, Pathway Assist, PubGene, Vector, PathBlazer and others), and for modeling and simulating gene regulatory networks (Genetic Network Analyzer, GenePath, among others).

1.3.4

A Regression Approach

Recently, there have been a few studies on *transcriptional compensation* (TC) interactions (Lesage et al., 2004; Kafri et al., 2005; Wong and Roth, 2005). Following the loss of a gene, the expression of its compensatory gene increases; this phenomenon is known as TC. Reverse-transcription (RT)-PCR experiments have shown that, aside from TC, compensatory gene expression decreases in some cases following the absence of a gene; we call this phenomenon *transcriptional diminishment* (TD). The TC interactions among a group of 51 yeast genes, which are synthetic sick or lethal to *SGS1* or *RAD27* (Tong et al., 2001), are of interest (Shieh et al., 2004). The SRS algo-

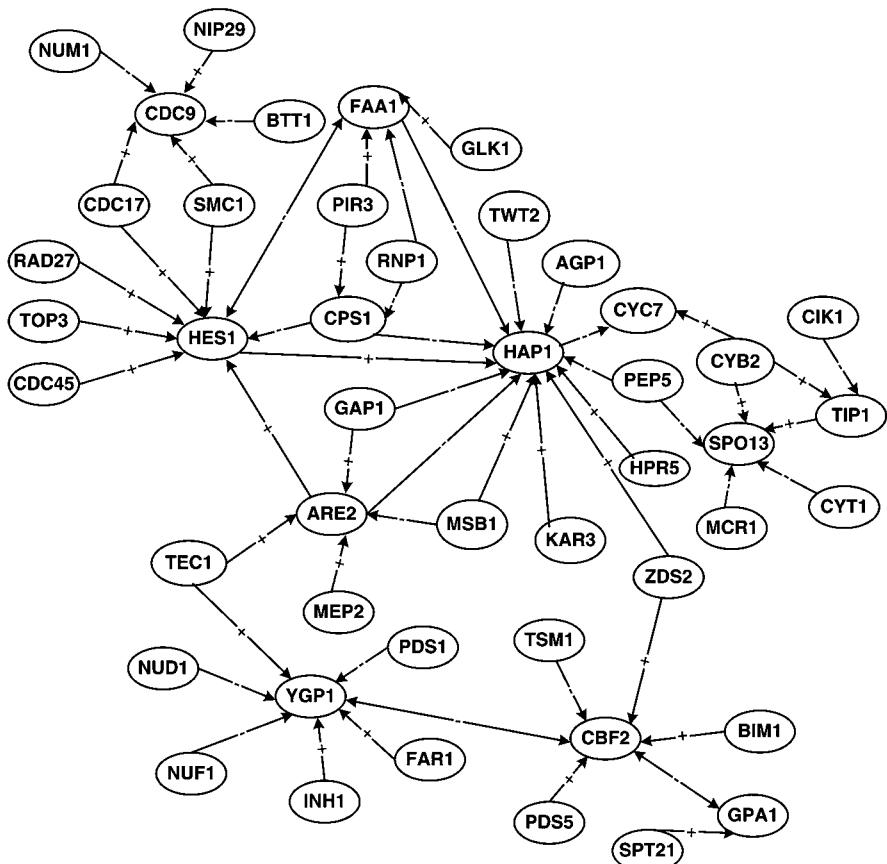


Figure 1.6. Reconstruction of a genetic network based on the fitting of the top-scoring triplets to Xu's model, where $A \rightarrow + \rightarrow - \rightarrow B$ denotes that A activates ($\rightarrow +$) B , respectively

rithm is a pioneer application of the response surface method that is used to reconstruct genetic networks. Unfortunately, it failed to reconstruct the TC interactions of interest. This stimulated Shieh and her colleagues to develop a regression approach, as described below.

Although the SRS approach utilizes the principle that an activator activates its target gene's expression and a repressor represses its target gene's expression, the coefficients of the response surface in Equation (1.1) were based only on that the effect of A (R) is positive (negative) and the resulting expression level of the target gene falls in the interval $[0, 1]$ (personal communication with Xu). In fact, there are an infinite number of sets of coefficients that satisfy the boundary condition. In biology, genes with similar functions may have similar interaction patterns. Similar patterns of interactions tend to identify components from the same biological pathway (Tong et al., 2004). Therefore, it was proposed that the surface should be determined by the data,

in particular, by the majority of triplets. Furthermore, a time lag was allowed in the model for the target gene to respond to the influences of its activator and repressor.

Given n genes, in order to attain the surface that fits the majority of triplets (A, R, T) , the following regression model was fitted to all $\binom{n}{3} \cdot 3!$ triplets:

$$T_i(t+1) = \beta_0 + \beta_1 A_i(t) + \beta_2 R_i(t) + \beta_3 A_i(t)R_i(t), \quad (1.5)$$

where $1 \leq i \leq n$, $\beta_1 > 0$, $\beta_2 < 0$ and $\beta_3 \in \mathbf{R}$.

Fitting (A_t, R_t, T_{t+1}) to the response surface with a time lag has the following biological bearing. Microarrays measure the concentration of mRNAs, the lag-1 in time characterizes that mRNAs of gene A (R) translate into protein a (r) within the time lag, then their protein a (r) activates (represses) its associated target gene. An A-R-T relationship, with T lagged -1 in time behind A and R , is depicted in Fig. 1.7, where the curves of R and T are roughly converse to each other but T 's curve roughly follows A 's. A few RT-PCR-confirmed A-R-T triplets have also shown similar patterns to those in Fig. 1.7; this reinforces the validity of the A-R-T model.

Time course microarray data for each triplet were fitted to the model in (1.5) to obtain $(\hat{\beta}_0, \hat{\beta}_1, \hat{\beta}_{2i}, \hat{\beta}_{3i})$, where $i = 1, \dots, n(n-1)(n-2)$ and n is the number of genes. That is, there were $n(n-1)(n-2)$ response surfaces (models) fitted in total. Next, the goodness-of-fit criteria that $R^2 > 0.7$ and all of the p -values of the $\hat{\beta}_i$ must be < 0.2 were utilized to select models that were good fits. Due to the multiple testing problem (four $\hat{\beta}_i$ values) and the significance level for entry in variable selection (Younger, 1979), 0.20 was set to be the threshold for all p -values. This threshold and that for R^2 can be adjusted according to the number of triplets that satisfy the goodness-of-fit criterion. For instance, if there are relatively few triplets that satisfy the criterion, one can loosen just the threshold for p -values or both thresholds.

To gain insights from \mathbf{R}^3 , triplets of 51 genes related to DNA synthesis and DNA repair in yeast were fitted to the model in (1.5); those models where $(\hat{\beta}_{1i}, \hat{\beta}_{2i}, \hat{\beta}_{3i})$ satisfied the criterion that $R^2 > 0.85$ and all p -values of $\hat{\beta}_i < 0.15$ were retained,

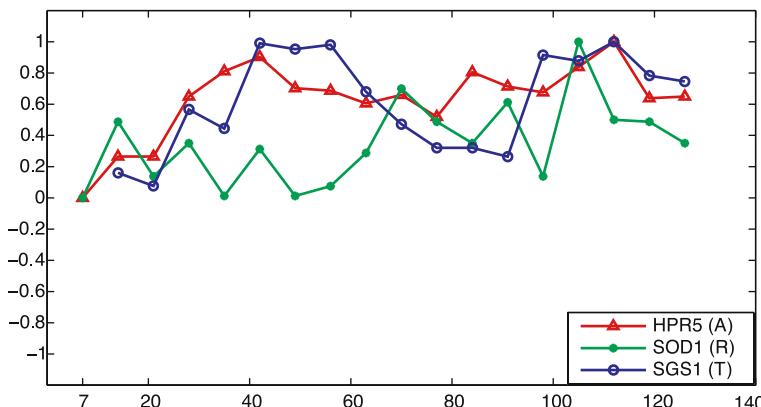


Figure 1.7. The Activator–Repressor–Target pattern displayed by three genes in the alpha data set

where $i = 1, 2, 3$. Those 571 data-fitted models $(\hat{\beta}_{1i}, \hat{\beta}_{2i}, \hat{\beta}_{3i})$ (denoted by +) and Xu's model in (1.1) (denoted by \circ) are plotted in Fig. 1.8. Figure 1.8 clearly shows that Xu's model does not fit the cluster of $(\hat{\beta}_{1i}, \hat{\beta}_{2i}, \hat{\beta}_{3i})$ points well. Instead, the center point of the clustered points could be a better fit. This confirmed the intuition that the A-R-T model should be data-driven. This data-driven response surface was developed as follows.

Among those models (surfaces) fitted well by microarray data of triplets, the following method was employed to determine exactly one surface, called *the mode surface*, that most of the triplets are close to. For any given i , treating $(\hat{\beta}_{0i}, \hat{\beta}_{1i}, \hat{\beta}_{2i}, \hat{\beta}_{12i})$ as a point in \mathbf{R}^4 , we partitioned $[0, 1]^4$ by Silverman's rule in nonparametric density estimation (Scott, 1992; Härdle et al., 2004).

Silverman's rule is a method that determines the mode of points that exist in high-dimensional space; the formula to compute the partition number for each coordinate h_i is:

$$h_i = 0.9 \times \min \{s_i, IQR_i/1.34\} \times n^{-1/(d+4)},$$

where s_i and IQR_i denote the standard error and interquartile range of the data in coordinate i , and d is the dimensionality of the points to be partitioned. A general

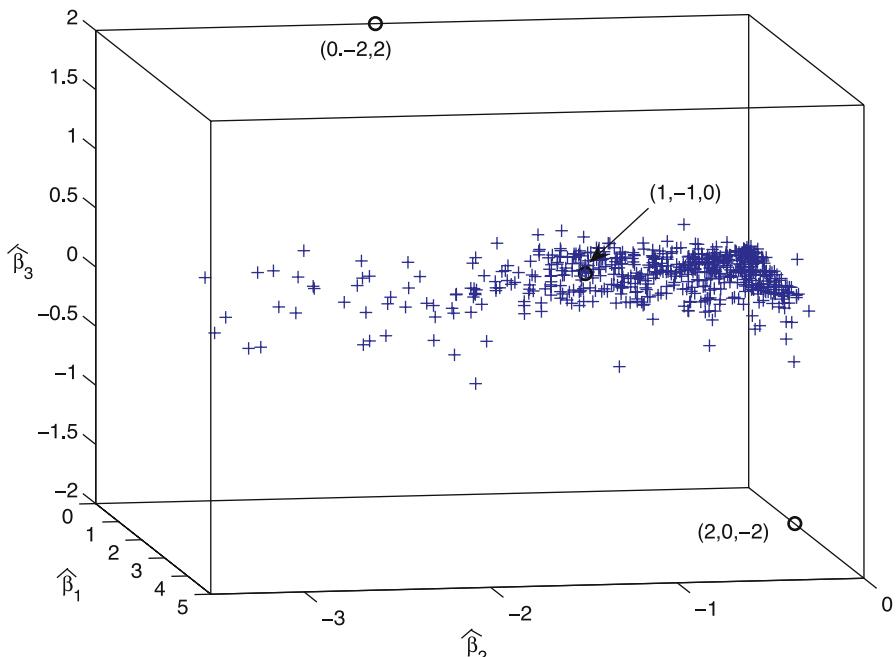


Figure 1.8. $\hat{\beta}_1, \hat{\beta}_2$ and $\hat{\beta}_3$ of models fitted well to (1.5) and those of Xu's model, denoted by + and \circ , respectively

smoothing spline method (Gu, 2002) is currently under investigation as a method to obtain the mode surface.

Genetic networks of 51 genes in yeast that are SSL to *SGS1* or *RAD27* were of interest. *SGS1* (*RAD27*) has homologs in human cells, including *WRN*, *BLM* and *RECQL* (*FEN1* and *ERCC5*) genes. Mutations in these genes lead to cancer-pre-disposition syndromes, symptoms resembling premature aging, and Cockayne syndrome (Tong et al., 2001 and NCBI OMIM database). The regression approach was applied to these 51 genes to infer their TC interactions using the alpha data set in Spellman et al. (1998). There were 544 quartets ($\hat{\beta}_0, \hat{\beta}_1, \hat{\beta}_2, \hat{\beta}_3$) that satisfied the criteria

$$R^2 > 0.7 \text{ and } p - \text{values of } \hat{\beta}_0, \hat{\beta}_1, \hat{\beta}_2 \text{ and } \hat{\beta}_3 < 0.2. \quad (1.6)$$

Upon partitioning the 544 quartets into $11 \times 16 \times 25 \times 18$ cubes according to Silverman's rule, Shieh and her colleagues obtained the following surface:

$$T_i(t+1) = 0.38 + 0.51A_i(t) - 0.85R_i(t) + 0.80A_i(t)R_i(t). \quad (1.7)$$

From the top ten triplets, a small network was reconstructed. Some TC and TD interactions were consistent with the published literature, so the prediction was promising. To infer novel TC interactions, the *Score* was relaxed to 0.30 and 83 triplets were included, which resulted in a larger network. Among those predictions, some TC and TD interactions of interest were checked by RT-PCR experiments; the prediction accuracy including both layers was about 71 %. Note that the mode surface is determined by the majority of triplets, so this regression approach can be applied to any data set. Figure 1.8 confirms not only the validity of taking a data-driven approach, but also the development of the mode surface.

1.3.5

A Pattern Recognition Approach

Similarly to Sect. 3.4, transcriptional compensation (TC) interactions among a group of 51 yeast genes which are synthetic sick or lethal to *SGS1* or *RAD27* (Tong et al., 2001) are also of interest. Chuang et al. (2005) proposed a pattern recognition approach to infer TC interactions. The proposed approach, in fact, was implemented on indirect interactions among RT-PCR-confirmed TC and transcriptional diminishment (TD) gene pairs. For ease of description, in this section we utilize A, R and T, which are involved in direct interactions, to denote the genes involved in TD and TC interactions. Among RT-PCR-confirmed gene pairs, when the time course microarray gene expression (Spellman et al., 1998) of a target gene T was plotted lagged -1 in time behind that of A or R, the AT gene pairs generally exhibited similar patterns across time, as depicted in the blue zone of Fig. 1.9. On the other hand, RT gene pairs showed complementary patterns across time, as illustrated in Fig. 1.10. This is consistent with the ideal patterns for RT and AT, in which the expression of R and that of its corresponding lagged -1 T are mirror images, whereas those of A and lagged -1 T are similar. Furthermore, the areas enclosed by the RT curves are much larger than those enclosed by the AT curves, and this phenomenon was displayed

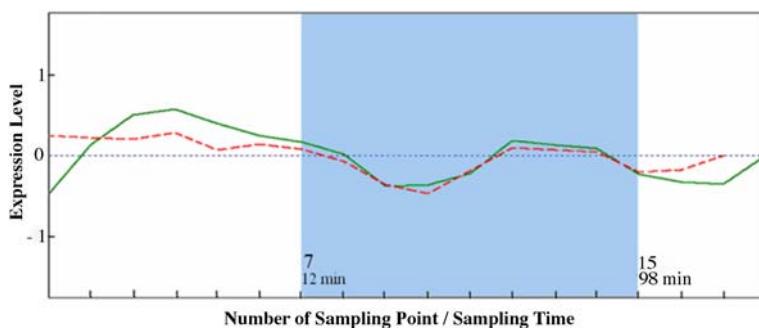


Figure 1.9. The gene expression pattern of Activator (solid line)–Target (dashed line) genes over time

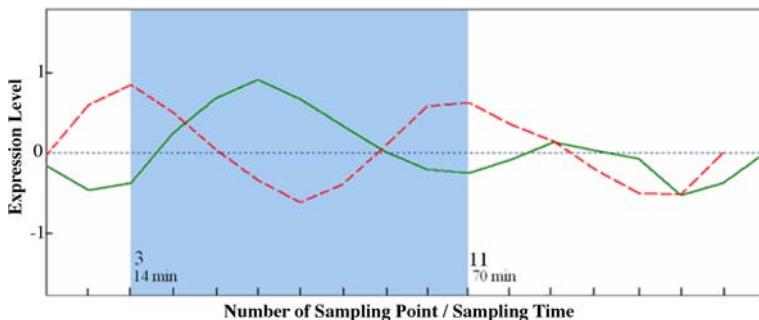


Figure 1.10. The gene expression pattern of Repressor (solid line)–Target (dashed line) genes over time

by many quantitative RT-PCT-confirmed gene pairs. This motivated Chuang and his colleagues to develop a pattern-recognition method that extracts the features of time course microarray data from confirmed gene pairs; the method was then applied to predict similar interactions among genes of interest. Specifically, they generalized the snake energy model (SEM) (Kass et al., 1988) by incorporating the particle swarm optimization (PSO) algorithm and a criterion including the areas enclosed by the AT and RT curves. Moreover, a cost function was integrated into the SEM to improve its discrimination power. This method is called the extended snake energy model (eSEM).

Typical indicators of a compensatory relationship are redundant genes (paralogs), redundant pathways, and synthetic sick or lethal (SSL) interactions (Wong and Roth, 2005). Among the 51 yeast genes of interest, those 17 genes whose TC and TD interactions were confirmed by RT-PCR experiments were focused upon (see the appendix of Chuang et al. (2005) for details). eSEM was applied to the cDNA microarray data in Spellman et al. (1998) to infer TC and TD interactions. For the alpha data set, experiment and control groups were mRNAs extracted from yeast cultures which were synchronized by alpha factor arrest and unsynchronized, respectively; there are 18 time points with no replicates. A full description and complete data sets are available at <http://cellcycle-www.stanford.edu>. The red (R) and green (G) fluorescence intensities were measured from the mRNA abundance in the experiment group and

Table 1.1. The prediction results of eSEM with *PSO_Set1* applied to the alpha data set with $(T_s, T_e) = (4, 12)$

Type	Score func.			<i>TH</i>	<i>Cutoff</i>	Num. of predicted gene pairs	Training set		Test set accuracy
		α	β				Accuracy		
TD	E_{Cost1}	2.8 (Std=0.2)	0.8 (Std=0.1)	0.1	1.8 (Std=0.1)	10	30 %	74 %	70 %
TC	E_{Cost2}	1.0 (Std=0.1)	1.1 (Std=0.1)	0.6	0.9 (Std=0.2)	8	25 %	67 %	70 %
TC						13	92 %		

control group, respectively. Log ratios of R to G were used to reconstruct the genetic interactions.

Among the RT-PCR-confirmed gene pairs, two-thirds (randomly chosen) were used as the training set to tune the parameters of eSEM. Leave-one-out cross-validation was performed in the training set to give the averaged accuracy, which was used to check against the prediction accuracy in order to detect any overfitting problem. Finally, the trained eSEM was applied to the test set (37 pairs) to yield the prediction accuracy. eSEM was first applied to one cell cycle of data, consisting of the fourth to the twelfth time points, which correspond to the first and second peaks of the two sine waves for genes exhibiting sinusoidal patterns. The prediction results of eSEM with the first set of the PSO cost function are summarized in Table 1.1. The prediction accuracy is defined to be the ratio of the number of correctly predicted pairs to the total number of gene pairs in the test set (37). Checked against the RT-PCR results, the prediction accuracy of eSEM with the first set of the PSO cost function is about 70 %. Note that if we consider only the 25 TC interactions in the test set, prediction accuracies range from 92 % to 100 %.

Next, the performance of eSEM was evaluated for the entire time course (18 time points) of the alpha data set. Table 1.2 summarizes the results of eSEM with the first

Table 1.2. The prediction results of eSEM with *PSO_Set1* applied to the alpha data set with $(T_s, T_e) = (1, 18)$

Type	Score func.			<i>TH</i>	<i>Cutoff</i>	Num. of predicted gene pairs	Training set		Test set accuracy
		α	β				Accuracy		
TD	E_{Cost1}	2.8 (Std=0.2)	0.8 (Std=0.1)	0.1	6.6 (Std=0.1)	11	46 %	72 %	73 %
TC	E_{Cost2}	1.0 (Std=0.1)	1.0 (Std=0.1)	0.6	1.7 (Std=0.1)	7	29 %	72 %	68 %
TC						11	100 %		

set of the PSO cost function. Checked against RT-PCRs, the prediction accuracy of eSEM ranges from 73 % to 68 %. Overall, the prediction accuracies of eSEM when applied to the alpha data set in Spellman et al. (1998) were as high as 76 % (about 70 % on average). Figures 1.9 and 1.10 motivated the incorporation of the areas enclosed by the RT curves versus those enclosed by the AT curves into eSEM, which improved the aforementioned prediction accuracy to 91 % for the alpha data set. This example demonstrates how visualization aids the development of methodology.

Acknowledgement. The authors wish to thank Drs. Chung-Ming Chen and Ming-Yen Cheng for constructive discussions. This research was supported in part by the National Science Council grant 93-2118-M001-028 and Academia Sinica Biotechnology development grant 23-33 to G.S.S.

References

- Aburatani, S., Saito, S., Toh, S. and Horimoto, K. (2006) A graphical chain model for inferring regulatory systems network from gene expression profiles. *Statistical Methodology*, 3:17–28.
- Akutsua, T., Kuhava, S., Maruyamac, O. and Miyano, S. (2003). Identification of genetic networks by strategic gene disruptions and gene overexpressions under a Boolean model. *Theoretical Computer Science*, 298:235–251.
- Arkin, A.P. and Ross, J. (1995). Statistical construction of chemical-reaction mechanisms from measured time-series. *J. Phys. Chem.*, 99:970–979.
- Arkin, A.P., Shen, P.D. and Ross, J. (1997). A Test Case of A Correlation Metric Construction of A Reaction Pathway from Measurements. *Science*, 277:1275–1279.
- Bar-Joseph, Z., Gerber, G., Gifford, D.K., Jaakkola, T.S. and Simon, I. (2002). A new approach to analyzing gene expression time series data. In Myer, G., Hannenhalli, S., Sankoff, D., Istrail, S., Pevzner, P. and Waterman, M. (eds) *Proceedings of the 6th annual international conference on Computational Biology*. ACM, New York, pp. 39–48.
- Barabási, A.-L. and Oltvai, Z.N. (2004). Network Biology: Understanding the cell's functional organization. *Nature Reviews*, 5:101–113.
- Beal, M.J., Falciani, F., Ghahramani, Z., Rangel, C. and Wild, D.L. (2005). A Bayesian approach to reconstructing genetic regulatory networks with hidden factors. *Bioinformatics*, 21:349–356.
- Cho, R.J., Campbell, H.J., Winzeler, E.A., Steinmetz, L., Conway, A., Wodicka, L., Wolfsberg, T.G., Gabrielian, A.E., Lansman, D., Lockhart, D.J. and Davis, R.W. (1998). A genome-wide transcriptional analysis of the mitotic cell cycle. *Molecular Cell*, 2:65–73.
- Chuang, C.L., Chen, C.M. and Shieh, G.S. (2005). *A pattern recognition approach to infer genetic networks*. Technical Report C2005-05, Institute of Statistical Science, Academia Sinica, Taiwan.
- Dillon, W.R. and Goldstein, M. (1984). *Multivariate analysis*. Wiley, New York.
- De Jong, H. (2002). Modeling and Simulation of Genetic Regulatory Systems: a literature review. *Journal of Computational Biology*, 9:67–103.

- Eisen, M.B., Spellman, P.T., Brown, P.O. and Botstein, D. (1998). Cluster analysis and display of genome-wide expression patterns. *Proceedings of National Academy of Sciences USA*, 95:14863–14868.
- Friedman, N., Linial, M., Nachman, I. and Pe'er, D. (2000). Using Bayesian networks to analyze expression data. *Journal of Computational Biology*, 7:601–620.
- Gu, C. (2002). *Smoothing Spline ANOVA Models*. Springer, New York.
- Härdle, W.H., Müller, M., Sperlich, S. and Werwatz, A. (2004). *Nonparametric and Semiparametric Modeling*. Springer, New York.
- Kafri, R.A., Rar-Even and Pilpel, Y. (2005). Transcription control reprogramming in genetic backup circuits. *Nat. Genet.*, 37:295–299.
- Kass, M., Witkin, A. and Terzopoulos, D. (1988). Snake: Snake energy models. *Int. J. Comput. Vision*, 1(4):321–331.
- Lesage, G., Sdicu, A.M., Manard, P., Shapir, J., Hussein, S. and Pilpel, Y. (2004). Analysis of beta-1,3-glucan assembly in *Saccharomyces cerevisiae* using a synthetic interaction network and altered sensitivity to caspofungin. *Genetics*, 167:35–49.
- Liang, S., Fuhrman, S. and Somogyi, R. (1998). REVEAL, a general reverse engineering algorithm for inference of genetic network architectures. *Pacific Symp. Biocomput.*, 3:18–29.
- Schäfer, J. and Strimmer, K. (2005). An empirical Bayes approach to inferring large-scale gene association networks. *Bioinformatics*, 21:754–764.
- Schmitt Jr, W.A. and Stephanopoulos, G. (2003). Prescription of transcriptional profiles of *Synechosystis PCC6803* by dynamic autoregressive modeling of DNA microarray data. *Biotechnol Bioeng*, 84:855–863.
- Schmitt Jr, W.A., Raab, R.M. and Stephanopoulos, G. (2004). Elucidation of Gene Interaction Networks Through Time-lagged Correlation Analysis of Transcriptional Data. *Genome Research*, 14:1654–1663.
- Scott, D.W. (1992). *Multivariate Density Estimation*. Wiley, New York.
- Shieh, G.S., Jiang, Y.C., Wang, T.F. and Hung, Y.C. (2004). A regression approach to reconstructing gene networks. In: *Proceedings of 2004 Taipei Symposium on Statistical Genomics*, 15–17 Dec 2004, Taipei, Taiwan, pp. 357–370.
- Shieh, G.S., Chen, C.M., Yu, C.Y., Huang, J. and Wang, W.F. (2005). A stepwise structural equation modeling algorithm to reconstruct genetic networks. Technical Report C2005-04, Institute of Statistical Science, Academia Sinica, Taiwan.
- Spellman, P.T., Sherlock, G., Zhang, M.Q., Iyer, V.R., Anders, K., Eisen, M.B., Brown, P.O., Botstein, D. and Futcher, B. (1998). Comprehensive identification of cell cycle-regulated genes of the yeast *Saccharomyces cerevisiae* by microarray hybridization. *Mol. Biol. Cell*, 9:3273–3297.
- Toh, H. and Horimoto, K. (2002). Inference of a genetic network by a combined approach of cluster analysis and graphical Gaussian modeling. *Bioinformatics*, 18:287–297.
- Tong, A.H. et al. (2001). Systematic genetic analysis with ordered arrays of yeast deletion mutants. *Science*, 294:2364–2366.
- Tong, A.H. et al. (2004). Global mapping of the yeast genetic interaction network. *Science*, 303:808–813.

- van Someren, E.P., Wessels, L.F.A. and Reinders, M.J.T. (2002). Genetic network modeling. *Pharmacogenomics*, 3:507–525.
- Wu, C.F.J. and Hamada, M. (2000). *Experiments: Planning, Analysis, and Parameter Design Optimization*. Wiley, New York.
- Wong, S.L. and Roth, F.P. (2005). Transcriptional compensation for gene loss plays a minor role in maintaining genetic robustness in *Saccharomyces cerevisiae*. *Genetics*, 171(2):829–833.
- Woolf, P.J. and Wang, Y. (2002). A fuzzy logic approach to analyzing gene expression data. *Physiological Genomics*, 3: 9–15.
- Xie, J. and Bentler, P.M. (2003). Covariance structure models for gene expression microarray data. *Structural Equation Modeling*, 10:566–582.
- Xu, H., Wu, P., Wu, C.F.J., Tidwell, C. and Wang, Y. (2002). A smooth response surface algorithm for constructing gene regulatory network. *Physiological Genomics*, 11:11–20.
- Yang, Y.H., Dudoit, S., Luu, P. and Speed, T.P. (2001). Normalization for cDNA microarray data. In: Bittner, M.L., Chen, Y., Dorsel, A.N., Dougherty, E.R. (eds) *Microarrays: Optical Technologies and Informatics* (Proc. SPIE vol. 4266). SPIE, Bellingham, WA.
- Younger, M.S. (1979). *Handbook for Linear Regression*. Duxbury, North Scituate, MA.

Reconstruction, Visualization and Analysis of Medical Images

IV.2

Henry Horng-Shing Lu

2.1	<i>Introduction</i>	814
2.2	<i>PET Images</i>	815
2.3	<i>Ultrasound Images</i>	819
2.4	<i>Magnetic Resonance Images</i>	822
2.5	<i>Conclusion and Discussion</i>	826

Advances in medical imaging systems have made significant contributions to medical diagnoses and treatments by providing anatomic and functional information about human bodies that is difficult to obtain without these techniques. These modalities also generate large quantities of noisy data that need modern techniques of computational statistics for image reconstruction, visualization and analysis. This article will report recent research in this area and suggest challenges that will need to be addressed by future studies. Specifically, I will discuss computational statistics for positron emission tomography, ultrasound images and magnetic resonance images from the perspectives of image reconstruction, image segmentation and vision model-based image analysis.

2.1

Introduction

It has been a boon to researchers from many scientific fields to be able to use data visualization to “see the unseen.” It is rather amazing that researchers can see through the human body and accurately visualize brain function using cutting edge medical imaging techniques – something that would have been very difficult to imagine at the beginning of the twentieth century (Kevles, 1997). Indeed, modern medical image modalities have made significant contributions to the understanding, diagnosis and treatment of biological activities and diseases inside the human body; these contributions are introduced and reviewed in most books on medical imaging (Suetens, 2002; Prince and Links, 2005). As techniques for capturing medical images continue to advance, it has become more of a challenge to visualize and analyze them, because they inherently contain a massive amount of noisy data. Modern techniques in computational statistics are crucial to extracting useful information from the various classes of modern medical images. Here, I discuss computational statistical methods for studying medical images, including images captured by positron emission tomography (PET), ultrasound images, and magnetic resonance images (MRI), from the perspectives of image reconstruction, image segmentation, and vision model-based image analysis.

Computerized tomography (CT) is an important technique for obtaining accurate information about the interior of a human body based on observations detected outside the body. The precise reconstruction of images of the interior of the human body from this data is a challenge suited to computational statistics. As the detected observations are indirectly related to the target image, the tomographic reconstruction problem is an inverse problem, which is often ill-posed or ill-conditioned. Due to the nature of ill-posedness, the reconstruction of, for example, positron emission tomography (PET) images by maximum likelihood estimation with the EM algorithm (MLE-EM) (Shepp and Vardi, 1982; Vardi et al., 1985) weighted least square estimation (WLSE), and other methods without regularization, will produce images with edge and noise artifacts (Fessler, 1994; Ouyang et al., 1994). Thus, computational statistical techniques must be used to integrate and fuse the correlated but incomplete structure information with other medical modalities, like X-ray CT, magnetic

resonance imaging, and others (Lu et al., 1998; Lu and Tseng, 1997; Tu et al., 2001). The reconstruction problems are even more challenging for the types of molecular imaging associated with genomic and medical studies in the post-genomic era, such as microPET, which requires further research to provide more accurate images at molecular levels of resolution.

Image synthesis and segmentation by spatial-frequential analysis were developed to model human vision; see Malik and Perona (1990); Jain and Farrokhnia (1991); Dunn et al. (1994); Tan (1995); Zhu et al. (1998). These studies reveal that the descriptors of texture images can be represented by the outputs from multichannel Gabor filters, which are constructed to detect the responses at different orientations and frequencies. This vision model can be applied to segment medical images using the techniques of image processing and computational statistics. In particular, ultrasound images are noisy due to inherent speckle noise. Vision model-based segmentation methods for ultrasound images were developed from active contour models (i.e., the snake-balloon model), region competition, and related methods (Chen et al., 2000; Chen and Lu, 2001; Chen et al., 2001a,b, 2002, 2003). In addition, the technique of sliced inverse regression (Li, 1991) was extended to segment dynamic images, including magnetic resonance images (Wu and Lu, 2004). Further developments and challenges associated with the visualization and analysis of dynamic images in 3-D are also discussed in this chapter.

PET Images

2.2

The scanning, acquisition and reconstruction process of a PET system is displayed in Fig. 2.1. An object with a radioactive chemical tracer is injected into the body in order to detect the biochemical activity inside it; detectors outside the body monitor the radiation emitted by the tracer. This type of imaging is described in more depth in various books on medical imaging (Suetens, 2002; Prince and Links, 2005).

An ideal model for PET images was introduced in Shepp and Vardi (1982); Vardi et al. (1985). Initially, positrons resulting from biochemical activity are emitted from inside the body. These positrons hit nearby electrons and annihilate themselves. When a positron hits an electron, a pair of photons are generated and they travel in opposite directions. This photon pair is detected by a pair of scintillation detectors outside the body. Before these pairs of photons arrive at the detectors, their energy is attenuated by Compton scattering as they pass through tissue. Therefore, some of these photon pairs do not retain a detectable level of energy, and remain undetected. The other attenuated photon pairs are recorded, and can be described by a survival probability (Politte and Snyder, 1991). The detectors may also receive stray photon pairs that are not generated by the annihilations that occur at the target image in a specific slice of the body. That is, there are accidental (or random) coincidence (AC or RC) events. The observations made by a pair of detectors are the sum of the photon pairs that occur at the target image and the AC events. Therefore, to recover the true

PET System

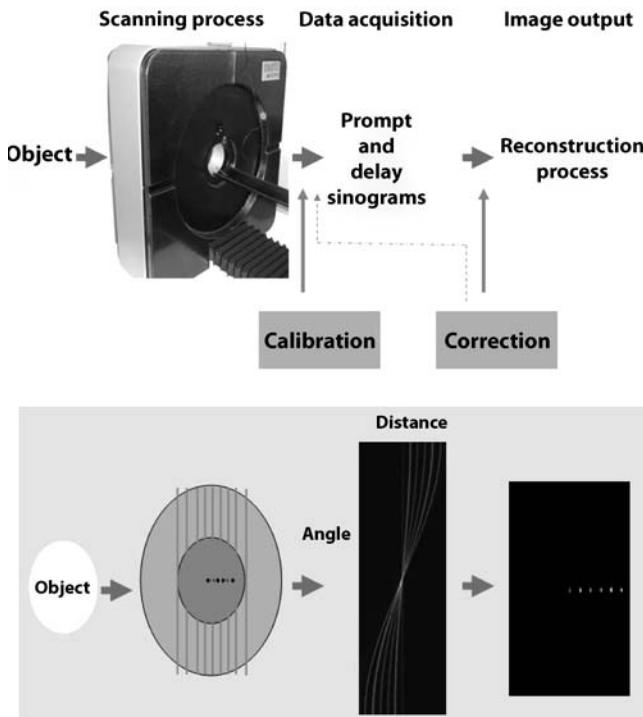


Figure 2.1. The scanning, acquisition and reconstruction process of a PET system

image, one can subtract the number of photon pairs resulting from the presence of AC events (Politte and Snyder, 1991; Fessler, 1994).

Suppose the target image is partitioned into B boxes (or pixels in 2-D or voxels in 3-D), and there are D detector tubes formed from pairs of detectors positioned outside the body. For each pixel, $b = 1, 2, \dots, B$, and each tube, $d = 1, 2, \dots, D$, the following notations are adopted:

- $\lambda(b)$: the emission intensity of the target at box b
- $q(b, d)$: the probability of an emission from box b being detected along detector tube d
- $s(b, d)$: the survival probability that a photon pair emitted from box b and traveling in tube d will have an energy that is greater than the threshold level of the detectors after attenuation.

With attenuation, the transition probability becomes

$$p(b, d) = q(b, d)s(b, d). \quad (2.1)$$

In order to correct for AC (or RC) events, the recent generation of commercial PET systems use prompt (or real-time) window coincidence to subtract the random coincidences from delay windows (Spinks et al., 1988; Fessler, 1994). The following notation is used:

- $n_p^*(d)$: the number of coincident photon pairs collected in the prompt windows for detector tube d
- $n_d^*(d)$: the number of coincident photon pairs collected in the delay windows for detector tube d .

Furthermore, $n_p^*(d)$ and $n_d^*(d)$ are assumed to be statistically independent and Poisson-distributed with different means:

$$n_p^*(d) \sim \text{Poisson}(\lambda^*(d) + \lambda_d^*(d)), \quad (2.2)$$

$$n_d^*(d) \sim \text{Poisson}(\lambda_d^*(d)), \quad (2.3)$$

where “ \sim ” means that the random variable “is distributed as,” $\lambda_d^*(d)$ is the mean intensity of $n_d^*(d)$, and

$$\lambda^*(d) = \sum_b p(b, d) \lambda(b). \quad (2.4)$$

One can define the precorrecting value in detector tube d , $n^*(d)$ by subtracting $n_d^*(d)$ from $n_p^*(d)$:

$$n^*(d) = n_p^*(d) - n_d^*(d). \quad (2.5)$$

However, $n^*(d)$ can take a negative value. Also, the mean and variance of $n^*(d)$ are different when $\lambda_d^*(d) > 0$. Therefore, $n^*(d)$ is not Poisson-distributed. When $\lambda^*(d) = \lambda_d^*(d) = 0$, $n^*(d) = 0$ with probability 1. Otherwise, by approximating the moment-generating function in the neighborhood of 0, $n^*(d)$ is approximately distributed as a normal distribution with mean $\lambda^*(d)$ and variance $\lambda^*(d) + 2\lambda_d^*(d)$. Therefore, the maximum likelihood estimate of this approximate model turns out to be the weighted least square estimate (WLSE).

WLSE methods can be achieved by applying finite series expansion reconstruction methods (Censor, 1983) like the algebraic reconstruction technique (ART) (Herman, 1980; Herman et al., 1980). The idea of ART is to solve the system of equations successively, but not simultaneously. Because it is an iterative method based on row operations, it is efficient in terms of computation time and storage requirements. Range limit constraints, such as nonnegativeness, can be easily implemented during every iteration as well.

Due to the ill-posedness of this inverse problem, the reconstruction procedure needs to be regularized. One way to regularize the WLSE is to combine the (weighted) least square term and a penalty term into a functional object to be optimized. In a Bayesian framework, the penalty term is related to the prior distribution. Various methods of Bayesian estimation with Markov chain Monte Carlo (MCMC) methods have been proposed in the literature. For example, it was proposed that the local

smoothness information should be incorporated in order to get better reconstructed PET images (Ouyang et al., 1994). This was accomplished by combining the prior boundary information with Gibbs sampling in the Bayesian reconstruction. However, the choices of the neighborhood system, the parameters in the Gibbs prior, and the weighting factors in the weighted line sites are difficult to determine, and their computational costs are considerable.

Therefore, this author and his colleagues proposed a more efficient method that used ART to combine the local smoothness information, to get better PET reconstruction in the presence of AC events and attenuation (termed the cross-reference weighted least square estimate, CRWLSE), with the algebraic reconstruction technique (ART) (Lu et al., 1998). First, the WLSE with ART algorithm is applied to get an initial reconstruction. Second, the boundaries and the WLSE are used to retrieve a mean estimate. Finally, a penalized WLSE incorporates the boundary information using the ART algorithm. The computational complexity is only linear with respect to the sizes of the pixels and detector tubes. The range limits and spatially variant penalty are easily incorporated without compromising computational efficiency. The reconstruction was quite successful at reducing the noise and edge effects, as reported in Lu et al. (1998).

The author and colleagues also proposed a cross-reference MLE (CRMLE) with a modified EM algorithm for PET with or without AC events (Lu and Tseng, 1997; Tu et al., 2001). These efficient methods equip the MLE with the related but incomplete boundary information and only one penalty parameter. These forms of penalized MLE have penalty terms that are derived from the boundary information of related medical modalities. Several speedy approaches can be used that apply the boundary information during reconstruction. The solution, the accelerated cross-reference maximum likelihood estimate (ACRMLE), is unique. New algorithms adapted from the expectation/conditional maximization (ECM) algorithm in Meng and Rubin (1993) as well as the space-alternating generalized expectation maximization (SAGE) algorithms in Fessler and Hero (1994, 1995) allow the computational complexity to remain linear and the range limits to be preserved. These algorithms are convergent, and even faster convergence speeds are achieved by using the modified SAGE algorithm first, followed by the modified ECM algorithm. The penalty parameter can be selected by users or data-driven methods. The penalty parameter can be quickly determined automatically using generalized approximate cross-validation (GACV) (Xiang and Wahba, 1996).

Currently, the high spatial resolution and sensitivity of microPET make it an ideal modality for *in vivo* molecular and genetic imaging in functional genomic studies. At this stage, it can be used to detect the effects of gene therapy inside animals. High-quality image reconstruction is important for establishing a solid basis for the quantitative study of microPET images (Chatzilooannou et al., 2000). Maximum likelihood estimation with the expectation maximum algorithm (MLE-EM) permits the reconstruction of microPET images with random correction using joint prompt and delay sinograms (PDEM) (Chen et al., 2007). The joint Poisson model has the advantages of preserving Poisson properties without increasing the variance caused by random correction. The stopping criterion for PDEM is determined by K-fold cross-validation.

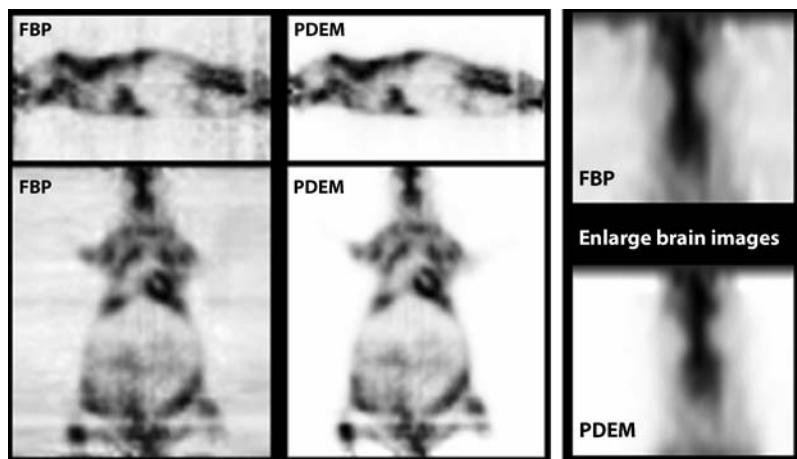


Figure 2.2. Coronal and sagittal images of a mouse reconstructed by PDEM (right) and FBP (left). The images reconstructed by PDEM have less noise than those reconstructed by FBP. The enlarged brain image reconstructed by PDEM has clearer boundaries than that reconstructed by FBP

The signal-to-noise ratio (SNR) is used to compare the quality of the reconstructed microPET images of the experimental phantom obtained by filtered backprojection (FBP) and PDEM methods. The results reveal that PDEM offers higher image quality than FBP does. A comparison of these two methods for the microPET reconstruction of a real mouse is shown in Fig. 2.2. More studies and comparisons are discussed in Chen et al. (2007).

Ultrasound Images

2.3

The noninvasive, real-time, convenient and economical properties of ultrasound images have resulted in their widespread clinical application to the early detection of several diseases and routine monitoring in clinical practice. However, it is difficult to segment the regions of tumors, for example, because of the low signal-to-noise ratio of speckle noises in ultrasound images (Burckhardt, 1978; Goodman, 1985). There are several possible computational statistical methods that can be used to address these issues during the analysis of ultrasound images.

An ultrasound image can be regarded as a type of texture image, and therefore the segmentation methods of texture images can be considered. Inspired by recent studies of the early vision system and human V1 cells, image synthesis and the segmentation of static images based on a spatial-frequential analysis that mimics human vision were investigated (Malik and Perona, 1990; Jain and Farrokhnia, 1991; Dunn et al., 1994; Tan, 1995; Zhu et al., 1998). Recent progress has been made in modeling textures and dynamic textures using statistical models (Wu et al., 2000; Zhu et al.,

2000; Soatto et al., 2001; Wu et al., 2002). Methods for extracting features from static and dynamic images in 2-D can be applied to ultrasound and MRI images, and these are discussed in Sects. 2.3 and 2.4 (Wu and Lu, 2004). These methods can also be extended to 3-D or higher dimensions.

Space Domain: Local Blocks

Let us denote a rectangular lattice containing a 2-D digital image of size $N \times M$ by

$$\mathcal{S} = \{(I, J) | 1 \leq I \leq N, 1 \leq J \leq M, I, J \in \mathbb{Z}\}.$$

The spatial characteristics of a pixel in the image are described by its neighboring pixels. Therefore, a local block in one time frame of size $b \times b$ can be created as a feature vector $\mathbf{x}^{(i)}(t_j)$ of the central pixel $i \in \mathcal{S}$, $i = 1, \dots, n$ and $n = (M - b + 1)(N - b + 1)$ for each time point t_j , $j = 1, \dots, m$. The dimension of the feature vector in the space domain is b^2 . Pixels at different time frames are not included in the feature vector because they may vary according to the motion. For the same pixel i , the collection of feature vectors along the sequence of images $\{\mathbf{x}^{(i)}(t_j), j = 1, \dots, m\}$ represents the temporal variation of the features due to motion over time.

Given a sequence of m training images, the class labels $\{y^{(i)}(t_j), j = 1, \dots, m\}$, and the feature vectors $\{\mathbf{x}^{(i)}(t_j), j = 1, \dots, m\}$, the projection directions of feature vectors for classification and prediction can be found. If the number of training images, m , is bigger than the dimension of the feature vectors, b^2 , then it is feasible to estimate the projection direction in the dimension of b^2 . However, for a short sequence of training images, m is not necessarily bigger than b^2 . Therefore, information from neighborhood pixels must be borrowed.

Let $\mathcal{N}_q^{(i)}$ be the set of neighboring sites for pixel i in a q -order neighborhood system. For example, a first-order neighborhood system is a 4-neighborhood system, since each interior site has four neighbors and the size of $\mathcal{N}_1^{(i)}$ is 5. There are eight neighbors for every interior site in a second-order neighborhood system, which is also called an 8-neighborhood system, and the size of $\mathcal{N}_2^{(i)}$ is 9. Then, for pixel i , the neighboring feature vectors can be collected as the training set:

$$\mathcal{X}^{(i)} = \left\{ \mathbf{x}_l^{(i)}(t_j), j = 1, \dots, m, l \in \mathcal{N}_q^{(i)} \right\}, \quad (2.6)$$

where $i = 1, \dots, n$ and $n = (M - b + 1)(N - b + 1)$. For instance, if $q = 2$, then the size of the training set becomes $9m$, which is bigger than b^2 when $m > b^2/9$.

Frequency Domain: Fourier Transform of Local Blocks

If the features of an image are periodic over space, then the features in a local block in the space domain can be transformed to the frequency domain by a Fourier transform. This transform will highlight the periodic pattern (Weaver, 1983). This can be performed using a fast Fourier transform (FFT) if the block size is of the power 2. For

a block of size $b \times b$, a two-dimensional discrete Fourier transform can be expressed as

$$F(u, v) = \frac{1}{b^2} \sum_{x=0}^{b-1} \sum_{y=0}^{b-1} f(x, y) \exp \left[-i2\pi \left(\frac{ux}{b} + \frac{vy}{b} \right) \right], \quad (2.7)$$

where $i = \sqrt{-1}$, $u, v = 0, \dots, b - 1$. Because the image intensity is real-valued, the Fourier transform is symmetrical about the center. Because of this symmetry, almost half of the FFT calculation is redundant. Therefore, the feature in the frequency domain consists of $|F(u, v)|$ with dimension $b^2/2 + 2$ if b is of the power 2.

Space-Frequency Domain: Gabor Filter Banks of Local Blocks

Human vision has demonstrated its superior capacity to detect the boundaries of desired objects. The vision model based upon the previous work of this author and his collaborators is described in Chen et al. (2000, 2001a), although similar approaches can be applied too. A neuroimage or distance map is constructed by convolving the observed image with a bank of specific frequency and orientation bands, such as a bank of Gabor functions. The general form of a Gabor function is given by

$$\begin{aligned} g(x, y) &= \exp \left\{ -[(x - x_0)^2 a_1^2 + (y - y_0)^2 a_2^2] \pi \right\} \\ &\exp \left\{ -2\pi i [u_0(x - x_0) + v_0(y - y_0)] \right\}, \end{aligned} \quad (2.8)$$

and its Fourier transform is

$$\begin{aligned} G(u, v) &= \exp \left\{ -\frac{1}{\pi} \left[\frac{(u - u_0)^2}{a^2} + \frac{(v - v_0)^2}{b^2} \right] \right\} \\ &\exp \left\{ -2\pi i [x_0(u - u_0) + y_0(v - v_0)] \right\}. \end{aligned} \quad (2.9)$$

Each local block is convolved with a bank of Gabor filters with different orientations and frequencies. The so-called *G-vector* is employed as the feature vector at pixel (I, J) , which is computed by

$$G_V(I, J) = \{g_{pk}(I, J), g_{nk}(I, J); k = 1, \dots, r\}, \quad (2.10)$$

where $g_{pk}(I, J)$ and $g_{nk}(I, J)$ are the summations of the positive and negative values for the neuroimage that is the convoluted image with the k th Gabor filter. Thus, the dimension of the feature vector is $2r$ in the space-frequency domain. For instance, a bank of $r = 3 \times 8 = 24$ Gabor filters are designed with three scales of center frequencies, where $\sqrt{2}/2$, $\sqrt{2}$, and $b\sqrt{2}/4 = 2\sqrt{2}$ when $b = 8$, as well as eight orientations of angles, 0, 30, 60, 90, 120, 150, 180 and 210° .

On the other hand, statistical distributions can be used to model speckle noise in ultrasound images. Suppose the intensities are independently and identically distributed as Rayleigh and related distributions (Burckhardt, 1978; Goodman, 1985). These different types of features can be combined with active contour models (i.e.,

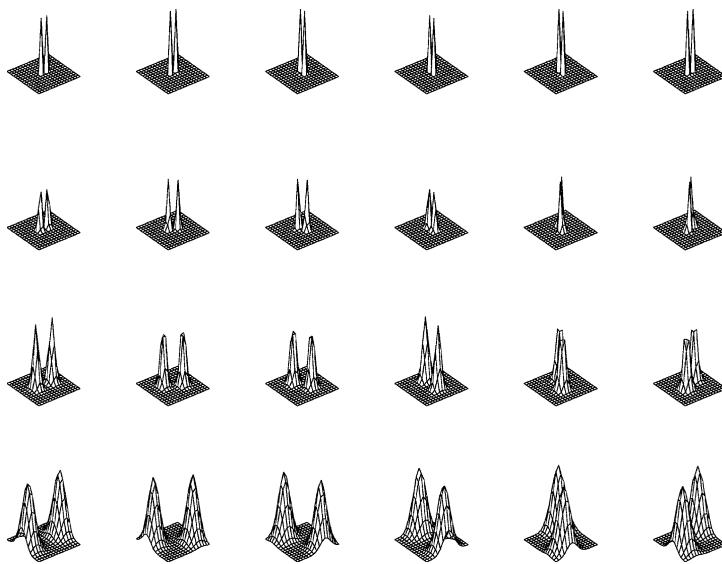


Figure 2.3. A rosette map consisting of 42 Gabor filters. The half-peak magnitude frequency bandwidth is set to one octave for each Gabor filter. Only the half-peak supports of the filters are shown in the map

snake-balloon models and their improvements), region growing/competition methods, and other segmentation methods in order to analyze ultrasound images (Chen et al., 2000; Chen and Lu, 2001; Chen et al., 2001a,b, 2002, 2003). For example, 24 Gabor filters can be considered in the space–frequency domain, as shown in Fig. 2.3. An ultrasound image is then transformed to a distance map by the Gabor filter banks, as illustrated in Fig. 2.4. The active contour model can be modified with the features in the distance map in order to obtain segmentation results, as demonstrated in Fig. 2.5. More studies are discussed in Chen et al. (2000). Further improvements are reported in Chen and Lu (2001); Chen et al. (2001a,b, 2002, 2003).

2.4 Magnetic Resonance Images

Magnetic resonance images (MRI) and functional MRI (fMRI) are important medical modalities for understanding human diseases and brain function. These modalities are discussed in depth in various books on medical images (Suetens, 2002; Prince and Links, 2005). A typical application of computational statistics to fMRI is the statistical parametric mapping (SPM, see <http://www.fil.ion.ucl.ac.uk/spm/>) system. In an attempt to advance the study of MRI and fMRI images, the author and his collaborators segmented dynamic images with just a few training images that extend the studies of spatial-frequential analysis to motion segmentation.

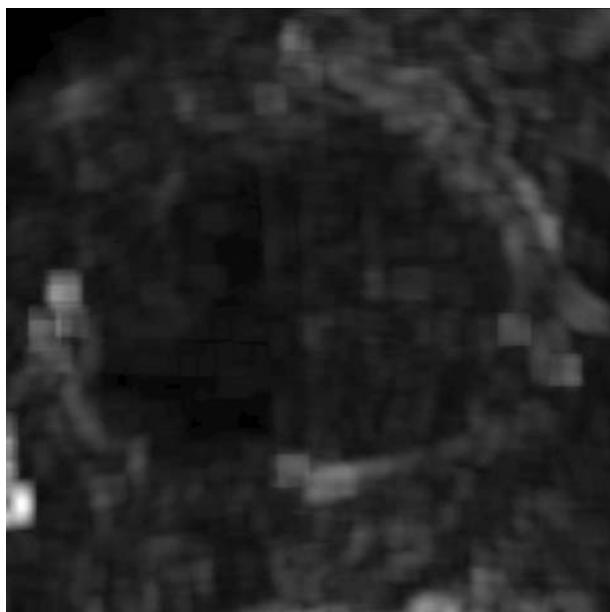


Figure 2.4. Derivation of the distance map. The underlying liver ultrasound image is decomposed into overlapping blocks of subimages. Each block is filtered with a set of Gabor functions to derive its G-vector. The distance map is formed from the G-vector lengths of all blocks

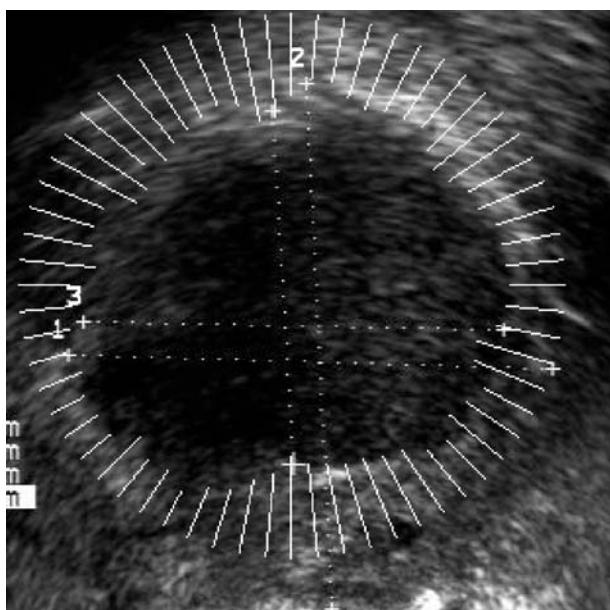


Figure 2.5. The boundaries derived by the proposed snake model are plotted; the initial contours and the derived boundaries are as indicated

Sliced inverse regression (SIR) was proposed in Li (1991) as a way to find compact representations that explore the intrinsic structure of high-dimensional observations. It has been extended and used in various applications (Chen and Li, 1998; Li, 2000; Chen and Li, 2001). This technique has been applied with spatial-frequent analysis in order to segment and diagnose static images, like ultrasound images in Sect. 2.3 (Chen et al., 2000; Chen and Lu, 2001; Chen et al., 2001a,b, 2002, 2003). We now consider the possibility of extending SIR and spatial-frequent analysis to dynamic images.

When extended to dynamic data, the SIR model is known as dynamic SIR (DSIR) (Wu and Lu, 2004). DSIR is combined with spatial-frequent analysis for motion segmentation. Every pixel in an image is regarded as a realization of a stochastic process over space and time. The feature vector for one pixel in one time frame is analyzed through the spatial-frequent analysis of local blocks centered at that pixel. Assuming that the relationship between these feature vectors and class labels remains similar between successive frames for neighboring pixels, then the intrinsic dimensions of feature vectors in the training images can be found by DSIR. These projected feature vectors thus provide prediction rules for forthcoming images in the test set. Only a small number of training images are needed to decide the projection of feature vectors and prediction rules.

The following model of SIR was introduced in Li (1991):

$$y = f(\beta'_1 \mathbf{x}, \dots, \beta'_K \mathbf{x}, \epsilon), \quad (2.11)$$

where y is a univariate variable, \mathbf{x} is a random vector with dimension $p \times 1$, $p \geq K$, the β 's are vectors with dimensions $p \times 1$, ϵ is a random variable that is independent of \mathbf{x} , and f is an arbitrary function. The β 's are referred to as the effective dimension reduction (*e.d.r.*) or projection directions. Sliced inverse regression (SIR) is a method used to estimate the *e.d.r.* directions based on y and \mathbf{x} . Under regular conditions, it is shown that the centered inverse regression curve $E[\mathbf{x}|y] - E[\mathbf{x}]$ occurs in the linear subspace spanned by $\beta_k \Sigma_{\mathbf{X}\mathbf{X}}$ ($k = 1, \dots, K$), where $\Sigma_{\mathbf{X}\mathbf{X}}$ denotes the covariance matrix of \mathbf{x} . Based on these facts, the estimated β 's can be obtained by standardizing \mathbf{x} , partitioning slices (or groups) according to the value of y , calculating the slice means of \mathbf{x} , and performing a principal component analysis of the slice means with weights.

The above model, (2.11), can be extended to dynamic data as follows:

$$y(t) = f(\beta'_1 \mathbf{x}(t), \dots, \beta'_K \mathbf{x}(t), \epsilon(t)), \quad (2.12)$$

where $y(t)$ and $\mathbf{x}(t)$ are response variables and p -dimensional covariates observed at time t . The projection directions, the β 's, are assumed to be invariant over time, and $\epsilon(t)$ is the stochastic process of noise. Analogous to the steps in Li (1991, 2000), the following conditions can be assumed, which prove the subsequent theorem (Wu and Lu, 2004).

Condition 1 For any b in R^p , $E[b' \mathbf{x}(t) | \beta'_1 \mathbf{x}(t), \dots, \beta'_K \mathbf{x}(t)]$ is linear in $\beta'_1 \mathbf{x}(t), \dots, \beta'_K \mathbf{x}(t)$ for any t . That is, $E[b' \mathbf{x}(t) | \beta'_1 \mathbf{x}(t), \dots, \beta'_K \mathbf{x}(t)] = c_0 + c_1 \beta'_1 \mathbf{x}(t) + \dots + c_K \beta'_K \mathbf{x}(t)$ for some constants c_0, c_1, \dots, c_K and any t .

Theorem 1 Under Model (2.12) and Condition 1, $E[\mathbf{x}(t)|y(t)] - E[\mathbf{x}(t)]$ falls within the linear subspace spanned by $\text{Cov}(\mathbf{x}(t))\beta_k$ for any $t, k = 1, \dots, K$.

When $\mathbf{x}(t)$ is elliptically symmetric for any time t , the above condition is fulfilled. This condition is weaker than the assumption of elliptical symmetry for $\mathbf{x}(t)$. When this condition is violated, the biases involved with estimating the projection directions are not large, as discussed in Li (1991, 2000). Because the model (2.12) does not consider any structural changes over time, successive data can be pulled together to estimate the projection directions more effectively, so long as the projection directions remain the same. Furthermore, all of the existing properties for SIR can be passed on to this dynamic model without any difficulty, so this model is called the dynamic SIR (DSIR) model.

The technique of MR imaging has become widely used as a diagnosis tool due to the high-quality soft tissue contrast it yields, its noninvasiveness and its functionality. It is necessary to identify components that correspond to different types of tissues and structures with computers automatically. For example, human brain tissues can be classified into three types by MR images: gray matter, white matter, and cerebrospinal fluid (CSF). The performance of the DSIR method when applied to a sequence of MR images of the epi- and endocardial surfaces of myocardium is a good example. Since the heart is an organ that exhibits motion, examining its image characteristics with a sequence of 2-D images leads to useful information about its physical condition. The goal in this experiment is to extract the boundary description for the inner and

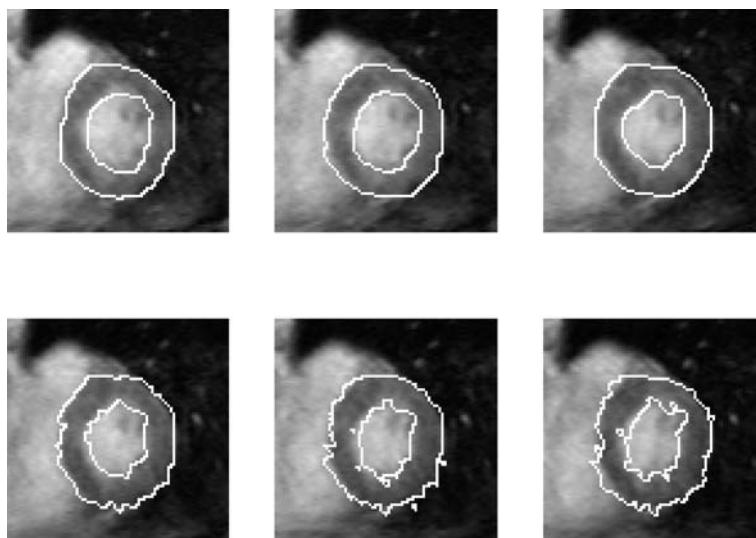


Figure 2.6. Segmentation results are displayed for a sequence of MR images of myocardium in the space domain. The top three images display classification results and the bottom three images show prediction results

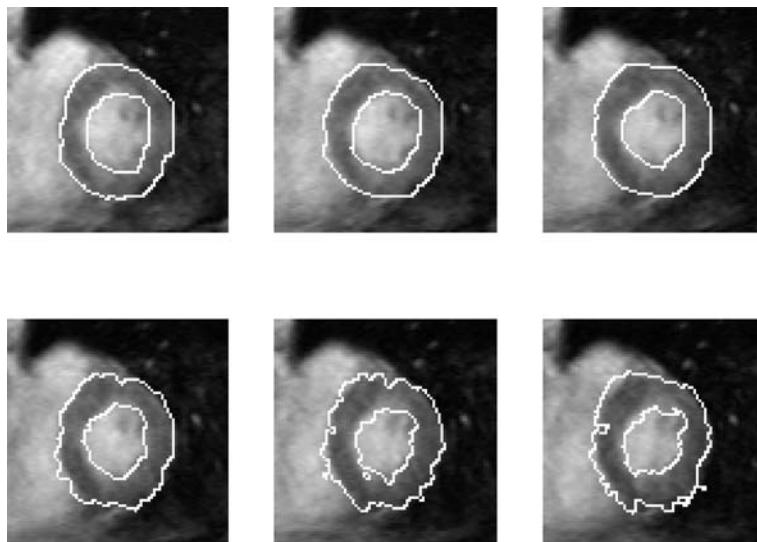


Figure 2.7. Segmentation results are displayed for a sequence of MR images of myocardium in the frequency domain. The top three images display classification results and the bottom three images show prediction results

Table 2.1. Classification error rates for a sequence of MR images where Obj represents the object, Bg denotes the background, and Total refers to the average error of the whole image

Feature	Frame 1			Frame 2			Frame 3		
	Obj	Bg	Total	Obj	Bg	Total	Obj	Bg	Total
Space	0.0132	0.0013	0.0034	0.0068	0.0027	0.0035	0.0032	0.0016	0.0019
FFT	0.0167	0.0023	0.0048	0.0188	0.0044	0.0070	0.0097	0.0033	0.0045
Gabor	0.0221	0.0032	0.0069	0.0154	0.0043	0.0066	0.0081	0.0040	0.0048

outer walls of the left ventricle endocardium. The training set can be obtained from the first three images by a medical expert. Using the proposed procedure, the segmentation results in the three feature domains shown in Figs. 2.6, 2.7, and 2.8. The classification error rates (derived based on the target boundaries drawn by a doctor) are reported in Tables 2.1 and 2.2. Thus, DSIR successfully performs the segmentation of this sequence of MR images. More studies are reported in Wu and Lu (2004).

Conclusion and Discussion

This article introduces and discusses several studies on the reconstruction, visualization and analysis of medical images. In particular, computational statistical methods are used as important tools in the analysis of PET, ultrasound and magnetic reso-

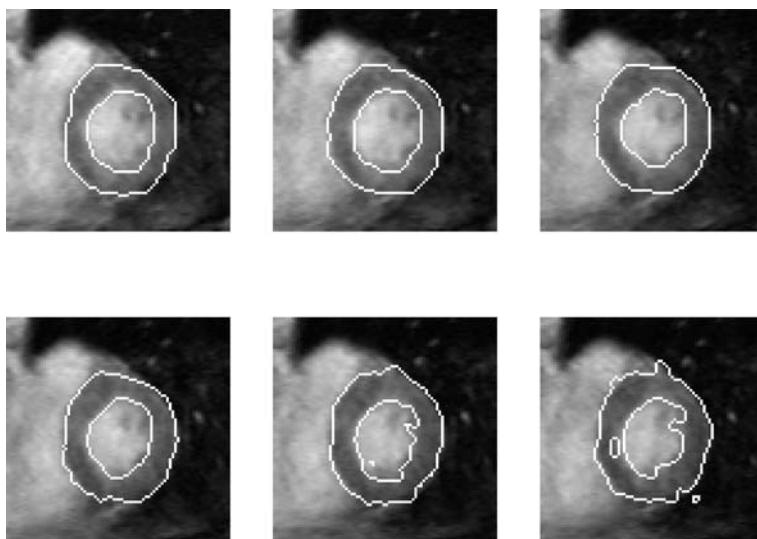


Figure 2.8. Segmentation results are displayed for a sequence of MR images of myocardium in the space–frequency domain. The top three images display classification results and the bottom three images show prediction results

Table 2.2. Prediction error rates for a sequence of MR images are reported, where Obj represents the object, Bg denotes the background, and Total refers to the average error of the whole image

Feature	Frame 4			Frame 5			Frame 6		
	Obj	Bg	Total	Obj	Bg	Total	Obj	Bg	Total
Space	0.0719	0.0216	0.0310	0.0630	0.0313	0.0370	0.0964	0.0319	0.0440
FFT	0.0832	0.0191	0.0312	0.0750	0.0302	0.0382	0.1264	0.0399	0.0562
Gabor	0.0816	0.0145	0.0289	0.0503	0.0337	0.0370	0.0899	0.0398	0.0505

nance images. These represent very interesting and challenging applications of computational statistics to medical and biological images in the scientific community.

Further studies of microPET, SPECT and other tomography systems along with the image fusion of other medical modalities will be challenging topics for the future development of computational statistics. Dynamic 2-D and 3-D images generated by medical imaging modalities, including different applications of tomography images, color Doppler ultrasound images, fMRI images, molecular images and so forth, will present special challenges to computational statistics.

Further successes in relation to the development of even more advanced medical imaging systems in the future can be expected from interdisciplinary collaborations between researchers in statistics, computation, biological and medical sciences. We will then be able to develop and apply state-of-art techniques in computational statis-

tics in order to reconstruct, visualize and analyze huge numbers of noisy biological and medical images with high accuracy and efficiency in the future.

Acknowledgement. The author would like to thank previous collaborators in this area, including Dr. Chung-Ming Chen, Jyh-Cheng Chen, Han-Ming Wu, Tai-Been Chen and others. He is also thankful for the comments from Dr. Josh Rest and the editors that subsequently helped him to improve this report.

References

- Burckhardt, C.B. (1978). Speckle in ultrasound B-mode scans. *IEEE Trans Ultrasonics*, 25:1–6.
- Censor, Y. (1983). Finite Series-Expansion Reconstruction Methods. *Proc IEEE*, 71:409–419.
- Chatzioannou, A., Qi, J., Moore, A., Annala, A., Nguyen, K., Leahy, R. and Cherry, S. (2000). Comparison of 3D MAP and filtered backprojection algorithms for high resolution animal imaging with microPET. *IEEE Trans Med Imaging*, 19:507–512.
- Chen, C.H. and Li, K.C. (1998). Can SIR be as popular as multiple linear regression? *Stat Sinica*, 8:289–316.
- Chen, C.H. and Li, K.C. (2001). Generalization of Fisher's linear discriminant analysis via the approach of sliced inverse regression. *J Korean Stat Soc*, 30:193–217.
- Chen, C.M., Lu, H.H.S. and Lin, Y.C. (2000). An Early Vision Based Snake Model for Ultrasound Image Segmentation. *Ultrasound Med Biol*, 26:273–285.
- Chen, C.M. and Lu, H.H.S. (2001). An Adaptive Snake Model for Ultrasound Image Segmentation: Modified Trimmed Mean Filter, Ramp Integration and Adaptive Weighting Parameters. *Ultrason Imag*, 22:214–236.
- Chen, C.M., Lu, H.H.S. and Han, K.C. (2001). A Textural Approach Based on Gabor Functions for Texture Edge Detection in Ultrasound Images. *Ultrasound Med Biol*, 27:513–534.
- Chen, C.M., Lu, H.H.S. and Hsiao, A.T. (2001). A Dual Snake Model of High Penetrability for Ultrasound Image Boundary Extraction. *Ultrasound Med Biol*, 27:1651–1665.
- Chen, C.M., Lu, H.H.S. and Huang, Y.S. (2002). Cell-Based Dual Snake Model: A New Approach to Extracting Highly Winding Boundaries in The Ultrasound Images. *Ultrasound Med Biol*, 28:1061–1073.
- Chen, C.M., Lu, H.H.S. and Chen, Y.L. (2003). A Discrete Region Competition Approach Incorporating Weak Edge Enhancement for Ultrasound Image Segmentation. *Pattern Recogn Lett*, 24:693–704.
- Chen, T.B., Chen, J.C., Lu, H.H.S. and Liu, R.S. (2007). MicroPET Reconstruction with Random Coincidence Correction via a Joint Poisson Model. *Medical Engineering & Physics*, (in press).
- Dunn, D., Higgins, W.E. and Wakeley, J. (1994). Texture segmentation using 2-D Gabor elementary functions. *IEEE Trans Pattern Anal Mach Intell*, 16:130–149.
- Fessler, J.A. (1994). Penalized Weighted Least-Squares Image Reconstruction for Positron Emission Tomography. *IEEE Trans Med Imag*, 13:292–300.

- Fessler, J.A. and Hero, A.O. (1994). Space-alternating generalized expectation maximization algorithm. *IEEE Trans Signal Proc*, 42:2664–2677.
- Fessler, J.A. and Hero, A.O. (1995). Penalized maximum-likelihood image reconstruction using space-alternating generalized expectation-maximization algorithms. *IEEE Trans Imag Process*, 4:1417–1429.
- Goodman, J.W. (1985). *Statistical Optics*. Wiley, New York.
- Herman, G.T. (1980). *Image Reconstruction From Projections: The Fundamentals of Computerized Tomography*. Academic, New York.
- Herman, G.T., Lent, A. and Hurwitz, H. (1980). A Storage-Efficient Algorithm for Finding the Regularized Solution of a Large Inconsistent system of Equations. *J Inst Math Applic*, 25:361–366.
- Jain, A.K. and Farrokhnia, F. (1991). Unsupervised texture segmentation using Gabor filters. *Pattern Recogn*, 24:1167–1186.
- Kevles, B.H. (1997). *Naked to the Bone: Medical Imaging in the Twentieth Century*. Rutgers University Press, Piscataway, NJ.
- Li, K.C. (1991). Sliced inverse regression for dimensional reduction (with discussion). *J Am Stat Assoc*, 86:316–342.
- Li, K.C. (2000). *High dimensional data analysis via the SIR/PHD approach*. Lecture Notes, Department of Statistics, UCLA, Los Angeles, CA (<http://www.stat.ucla.edu/~kcli/sir-PHD.pdf>).
- Lu, H.H.S., Chen, C.M. and Yang, I.H. (1998). Cross-Reference Weighted Least Square Estimates for Positron Emission Tomography. *IEEE Trans Med Imag*, 17:1–8.
- Lu, H.H.S. and Tseng, W.J. (1997). On Accelerated Cross-Reference Maximum Likelihood Estimates for Positron Emission Tomography. *Proc IEEE Nucl Sci Symp*, 2:1484–1488.
- Malik, J. and Perona, P. (1990). Preattentive texture discrimination with early vision mechanisms. *J Opt Soc Am A*, 7:923–932.
- Meng, X.L. and Rubin, D.B. (1993). Maximum likelihood estimation via the ECM algorithm: A general framework. *Biometrika*, 80:267–278.
- Ouyang, X., Wong, W.H., Johnson, V.E., Hu, X. and Chen, C.T. (1994). Incorporation of Correlated Structural Images in PET Image Reconstruction. *IEEE Trans Med Imag*, 13:627–640.
- Politte, F.G. and Snyder, D.L. (1991). Corrections for Accidental Coincidences and Attenuation in Maximum-Likelihood Image Reconstruction for Positron-Emission Tomography. *IEEE Trans Med Imag*, 10:82–89.
- Prince, J.L. and Links, J. (2005). *Medical Imaging Signals and Systems*. Prentice Hall, Upper Saddle River, NJ.
- Shepp, L.A. and Vardi, Y. (1982). Maximum Likelihood Reconstruction for Emission Tomography. *IEEE Trans Med Imag*, 1:113–122.
- Soatto, S., Doretto, G. and Wu, Y. (2001). Dynamic textures. *Intl Conf Comput Vis*, 439–446.
- Spinks, T.J., Jones, T., Gilardi, M.C. and Heather, J.D. (1988). Physical Performance of the Latest Generation of Commercial Positron Scanner. *IEEE Trans Nucl Sci*, 35:721–725.

-
- Suetens, P. (2002). *Fundamentals of Medical Imaging*. Cambridge University Press, Cambridge.
- Tan, T.N. (1995). Texture edge detection by modelling visual cortical channels. *Pattern Recogn.*, 28:1283–1298.
- Tu, K.Y., Chen, T.B., Lu, H.H.S., Liu, R.S., Chen, K.L., Chen, C.M. and Chen, J.C. (2001). Empirical Studies of Cross-Reference Maximum Likelihood Estimate Reconstruction for Positron Emission Tomography. *Biomed Eng – Appl Basis Commun*, 13:1–7.
- Vardi, Y., Shepp, L.A. and Kaufman, L. (1985). A Statistical Model for Positron Emission Tomography. *J Am Stat Assoc*, 80:8–20.
- Weaver, H.J. (1983). *Applications of Discrete and Continuous Fourier Analysis*. Wiley, New York.
- Wu, H.M. and Lu, H.H.S. (2004). Supervised motion segmentation by spatial-frequential analysis and dynamic sliced inverse regression. *Stat Sinica*, 14:413–430.
- Wu, Y., Zhu, S.C. and Guo, C. (2002). Statistical modelling of texture sketch. *Proc Eur Conf Comp Vis*, 240–254.
- Wu, Y., Zhu, S.C. and Liu, X. (2000). Equivalence of Julesz texture ensembles and FRAME models. *Int J Comp Vis*, 38:247–265.
- Xiang, D. and Wahba, G. (1996). A generalized approximate cross validation for smoothing splines with non-Gaussian data. *Stat Sinica*, 6:675–692.
- Zhu, S.C., Liu, X. and Wu, Y. (2000). Exploring texture ensembles by efficient Markov Chain Monte Carlo – towards a ‘Trichromacy’ theory of texture. *IEEE Trans Pattern Anal Mach Intell*, 22:554–569.
- Zhu, S.C., Wu, Y. and Mumford, D.B. (1998). Filter, random field, and maximum entropy (FRAME): towards a unified theory for texture modelling. *Int J Comp Vis*, 27:107–126.

Exploratory Graphics of a Financial Dataset

IV.3

Antony Unwin, Martin Theus, Wolfgang K. Härdle

3.1	<i>Introduction</i>	832
3.2	<i>Description of the Data</i>	833
3.3	<i>First Graphics</i>	834
3.4	<i>Outliers</i>	837
3.5	<i>Scatterplots</i>	841
3.6	<i>Mosaic Plots</i>	843
3.7	<i>Initial Comparisons Between Bankrupt Companies</i>	844
3.8	<i>Investigating Bigger Companies</i>	848
3.9	<i>Summary</i>	851
3.10	<i>Software</i>	852

Introduction

The first stages of any data analysis are to get to know the aims of the study and to get to know the data. In this study the main goal is to predict a company's chances of going bankrupt based on its recent financial returns. In another chapter of the Handbook, some sophisticated prediction models based on support vector machines are discussed for a similar dataset. Here, visualization methods are used to explore the large dataset of American company accounts that was made available for predicting bankruptcy in order to get to know the data and to assess the quality of the dataset. This is an initial exploratory analysis that does not use any expert accounting knowledge.

Exploratory data analysis (EDA) has been a well-known term in the field of statistics since Tukey's historic book (Tukey, 1977). While everyone acknowledges the importance of EDA, little else has been written about it, and modern methods – including interactive graphics (Unwin, 1999) – are not as commonly applied in practice as they might be. Interactive graphics were used extensively in the exploratory work for this chapter. The dataset is by no means particularly big but it does contain more than 80 000 cases. Ways of graphically displaying large datasets are discussed in detail in Unwin et al. (2006).

When considering graphic displays, it is necessary to distinguish between presentation and exploratory graphics. Graphics for displaying single variables or pairs of variables are often used to present distributions of data or to present results. Care must be taken with scales, with aspect ratios, with legends, and with every graphical property that may affect the success of the display at conveying information to others. Graphics for exploration are very different. They are more likely to be multivariate, and there is no need to be particularly concerned about the aesthetic features of the graphic; the important thing is that they give a clear representation of the data. Presentation graphics are drawn to be inspected by many people (possibly millions of people, if they are used on television), and they can be long-lived. For example, Playfair's plots (Playfair, 2005) of English trade data are over 200 years old, but are still informative. Exploratory graphics are drawn for one or two people and may be very short-lived. A data analyst may examine a large number of graphics before finding one that reveals information, and, having found that information, the analyst may decide that another kind of display is actually better at presenting it than the display or displays that led to its discovery.

The graphics shown in this chapter are a subset of those used in the study. They are not supposed to be "pretty;" they have been drawn to do a job. Detailed scales have not been included, as it is always the distributional form that is of primary interest. Exploratory analyses are subjective, and each analyst may choose different graphics and different combinations of them to uncover information from data. The only thing that can be said with certainty is that analysts who do not use graphics to get to know their data will have a poor understanding of the properties of the data they are working with. If nothing else, graphics are extremely useful for assessing the quality of data and for cleaning data.

Description of the Data

There are 82 626 records in the dataset. Each record contains financial information for a company for one year. Table 3.1 gives a list of the variables in the dataset.

For each company there are 13 ratios, one size variable (log transform of assets), and a binary variable, which records whether or not the company went bankrupt within three years of the financial returns. There is also information on the State in which the company is registered, the industry sector to which it belongs, and the year of the accounts. The term “bankruptcy” includes Chapter 11 reorganization as well as liquidation under Chapter 7 of the US Bankruptcy Code. Financial ratios are commonly used so that data are comparable across years. Sometimes it is helpful to look at the raw data as well, particularly if there are data cleaning issues. An unusual value or a cluster of unusual values in *Total Assets* would affect all twelve ratios that depend on this variable. There were no missing values. Some of the ratio variables can be grouped into categories: profit measures (*Ebit.TA*, *NI.TA*); leverage ratios (*Kap.TA*, *TL.TA*, and *Eq.TA*); liquidity ratios (*Cash.TA*, *CA.TA* and *CA.CL.TA*); and activity/turnover ratios (*Inv.TA*, *S.TA*, and *Ebit.Int*).

In order to be able to generalize results from statistical models, the dataset being analyzed must be a random sample from the population under study. For the bankruptcy dataset, there are a very large number of cases, but it is not clear whether they comprise a random sample. Apart from anything else, it is not at all clear that they can be considered to be homogeneous. Most companies are rather small and

Table 3.1. Variables in the Bankruptcy dataset

Variable	Description
Cash.TA	Cash/Total Assets
Inv.TA	Inventories/Total Assets
CA.TA	Current Assets/Total Assets
Kap.TA	Property, Plant and Equipment/Total Assets
Intg.TA	Intangibles/Total Assets
LogTA	log(Total Assets)
CL.TA	Current Liabilities/Total Assets
TL.TA	Total Liabilities/Total Assets
Eq.TA	Equity/Total Assets
S.TA	Sales/Total Assets
Ebit.TA	EBIT/Total Assets
Ebit.Int	EBIT/Interest Payments
NI.TA	Net Income/Total Assets
CA.CL.TA	(Current Assets–Current Liabilities)/Total Assets
BANKR	status (Bankrupt or OK)
Year	Year of accounts
State	Where the company was registered
NAICS	North American Industry Classification System

a few are very large. Can the same financial ratios really be used for companies that are so different in scale? This is the kind of question that exploratory analysis can help us to answer by looking at the distributions of data values for the different groups.

The assumption is that results from this dataset can be applied to datasets collected in a similar fashion in the future.

3.3

First Graphics

Figure 3.1 is a barchart for the bankruptcy variable. Only a small proportion of companies actually went bankrupt, which is a good thing for the companies, but it makes any statistical analysis more difficult.

The displays in Figs. 3.2 and 3.3 show that the companies are fairly equally distributed across the regions, but that some of the data are surprisingly old, with a few cases prior to the mid 1960s. In the 1950s there is one record per year, and by linking to the geographic data we can show that this was not always for the same company, as might have been suspected.

The geographic information was originally provided by State, so there were many small counts and only a few big ones. Grouping by region gives a good overview, although other groupings (e.g., by population) could also be attempted. The regional classification used here is one from the FBI. Selecting the foreign group (comprising a little more than 10 % of the cases) and linking to a spinogram (Hofmann and Theus, submitted) of years (see Fig. 3.4) shows that the percentage of foreign registered companies has increased over the years. Querying shows that this percentage increases from about 4 % in the early 1970s to 15 % in 2002. In the most recent year, 2003, the rate falls to just under 11 %. It is expensive for foreign firms to be listed on the US exchanges, and opinion has changed as to what benefits it brings. The Sarbanes–Oxley Act has also made it less attractive to be listed in the US.

Information was also available on industry sector in two different ways; one classification involved 426 categories by name and numerical coding, while the NAICS classification used number, and had 925 categories. Both are too detailed for graphical analysis, and a hierarchical grouping similar to the spatial grouping of the States can be attempted. The six-digit NAICS codes can be aggregated by their first two digits and then further grouped by sector to give Fig. 3.5.

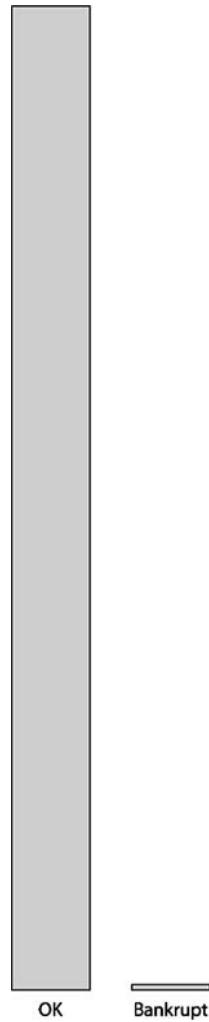


Figure 3.1. A bankruptcy barchart. 506 of the 82626 records refer to bankrupt

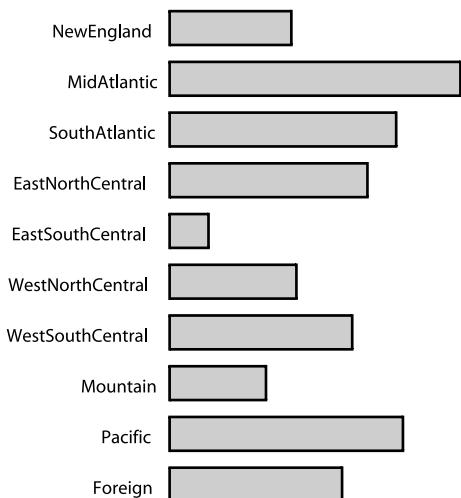


Figure 3.2. A barchart of the number of records by US region

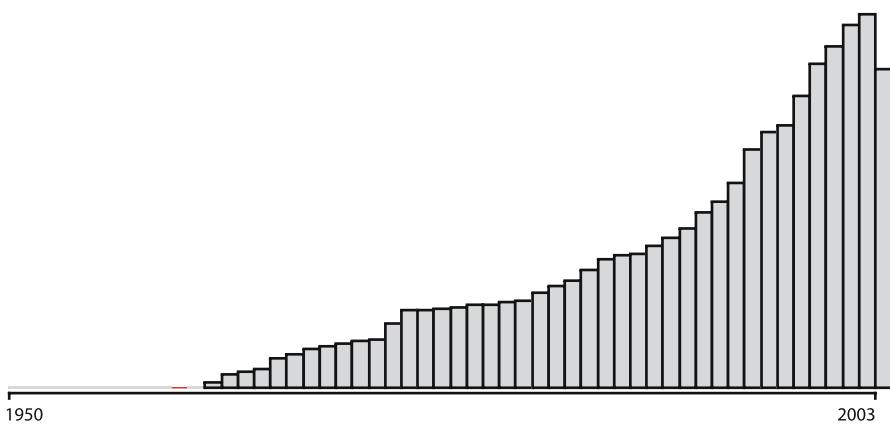


Figure 3.3. A histogram of the number of records per year

The manufacturing sector clearly dominates. To check for associations between the two classifications, a crude scatterplot approach was tried. A fairly random spread of points was obtained, but no particular pattern.

Histograms can be drawn for the continuous variables, but they take up quite a lot of room, so boxplots are more efficient at displaying many variables at once. Since $\log TA$, the log of total assets, is a different kind of variable from the others and important in its own right too (because it groups the companies by size), a histogram has been drawn specifically for it (see Fig. 3.6). This shows a roughly symmetric distribution with a few very low values. Selecting the foreign registered companies again and linking to a spinogram of $\log TA$ reveals that for the 25 % of the companies that are the biggest, the percentage of foreign registered companies rose steadily, from 7 % up to more than 50 %.

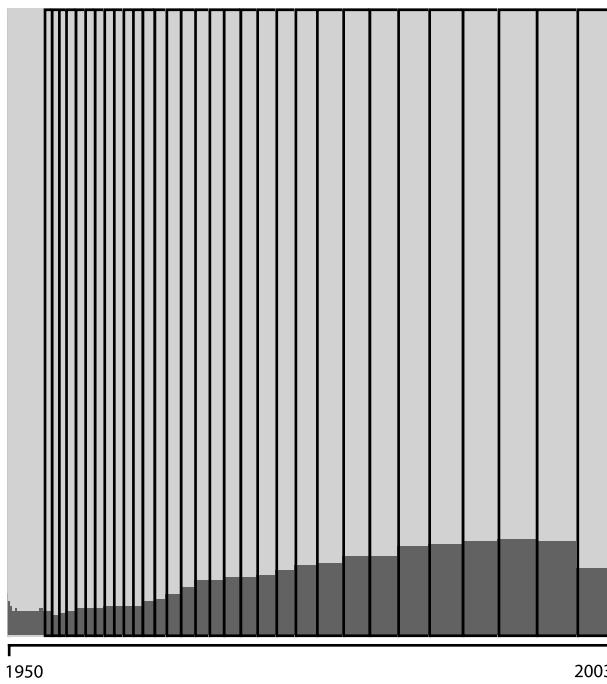


Figure 3.4. A spinogram of the number of records per year, with foreign registered companies selected. The width of a bar in the spinogram is proportional to the height of the corresponding bar in the original histogram

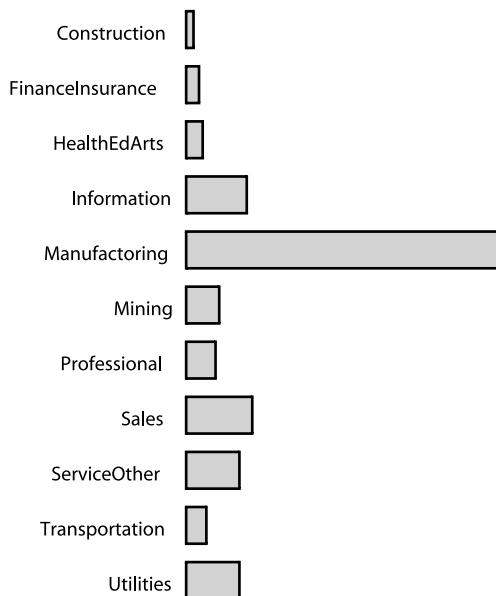


Figure 3.5. A barchart of the number of records categorized by NAICS group

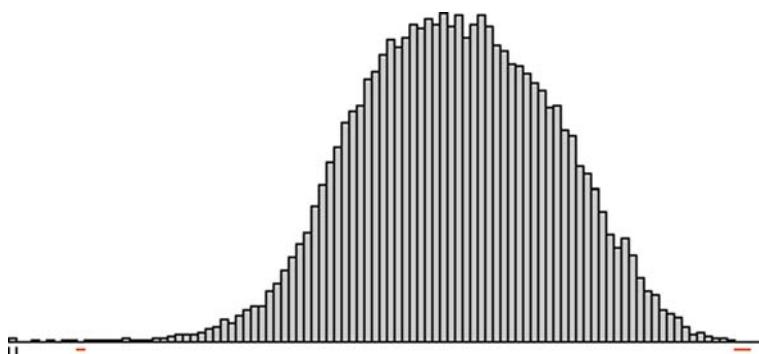


Figure 3.6. A histogram of $\log(\text{Total Assets})$, $\log\text{TA}$, for the companies. The marks below the axis to the left are interactive controls for the anchorpoint and binwidth. The horizontal (red) marks record bins where the count is too small to be drawn

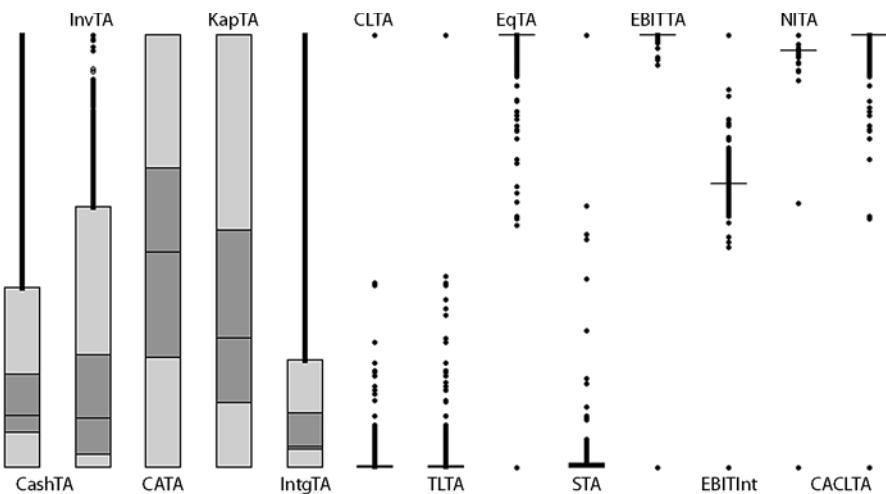


Figure 3.7. Parallel boxplots of financial ratios. Each boxplot is individually scaled

A set of parallel boxplots for the ratio variables is shown in Fig. 3.7. The boxplots reveal that several of the ratios are highly skewed, and this may affect whether they can be of much help in discriminating between the companies which went bankrupt and those which did not. It is possible that many of these outliers are errors of some kind, and it may be that most of the outliers for individual variables are actually outliers for several variables.

Outliers

3.4

Outliers can be checked with a parallel coordinates display (Inselberg, 1999), as in Fig. 3.8, where the eight ratios with highly skewed distributions have been plotted

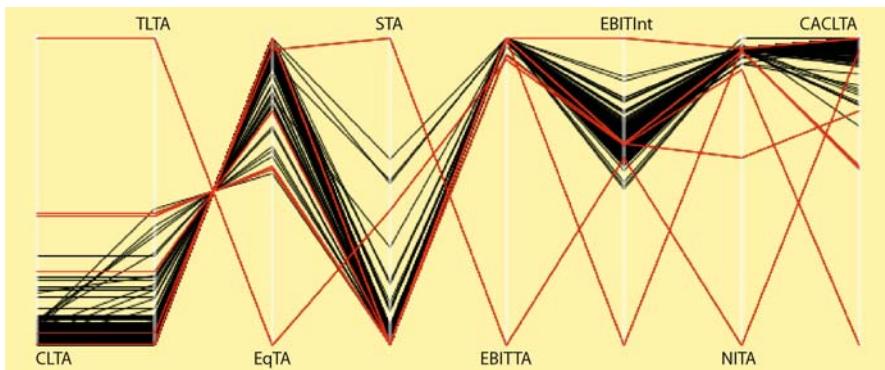


Figure 3.8. [This figure also appears in the color insert.] Parallel coordinate plot of financial ratios with skew distributions. Seven outliers have been selected

and seven of the worst outliers selected. It is easy to see that several are outliers for more than one variable. It is also apparent that the ratio of equity to total assets $Eq.TA$ is perfectly inversely correlated with the ratio of total liabilities to total assets $TL.TA$. This is a matter of definition, with $TL.TA + Eq.TA = 1$. Although this equation looks innocuous, it masks the fact that $TL.TA$ ranges from 0 to 5838 in this dataset. Not surprisingly, the high values of $TL.TA$ only arise for low values of *Total Assets*, as the L-shaped scatterplot on the left of Fig. 3.9 shows. This plot is somewhat misleading. The zoomed version on the right reveals that there is more variability amongst the low values than the default plot suggests, although the most extreme values of $TL.TA$ still only occur for very low values of *Total Assets*. The bulk of the pattern suggests that very small companies have a broader range of possible liability ratio values than small companies. The low-density region to the lower right implies that companies

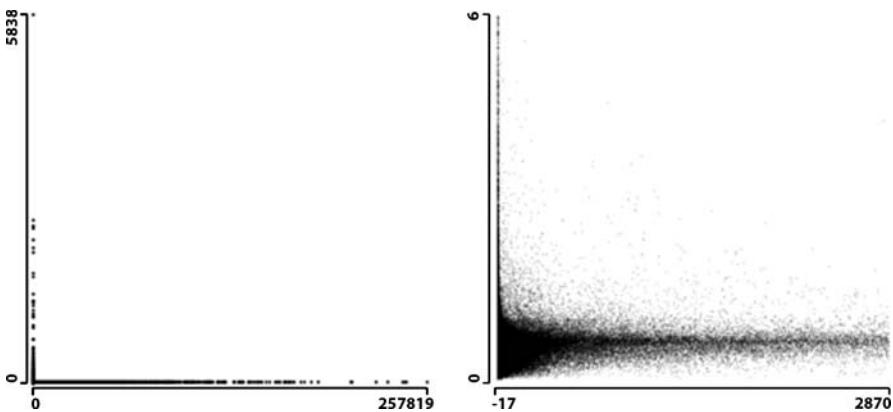


Figure 3.9. Scatterplots of $TL.TA$, the ratio of total liabilities to total assets, plotted against *Total Assets*. The left-hand plot presents all of the data, and it shows that all high values of liabilities are associated with low asset values. The right-hand plot presents a zoom of about 10^{-2} on the *x*-axis by 10^{-3} on the *y*-axis, along with some α -blending

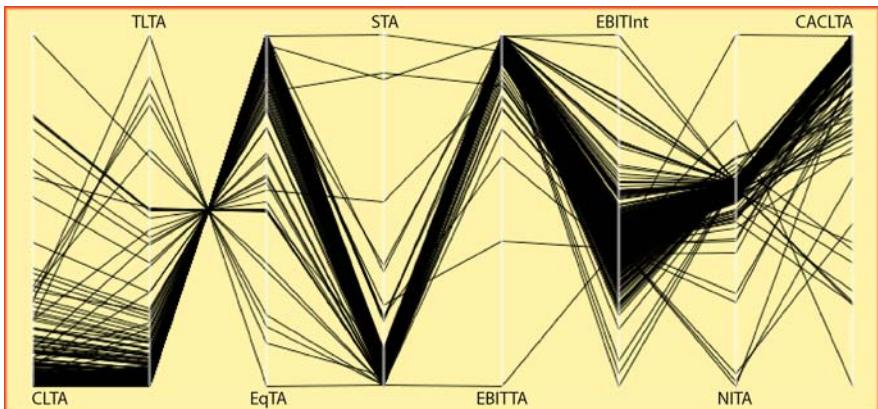


Figure 3.10. [This figure also appears in the color insert.] Parallel coordinate plot of the financial ratios with skewed distributions. The seven outliers selected in Fig. 3.8 have been removed. The plot's (red) border is a sign that not all data are displayed

exceeding a certain size must have some liabilities. The distorting effect of the extreme values is demonstrated by the fact that the zoomed plot, which covers a region 10^{-5} times the size of the default plot, contains 87% of the data. α -Blending has been used to make the distributional structure more visible. α -Blending weights the shading of each object by a specified fraction. The darkness of a pixel is that fraction multiplied by either the number of objects overlapping the pixel or 1, whichever is smaller.

Setting aside the seven worst outliers, Fig. 3.8 rescales to Fig. 3.10. Some of the variables become potentially more informative, while the scales of some of the others are dominated by newly visible outliers.

Outliers can be dealt with in several ways. A transformation might help (but it is not always appropriate; in this case, several ratios have some negative or zero values). The outliers could be trimmed or discarded (depending on what kinds of outliers they are). With ratio variables, such as those used here, the component variables of the ratios can be examined. Figure 3.11 shows a scatterplot of the variable *Sales* against *Total Assets* with the same seven outliers still highlighted. All turn out to be small companies in terms of both *Sales* and *Total Assets*, and small companies should probably be treated separately anyway. The scatterplot also reveals that there are some bivariate outliers. Querying shows that the six cases with high *Sales* but low *Total Assets* (the upper left group) are all data for the same big retail company over several successive years. The seven cases with low *Sales* but very high *Total Assets* (the lower right group) represent three companies from the information and communication sector. These two groups make up a tiny proportion of the dataset, even if they all have very large *Sales* or *Total Assets*.

For exploratory work it is distributional structure that is of interest, not precise values. The minima and maxima of the variables are used to determine the limits of the axes. The lower limit for *Sales* in Fig. 3.11 shows that the lowest *Sales* values are actually negative and that plenty of them are zero or almost zero too. The negative

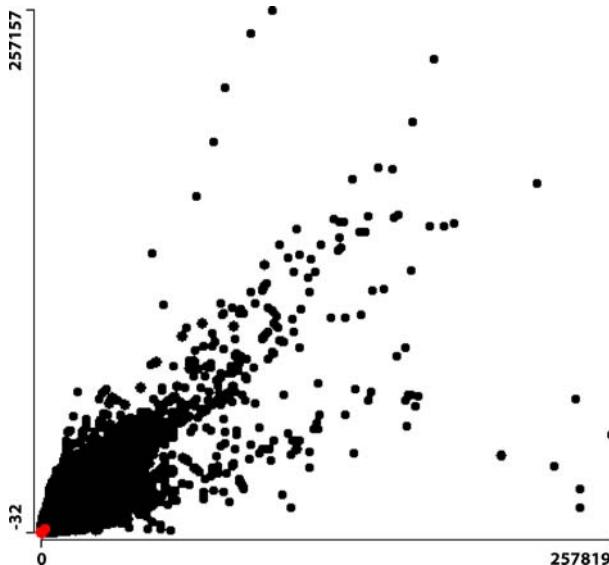


Figure 3.11. Scatterplot of *Sales* vs. *Total Assets* with the seven outlying companies highlighted (the lighter blob in the lower left corner)

values could be an accounting quirk, but should undoubtedly be discarded from the dataset, although there are only five of them. The low *Sales* values are also worth considering; what types of companies do these correspond to, and should they be kept in the dataset? They may be very new companies or companies on their last legs. There are 1412 companies with zero *Sales* and another 3674 with *Sales* that are more than zero but less than 1. These data could in principle have been obtained by zooming into a histogram for *Sales* and querying the cells, but for queries with precise boundaries it is quicker to calculate the appropriate frequency table. Graphics are better for more general, qualitative insights, while tables and statistical summaries are better for exact details.

The empirical distributions can be examined in many ways. When cases are of different sizes or weights, it can be illuminating to look at weighted distributions. For instance, Fig. 3.12 shows a histogram of *CA.TA*, the ratio of current assets to total assets, on the left and a histogram of the same variable weighted by *Total Assets* on the right. Companies with the highest current assets ratios clearly have low *Total Assets*.

Outliers and negative values are some of the data cleaning problems that can arise; there may be others as well.

Some statistical modeling approaches are hardly affected by individual gross errors, and it may be that it matters little to the model fit whether these cases are excluded, adjusted, or just included in the analysis anyway. Even when this is the case, it is useful to know what kinds of errors or unusual values can occur. It is also an opportunity to talk to people who know the context of the data well, and to get them to provide more background information. Analyzing data blind without any background information is just reckless. A surprising (and sometimes shocking) amount

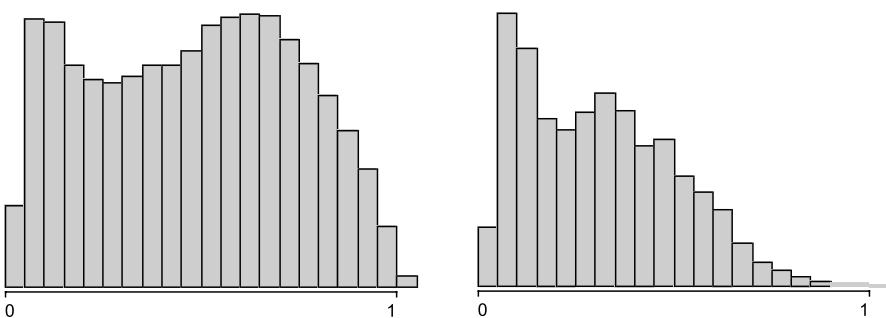


Figure 3.12. A histogram of the current assets ratio on the left and a weighted histogram of the same variable, weighted by *Total Assets*, on the right

of the useful information known about datasets is not incorporated into analyses, such as information about the selection of the variables collected, the way the sample was chosen, or details about the collection and recording of data. Dataset owners may assume that analysts know this information, but it is useful to check.

Scatterplots

3.5

There is often a temptation with large numbers of continuous variables to calculate correlation coefficients for all possible pairs of variables. The trouble with this approach is that correlations can be high because of outliers, or low because the association between the two variables is nonlinear. Fourteen variables (the original ratios plus $\log TA$) are too many to draw a scatterplot matrix, but it is interesting to look at the associations between subsets of the ratios. Scatterplots for two pairs of the financial variables *Cash.TA*, *Inv.TA*, *CA.TA* and *Kap.TA* are shown in Fig. 3.13. The trian-

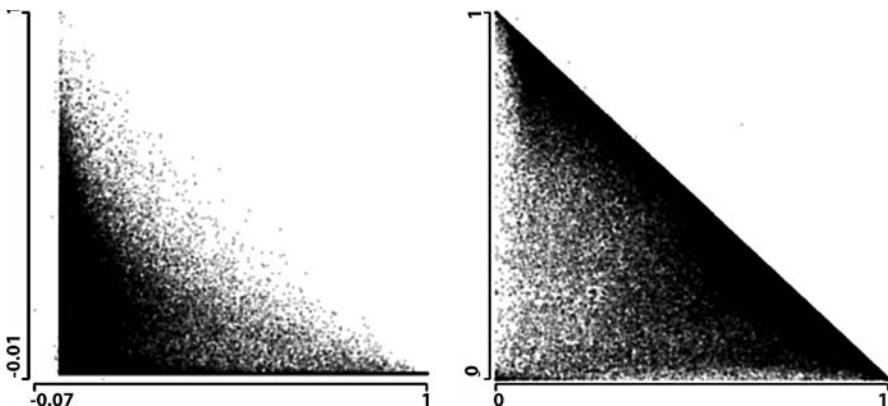


Figure 3.13. Scatterplots of *Cash.TA* and *Inv.TA*, the ratios of cash and inventories to total assets (left), and of *CA.TA* and *Kap.TA*, current assets and property to total assets (right)

Table 3.2. Correlations between the four ratio variables in Fig. 3.13

	Cash.TA	Inv.TA	CA.TA	Kap.TA
Cash.TA	1.000			
Inv.TA	-0.224	1.000		
CA.TA	0.482	0.553	1.000	
Kap.TA	-0.378	-0.352	-0.752	1.000

gular shapes to the lower left show that the sum of the corresponding ratios is less than a particular limit; 1 here. (The triangular shapes to the lower right are obtained when the y variable is always lower in value than the x variable, such as when the cash or inventory ratio is plotted against the current assets ratio.) The negative cash values that have already been discussed are easily seen to the left of the bulk of the data in the plots of cash and inventory ratios. In the plot of the current assets and property ratios there are a few companies that surprisingly lie above the bounding diagonal – more outlying cases to investigate. The correlation coefficients for the variables are given in Table 3.2. Some of them are quite high, and they are certainly all significantly different from zero, but they hint little at the structure shown. The highest value in absolute terms is the correlation between current assets and property (-0.752). This deserves further investigation, and Fig. 3.14 shows the same scatterplots as shown in Fig. 3.13, but this time using the smallest pointsize and some α -blending instead of the defaults, providing a rough bivariate density estimate. It is now possible to see that, for many of the companies, the sum of current assets and property assets is almost equal to the *Total Assets*.

The scatterplots of the corresponding raw data variables provide another view of the associations between variables in the dataset, and more structure can be seen than with scatterplots of the ratios. A few of the raw scatterplots exhibit a funnel structure, such as the plot of inventories against fixed assets and that of cash against current as-

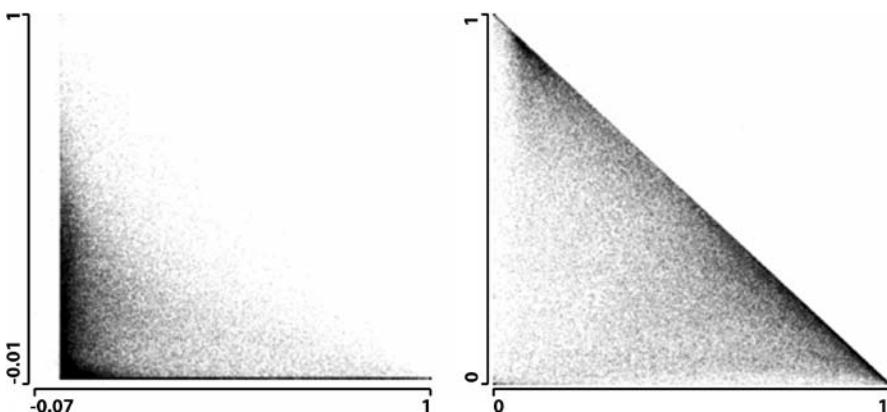


Figure 3.14. The same scatterplots as shown in Fig. 3.13, but with a smaller pointsize and α -blending to better display the bivariate structures

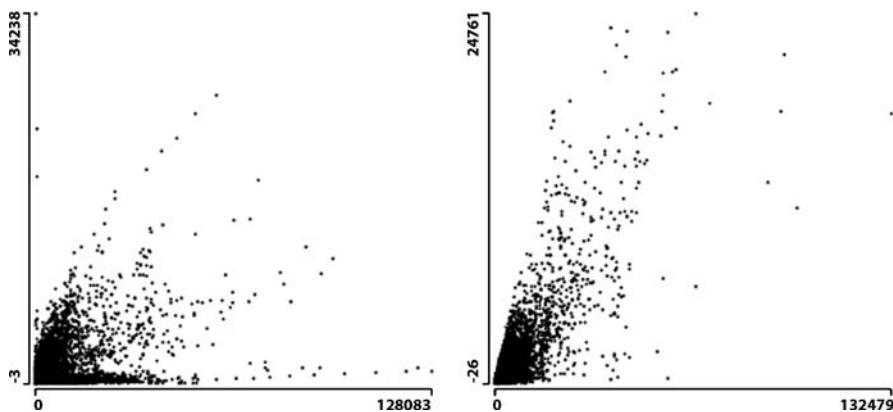


Figure 3.15. Scatterplots of inventories against fixed assets (*left*) and of cash against current assets (*right*)

sets (Fig. 3.15). Querying and linking can be used to identify specific sectors or outliers. The companies with high assets and low inventories are in the information and communications sector. Companies in the retail sector have higher inventories and lower assets. The three points to the top left in the inventories/fixed assets plot which initially appear to be unusual are from investment banking and are consistent with other companies in that sector – except for being considerably bigger. The biggest companies for both variables in the left-hand plot are car manufacturers (higher inventories) and oil companies (higher assets).

Sometimes features stand out in a parallel coordinate plot, while sometimes they are more visible in raw data scatterplots. As always, a range of different graphics should be explored.

Mosaic Plots

3.6

Combinations of categorical variables can be displayed in mosaicplots of various kinds (Hofmann, 2000). However, if we attempt to draw such plots we encounter two difficulties. Firstly, all but one of the categorical variables have a large number of categories (there are 54 states and 925 NAICS categories). Secondly, the one categorical variable that is binary (*BANKR*) comprises less than 1% of the cases in one category, so that highlighting is rarely visible. The first problem can be circumvented to some degree by grouping, combining states into regions, and using a less detailed form of the NAICS. The second problem can be solved by using a special form of zooming.

Figure 3.16 shows a fluctuation diagram of the numbers of companies by industry sector and region. Classical mosaicplots try to make the most efficient use of the space available by making each cell as large as possible. This can make the plot difficult to interpret, especially when there are many cells. Fluctuation diagrams preserve a grid

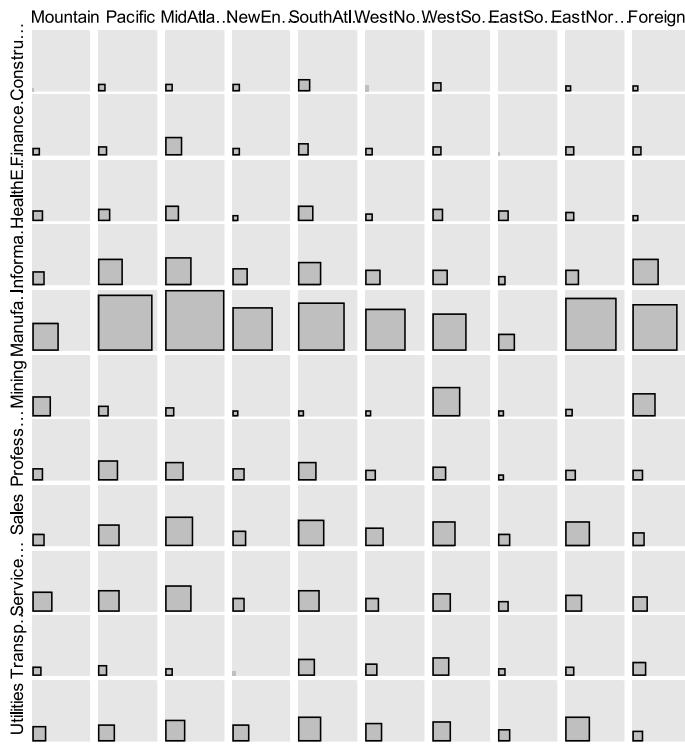


Figure 3.16. Fluctuation diagram of industry sectors by region

structure, so that comparisons by rows and columns are easier. The dominance of the manufacturing sector stands out in Fig. 3.16, as well as individual details, such as the concentration of mining companies in WestSouthCentral.

All cases are treated equally in Fig. 3.16. Of course, some companies are much bigger than others, and Fig. 3.17 shows the same cases weighted by *Total Assets*. The differences between the two figures are striking. Foreign registered companies in the manufacturing sector are much bigger than was apparent previously. The regional distribution of companies in the professional sector changes a lot.

Initial Comparisons Between Bankrupt Companies

Up to now, our analysis has considered all of the data as one group, and a number of outliers, distributional features, and specific properties have been identified. The main aim of the study is still to investigate how the companies that went bankrupt differ from the rest. A first approach would be to look at the information available. Any company whose total liabilities exceed their total assets ($TL.TA > 1$) is likely to be

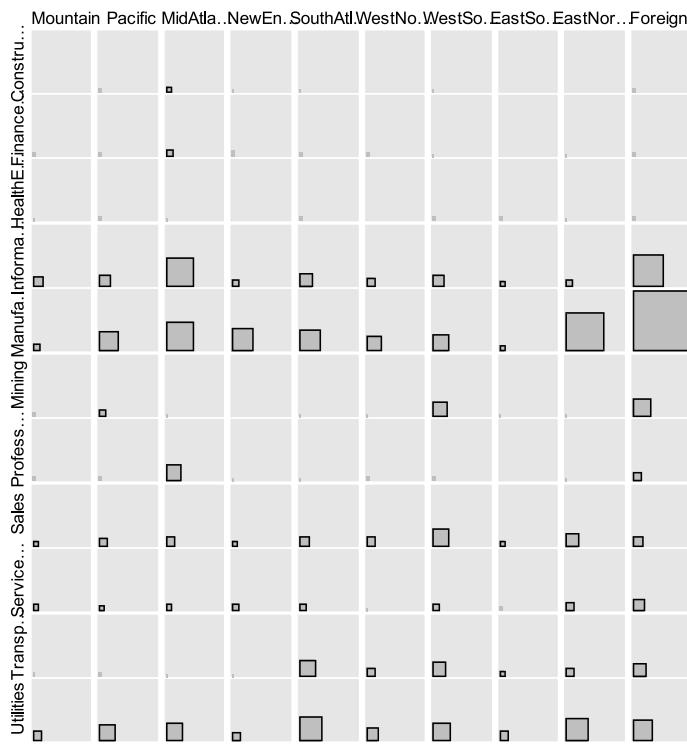


Figure 3.17. Fluctuation diagram of sectors by regions weighted by *Total Assets*

in trouble. This variable is bizarrely skewed and so difficult to visualize with standard plots. In this situation a simple table is better.

Clearly the variable *TL.TA* on its own is very informative. The choice of the boundary limit (1) is determined by the context. Varying the limit interactively using a slider confirms that it is a good choice in that the differences in both bankruptcy rates and the number of bankrupt companies are high.

To investigate the effects of more than one variable, two kinds of parallel plots can be used with the bankruptcy cases highlighted. Parallel boxplots give univariate summary comparisons, while parallel coordinate plots potentially offer multivariate comparisons.

Figure 3.18 shows the default parallel coordinate plot of the ratios (without *Eq.TA* but with *logTA*) drawn for all data except for the 55 outliers removed in an initial data clean. The companies that went bankrupt have been selected. The heavy overplotting may be obscuring information, and the fact that every line is drawn in the same way whether it represents one case or many may also mislead. Nevertheless, some features can be identified: all bankrupt companies had low values of *CL.TA*, high values of *Ebit.TA*, and medium values of *Ebit.Int*; there are two unusual bankrupt companies (one a high outlier on *TL.TA*, and the other a high outlier on *S.TA*).

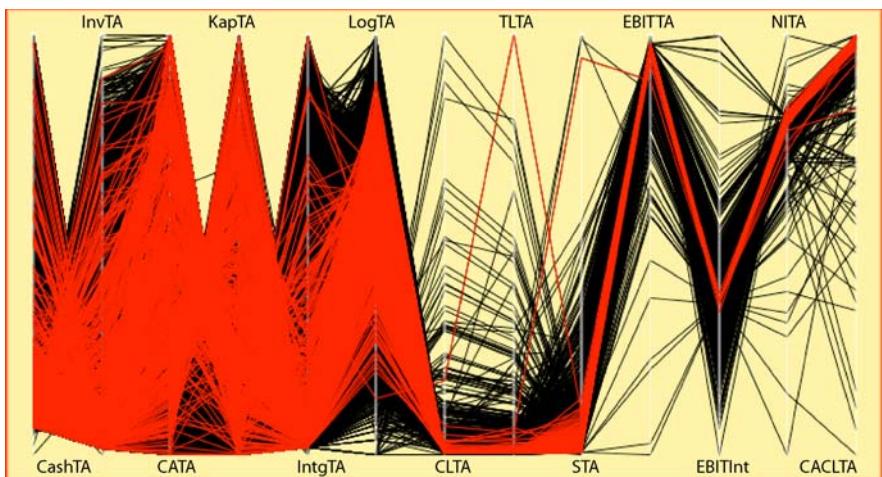


Figure 3.18. Parallel coordinate plot of financial ratios and $\log TA$, excluding 55 outliers, with bankrupt companies highlighted

One way to get better discriminatory power is to use α -blending. A factor of 0.1 has been used in Fig. 3.19, and it is now possible to see that the concentration of values for bankrupt companies for some of the variables applies to the bulk of the rest of the data as well, so that variables like $CL.TA$ and $Ebit.TA$ will not be as informative as might have been hoped. A final step can be to apply α -blending to the highlighting as well, and this has been tried in Fig. 3.20. This suggests that $Cash.TA$ and $Intg.TA$ might be more helpful than at first thought.

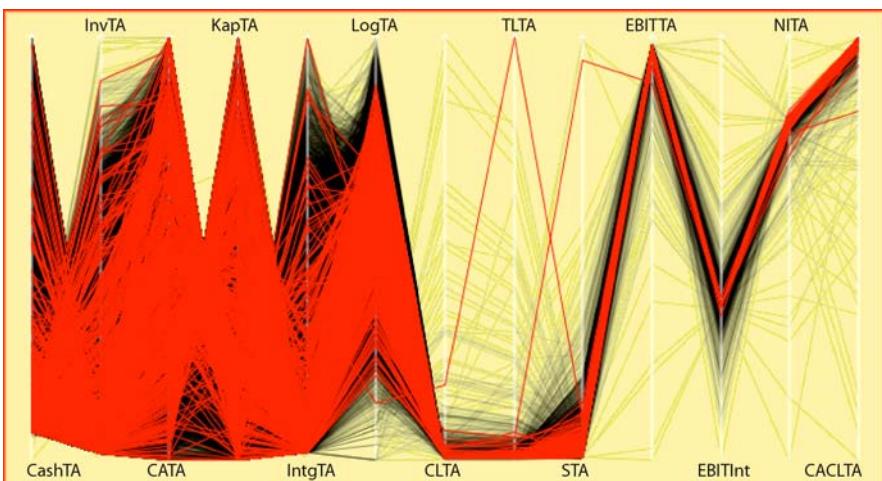


Figure 3.19. Parallel coordinate plot of financial ratios and $\log TA$, excluding 55 outliers, with bankrupt companies highlighted, α -blending=0.1 only for unselected data

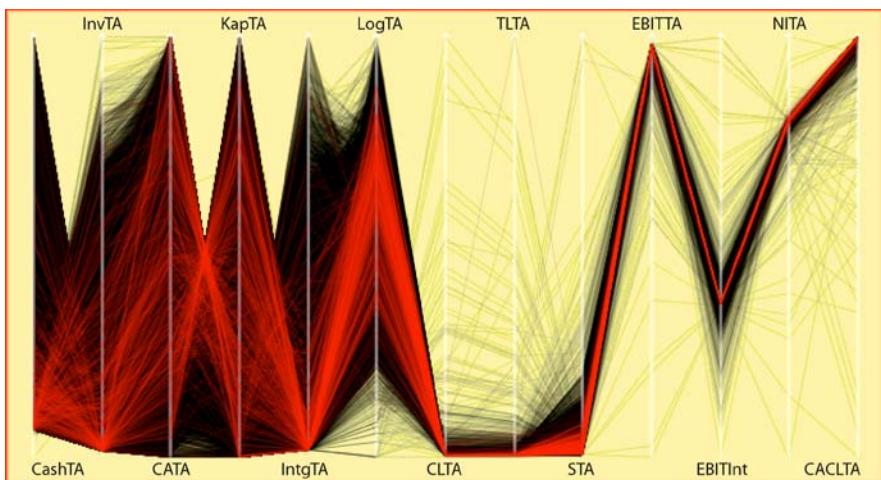


Figure 3.20. Parallel coordinate plot of financial ratios and $\log TA$, excluding 55 outliers, with bankrupt companies highlighted, α -blending=0.1 for selected and unselected data

The utilization of selection and linking for scatterplots can be highly effective, but it is dependent on the data distribution. Consider the two variables just mentioned above, *Cash.TA* and *Intg.TA*, and their distribution for the companies that went bankrupt. A scatterplot can be drawn with a little α -blending, as shown on the left of Fig. 3.21, or with a lot of α -blending, as shown on the right. (α -Blending has not been used for the highlighted cases, as they then disappear on this scale.) Both plots contribute information but neither is fully satisfactory. Another alternative would be to draw a pair of scatterplots, one for each of the two groups of companies, but this is difficult to interpret. The success of a particular plot depends on there being clear-cut

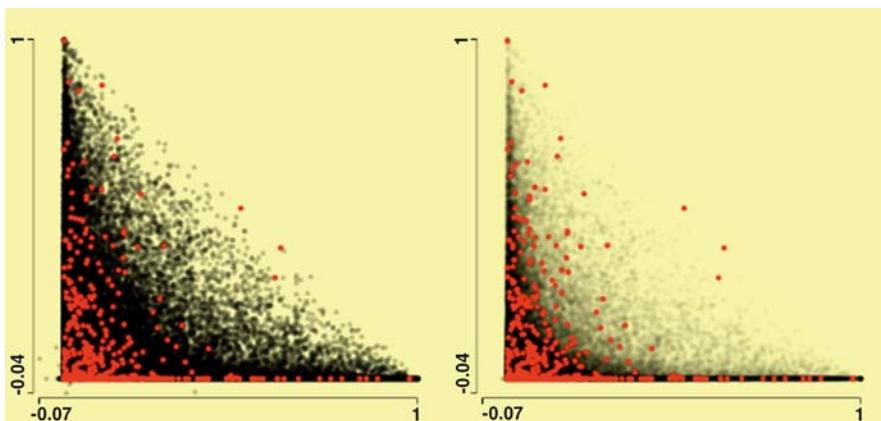


Figure 3.21. Scatterplots of the ratios of intangibles and cash to total assets, with companies that went bankrupt selected. More α -blending has been used in the right-hand plot

information to find. The differences between the companies that went bankrupt and the others that did not are more complicated than can be displayed in a scatterplot of two variables.

Another alternative would be to use spinograms (as shown in Fig. 3.24), but the number of bankrupt companies is so small relative to the total number of companies that little can be seen. Fitted smooths would be better, although they require more computation.

The parallel coordinate plots employed here are a selection of many that might have been shown. The choice and order of variables influences what might be seen. The decision to discard some extreme outliers does too. The level of α -blending is also an influential factor. In other words, a parallel coordinate plot is like any other multivariate analysis in that the user has a lot of freedom of choice. Careful thought helps, and so does statistical modeling. Having explored the data and built a model, parallel coordinate plots can again be useful, this time as a way to understand the model's relationship to the dataset.

3.8

Investigating Bigger Companies

Financial data for small companies is highly variable and could well be more unreliable than data for large companies, although this is difficult to assess. Large companies are certainly different from small companies, and so studying them separately makes sense. In one important way they do not differ: the bankruptcy rate for the biggest 7690 companies (each with *Total Assets* > 4000), 0.59 %, is close to the rate for the rest of the dataset, 0.62 % for the remaining 74 936 companies. On the other hand, a looser definition of big (*Total Assets* > 1000) yields bankruptcy rates of 0.73 % for the 18 610 “big” companies and 0.58 % for the rest, a significant difference. Given the many different limits that might be used, a wide range of results is possible. However, it is not modest variations in bankruptcy rates by company size that are of interest to us; it is identifying which companies might go bankrupt.

Over forty years it would be reasonable to expect that company size has increased. Curiously, for this dataset, the effect on *logTA* is negligible, as Fig. 3.22 shows.

Figure 3.23 shows boxplots of the original ratio variables for all of the data with the group of big companies highlighted. Only the first five financial ratios are shown, as the distributions of the others are, as Fig. 3.7 shows, far too skewed to be informative. *logTA* is included to show the size distribution. Although the medians for the bigger companies differ noticeably from the median ratios for all companies (for cash, inventories and hence, obviously, current assets, they are lower, while the median ratio is higher for the capital assets ratio), the distributions overlap substantially.

A more effective way of looking at the ratios is to use spinograms. In Fig. 3.24 there are plots of the ratios *Cash.TA*, *Inv.TA*, *Kap.TA*, and *Intg.TA* with the big companies selected. The proportion of big companies declines as the cash ratio increases; it also declines as the inventory ratio increases, apart from the lowest group (*Inv.TA* < 0.01),

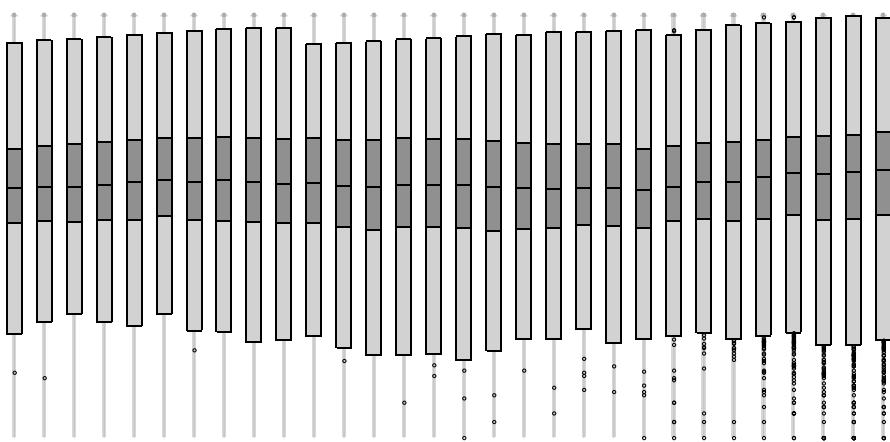


Figure 3.22. Parallel boxplots of $\log TA$ by year from 1974 to 2003, all on the same scale

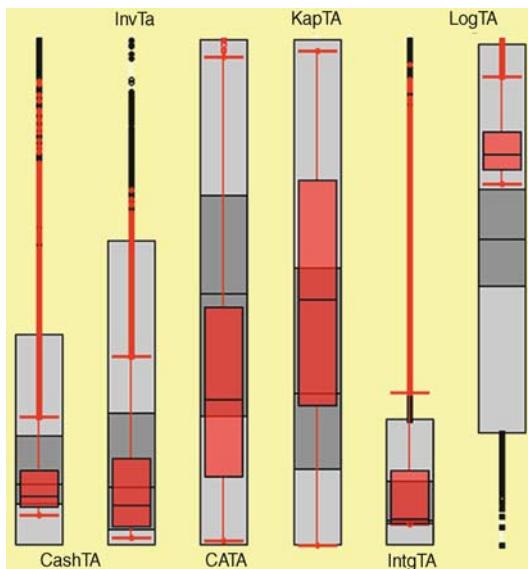


Figure 3.23. Parallel boxplots of financial ratios and $\log TA$ for all companies. The background boxplots are for all of the data, and the superimposed standard boxplots are for the selected cases, companies with *Total Assets* > 1000

where the proportion is relatively low; the proportion increases as the fixed capital ratio increases; and the proportion is fairly constant for the intangibles ratio.

Figure 3.25 shows just the data for the bigger companies for all variables, with the bankrupt companies selected. Outliers still affect most of the second group of ratios, but two features stand out: *TL.TA* is generally higher, as we would expect from Table 3.3 for the whole dataset, and *Ebit.TA* is generally lower. It is tempting to draw

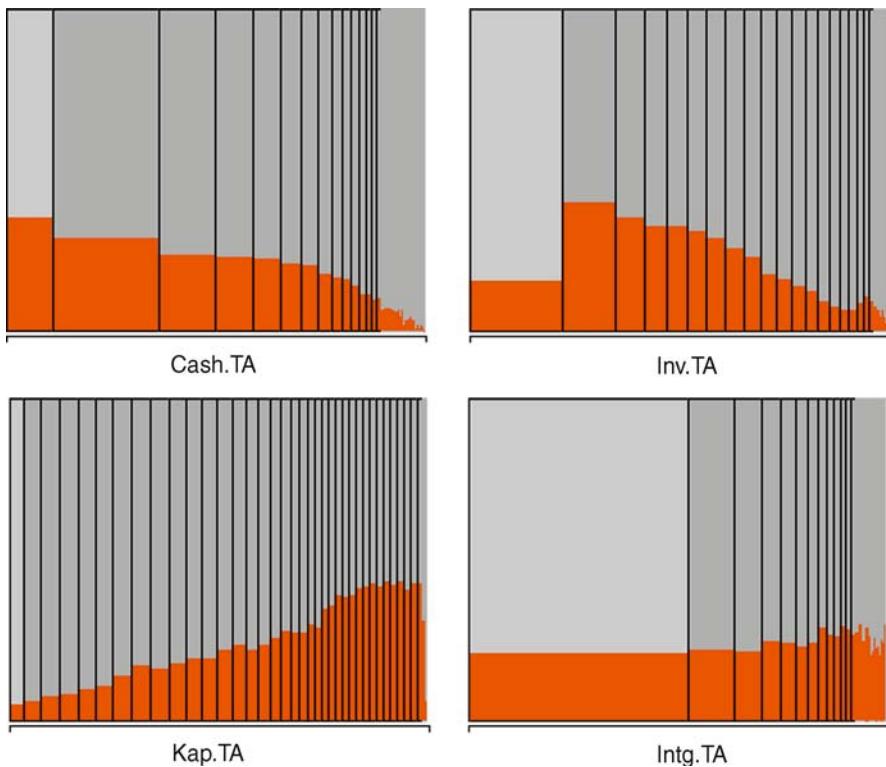


Figure 3.24. Spinograms of the ratios *Cash.TA*, *Inv.TA*, *Kap.TA*, and *Intg.TA*. The companies with *Total Assets* > 1000 have been selected

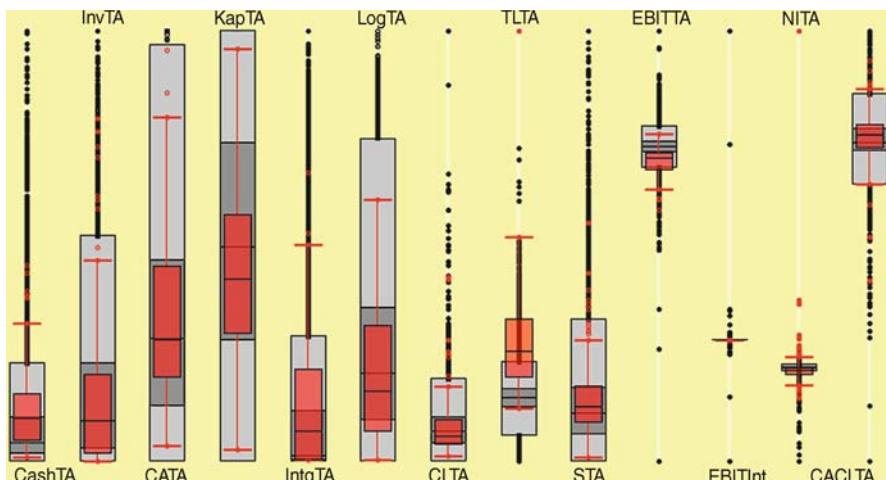


Figure 3.25. Parallel boxplots of financial ratios and $\log TA$ for the 18610 companies with *Total Assets* > 1000. Companies that went bankrupt have been selected

Table 3.3. Cases with more liabilities than assets, and their bankruptcy status

	Bankrupt	OK
TL.TA > 1	272 4.44 %	5856 95.6 %
TL.TA ≤ 1	234 0.31 %	76264 99.7 %

conclusions about the fact that none of the biggest companies went bankrupt and that none of the companies that went bankrupt had a high cash ratio (even though the median is higher for these companies), but the selected cases make up such a small percentage of the total that caution should be exercised before drawing any conclusions.

Summary

3.9

Every data analysis is unique because the data are always different. In the study reported here, there were mainly continuous variables (so parallel coordinate plots were useful); the few categorical variables usually had large numbers of categories (so these had to be combined into groups); there was a fairly large number of cases (so approximating density estimations were helpful); and there were some variables that were highly skewed (so that outliers and transformations were issues). A variety of plots were used, including barcharts, histograms, spinograms, boxplots, scatterplots, mosaicplots and parallel coordinate plots. Weighted versions of some plots also contributed. Trellis displays might have been tried, but then shingling of the conditional variables would have been required. That would be more appropriate after modeling. Interactivity, primarily selection, querying and linking, was used extensively, as is clear from the plots, but zooming and reformatting were also used a lot in the exploratory analyses. It is not easy to illustrate EDA in print, and the chapter can only convey a pale shadow of the actual process.

Data exploration is an important part of any data analysis. It is necessary to learn about the data, to check data quality and to carry out the data cleaning that is needed (and data cleaning is always needed with real datasets). EDA revealed here that there were some extreme outliers and some suspicious negative values. It underlined the need to transform some of the variables and it highlighted the geographic and sectoral structure of the dataset. It also revealed the surprising age of some of the data and the unexpected stability of the size distribution over time. Several interesting associations between variables were uncovered. An investigation of the factors influencing bankruptcy provided further insight into the data and prepared the ground for statistical modeling.

Applying statistical models before exploring data is an inefficient approach. Problems may arise because of peculiarities in the data. Features are revealed that could have been found much more easily just by looking. The only possible advantage of

modeling without exploring the data first is the purist one of satisfying the idealist prerequisite for a hypothesis test – although whether that is relevant for any real analysis is moot.

Graphics are essential for exploratory work. They provide overviews and insights that complement statistical summaries. On the other hand, graphical analyses without follow-up analytic support remain inconclusive. Visualization results tend to be qualitative rather than quantitative, general rather than precise. Statistical modeling can assess the strength of evidence supporting ideas generated from graphical EDA and help to define those ideas more exactly. On top of that, statistical modeling can tease out more complex relationships that are not immediately visually apparent. However, there is not much point in looking for complex relationships if the quality of the data is in doubt, which is one of the reasons that modeling benefits from prior data visualization. Modeling also benefits from graphical support after analysis, in relation to investigating residual patterns for individual models and comparing and combining groups of models. These are discussed in other chapters in the Handbook.

3.10

Software

The software used for most of the displays in this paper was Martin Theus's Mondrian (<http://stats.math.uni-augsburg.de/Mondrian/>). Other software was used at various stages to assist with data cleaning and restructuring.

Acknowledgement. Financial support from the Deutsche Forschungsgemeinschaft via SFB 649 "Ökonomisches Risiko" is gratefully acknowledged. Thanks also to Rouslan Moro, Dorothea Schäfer and Uwe Ziegenhagen for helpful comments on earlier drafts.

References

- Hofmann, H. (2000). Exploring categorical data: interactive mosaic plots, *Metrika*, 51(1):11–26.
- Hofmann, H. and Theus, M. (submitted). Interactive graphics for visualizing conditional densities, *Journal of Computational and Graphical Statistics*.
- Inselberg, A. (1999). Don't panic ... do it in parallel, *Computational Statistics*, 14(1):53–77.
- Playfair, W. (2005). *Playfair's Commercial and Political Atlas and Statistical Breviary*, Cambridge, London.
- Tukey, J. (1977). *Exploratory Data Analysis*, Addison-Wesley, London.
- Unwin, A. (1999). Requirements for interactive graphics software for exploratory data analysis, *Computational Statistics*, 14:7–22.
- Unwin, A. R., Theus, M. and Hofmann, H. (2006). *Graphics of Large Datasets*, Springer, Berlin.

Graphical Data Representation

in Bankruptcy Analysis

IV.4

Wolfgang K. Härdle, Rouslan A. Moro, Dorothea Schäfer

4.1	<i>Company Rating Methodology</i>	854
4.2	<i>The SVM Approach</i>	857
4.3	<i>Company Score Evaluation</i>	859
4.4	<i>Variable Selection</i>	860
4.5	<i>Conversion of Scores into PDs</i>	865
4.6	<i>Colour Coding</i>	867
4.7	<i>Conclusion</i>	871

Graphical data representation is an important model selection tool in bankruptcy analysis, since this problem is highly nonlinear and its numerical representation is not very transparent. In classical rating models, the convenient representation of the ratings in a closed form reduces the need for graphical tools. In contrast to this, more accurate nonlinear nonparametric models often rely on visualisation. We demonstrate the utilisation of visualisation techniques at different stages of corporate default analysis, which is based on the application of support vector machines (SVM). These stages are the selection of variables (predictors), probability of default (PD) estimation, and the representation of PDs for two- and higher dimensional models with colour coding. The selection of a proper colour scheme becomes crucial to the correct visualisation of PDs at this stage. The mapping of scores into PDs is done as a nonparametric regression with monotonisation. The SVM learns a nonparametric score function that is, in turn, nonparametrically transformed into PDs. Since PDs cannot be represented in a closed form, other ways of displaying them must be found. Graphical tools make this possible.

4.1 Company Rating Methodology

Statistical techniques were first applied to corporate bankruptcy in the 1960s due to the advent of computers. The first technique introduced was discriminant analysis (DA) for univariate (Beaver, 1966) and multivariate models (Altman, 1968). The logit and probit models were then introduced in (Martin, 1977) and (Ohlson, 1980). These models are now widely used in practice – they are at the core of the rating solutions used by most European central banks. The solution in the traditional framework is a linear function (a hyperplane in a multidimensional feature space) that separates successful and failing companies. A company score is computed as a value of that function. In the probit and logit models, the score can be transformed directly into a probability of default (PD), which denotes the probability of a company going bankrupt within a certain period. The major disadvantages of these popular approaches is the linearity of the solution and, in the logit and probit models, the prespecified form of the link function between the PDs and the linear combination of predictors (Fig. 4.1).

In Fig. 4.1, successful and failing companies are denoted by black triangles and white quadrangles, respectively. Both classes contain the same number of companies in the sample. According to the DA and logit classification rules, which give virtually the same results, we are more likely to find a failing company above and to the right of the straight line. This may lead to the conclusion that companies with significantly negative values of operating profit margin and equity ratio can be classified as being successful. This, for example, means that companies with liabilities that far outweigh their total assets can be classified as successful. Such a situation is avoided through the use of a nonlinear classification method, such as support vector machines (SVM), which produces a nonlinear boundary.

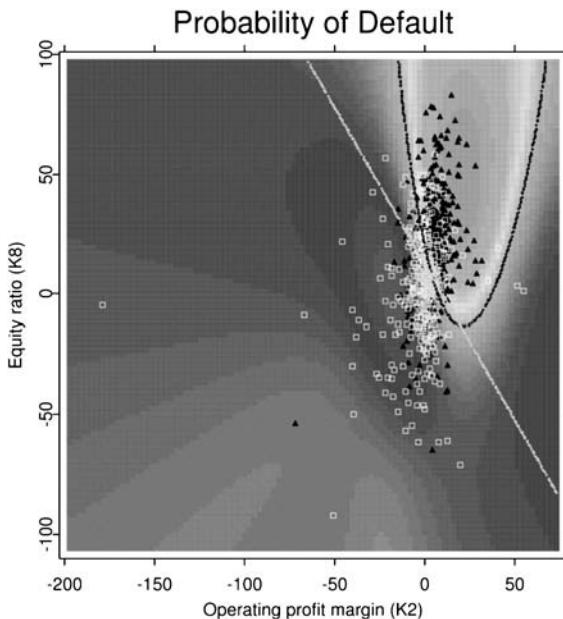


Figure 4.1. A classification example. The boundary between the classes of solvent (black triangles) and insolvent (white squares) companies was estimated using DA, the logit regression (two indistinguishable linear boundaries) and an SVM (a nonlinear boundary) for a subsample of the Bundesbank data. The background corresponds to the PDs computed with an SVM

Following a traditional approach, we would expect a monotonic relationship between predictors and PDs, like the falling relation for the interest coverage ratio (Fig. 4.2). However, in reality this dependence is often nonmonotonic for important indicators such as the company size or net income change. In the latter case, companies that grow too quickly or slowly have a higher probability of default. This is why nonlinear techniques are considered as alternatives. Two prominent examples are recursive partitioning (Frydman et al., 1985) and neural networks (Tam and Kiang, 1992). Despite the strengths of these two approaches they also have noticeable drawbacks: orthogonal division of the data space for recursive partitioning, which is usually not justified, and heuristic model specification for neural networks.

Recursive partitioning, also known as classification and regression trees (CART) (Breiman, 1984), performs classification by dividing up the data space orthogonally. A division (split) along just one of the axes is possible at each step. The axis is chosen such that a split along it reduces the misclassification impurity. Entropy-based criteria can also be used. The visible drawback is the orthogonal division itself, which imposes severe restrictions on the smoothness of the classifying function and may not adequately capture the correlation structure between the variables. Orthogonal division means that the separating hyperplane can only consist of orthogonal segments parallel to the coordinate grid, whereas the boundary between the classes has a smoothly changing gradient.

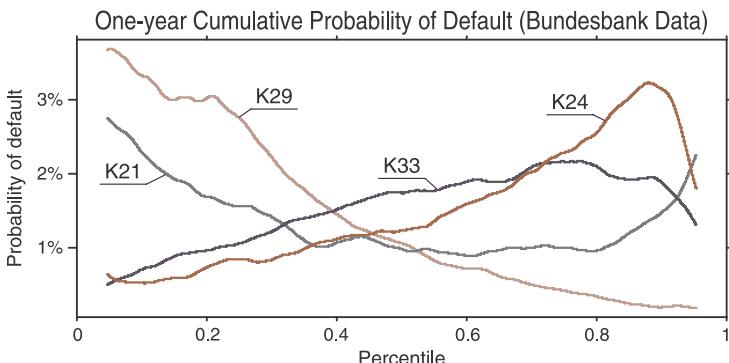


Figure 4.2. One-year cumulative PDs evaluated for several financial ratios from the German Bundesbank data. The ratios are net income change (K21), net interest ratio (K24), interest coverage ratio (K29), and logarithm of total assets (K33). The k nearest neighbors procedure was used with a window size of around 8 % of all of the observations. The total number of observations is 553 500

The neural network (NN) (Rosenblatt, 1962; Minsky and Papert, 1969) is a network of linear classifiers (neurons) that are connected to one another in a prespecified way. The outputs of some of the neurons are inputs for others. The performance of a NN greatly depends on its structure, which must be adapted to solve different problems. The network must be designed manually, which requires substantial operator experience. Moreover, NNs do not usually provide a global solution, only a local one that is valid for some range of variables. This feature and the many heuristics involved make NNs difficult to use in the rating departments of banks.

We would like to have a model that is able to select a classifying function based on very general criteria. The SVM is a statistical technique that in many applications, such as optical character recognition and medical diagnostics, has shown very good performance. It has a flexible solution and is controlled by adjusting only a few parameters. Its overall good performance and flexibility make the SVM a suitable candidate (Härdle et al., 2004a).

Within a rating methodology, each company is described by a set of variables x , such as financial ratios. Financial ratios, such as debt ratio (leverage) or interest coverage (earnings before interest and taxes to interest) characterise different sides of company operation. They are constructed based on balance sheets and income statements. For example, the Bundesbank uses 32 ratios (predictors) computed using the company statements from its corporate bankruptcy database. The predictors and basic statistics are given in Table 4.1. The whole Bundesbank database covers the period 1987–2005 and consists of 553 500 anonymised statements of solvent and insolvent companies. Most companies appear in the database several times in different years.

The class y of a company can be either $y = -1$ ("successful") or $y = 1$ ("bankrupt"). Initially, an unknown classifier function $f : x \rightarrow y$ is estimated for a training set of companies (x_i, y_i) , $i = 1, \dots, n$. The training set represents the data for companies that are known to have survived or gone bankrupt. In order to obtain PDs from the estimated scores f , financial analysts rely on verbally defined classes such as CCC,

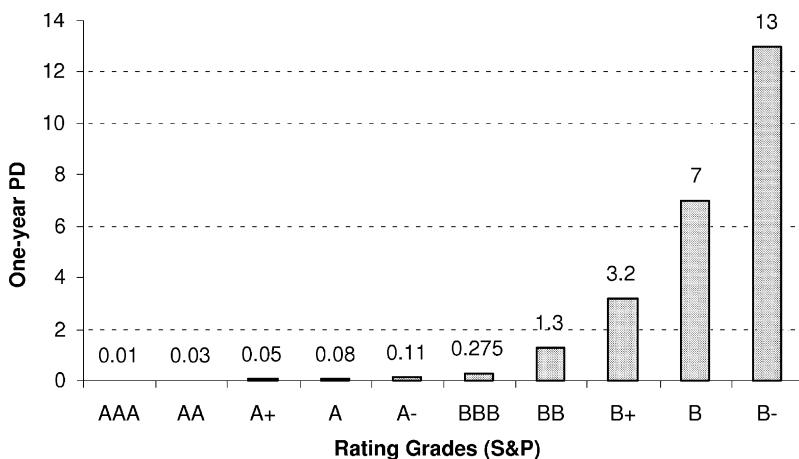


Figure 4.3. One-year probabilities of default for different rating grades (Füser, 2002)

A, or BB. Each company is placed into a specific class depending on how well it fits the description. For example, the AAA grade is associated with extremely strong, AA with very strong, A with strong, BBB with good, BB with marginal, B with weak, CCC with very weak and CC with extremely weak financial security characteristics, while C signals imminent default, and D default.

A certain range of scores and PDs belongs to each rating class. The ranges were computed on the basis of historical data. To derive a PD for a newly scored company, its score f is compared with the historical values of the f values for each class. Based on how similar the scores are, the company is assigned to one particular class. The PD of this class becomes the PD of the company.

Company bond ratings play an important role in determining the cost of debt refinancing, since they reflect the probability of defaulting on the debt (Fig. 4.3). Note that the differences between the classes in terms of PDs are not the same. For example, the PD increases by 6.7 % or 24-fold between classes BBB and B, but only by 0.07 or 8-fold between classes AAA und A. The colours used to code PDs must be selected so that the classes appear to be equidistant, no matter what the absolute PD is. This can be achieved by using an appropriate colour scheme and colour distance scaling. The use of the HLS colour scheme in combination with a logarithmic colour scaling will be demonstrated in Sect. 4.6.

The SVM Approach

4.2

The SVM (Vapnik, 1995) is a classification and regression (which is not applied in this case) technique that is based on margin maximisation (Fig. 4.4) between two data classes. The margin is the region between the hyperplanes bounding each class where no observation can lie in a linear perfectly separable case. The classifier function used

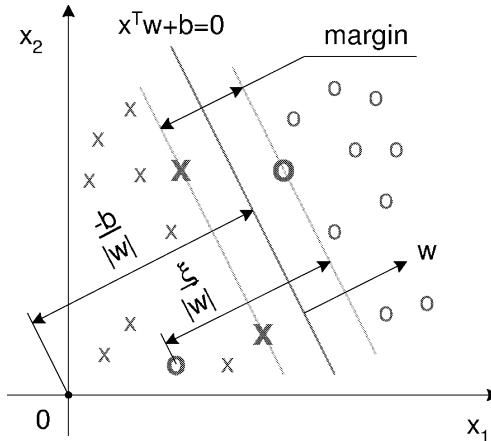


Figure 4.4. The separating hyperplane $x^T w + b = 0$ and the margin in an inseparable case. The observations marked with bold crosses and zeros are support vectors. The hyperplanes bounding the margin zone (which are equidistant from the separating hyperplane) are represented as $x^T w + b = 1$ and $x^T w + b = -1$

by the linear SVM is a hyperplane symmetrically surrounded by a margin zone. It can be shown (Härdle et al., 2004a) that the complexity of such a classifier can be reduced by maximizing the margin. By applying kernel techniques, the SVM can be extended to learn nonlinear classifying functions (Fig. 4.5).

In Fig. 4.4, misclassifications are unavoidable when linear classifying functions are used (linearly inseparable case). To account for misclassifications, the penalty ξ_i is introduced, which is related to the distance from the hyperplane bounding observations of the same class to observation i . $\xi_i > 0$ if a misclassification occurs. All observations satisfy the following two constraints:

$$y_i(x_i^T w + b) \geq 1 - \xi_i, \quad (4.1)$$

$$\xi_i \geq 0. \quad (4.2)$$

With the normalisation of w , b and ξ_i as shown in (4.1), the margin equals $2/\|w\|$. The convex objective function to be minimised given the constraints (4.1) and (4.2) is:

$$\frac{1}{2} \|w\|^2 + C \sum_{i=1}^n \xi_i. \quad (4.3)$$

The parameter C , called the capacity, is related to the width of the margin zone. Larger margins become possible as the value of C decreases. Using a well-established theory for the optimisation of convex functions (Gale et al., 1951), we can derive the dual Lagrangian

$$L_D = \frac{1}{2} w(\alpha)^T w(\alpha) - \sum_{i=1}^n \alpha_i - \sum_{i=1}^n \delta_i \alpha_i + \sum_{i=1}^n \gamma_i (\alpha_i - C) - \beta \sum_{i=1}^n \alpha_i \gamma_i \quad (4.4)$$

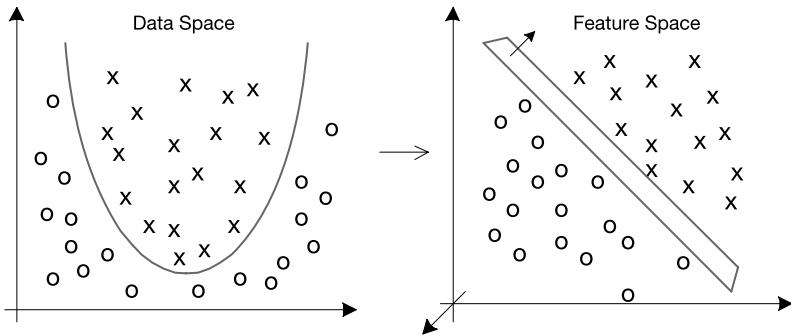


Figure 4.5. Mapping from a two-dimensional data space to a three-dimensional space of features $\mathbb{R}^2 \mapsto \mathbb{R}^3$ using a quadratic kernel function $K(x_i, x_j) = (x_i^\top x_j)^2$. The three features correspond to the three components of a quadratic form: $\tilde{x}_1 = x_1^2$, $\tilde{x}_2 = \sqrt{2}x_1x_2$, and $\tilde{x}_3 = x_2^2$. The transformation is thus $\Psi(x_1, x_2) = (x_1^2, \sqrt{2}x_1x_2, x_2^2)^\top$. The data that are separable in the data space with a quadratic function will be separable in the feature space with a linear function. A nonlinear SVM in the data space is equivalent to a linear SVM in the feature space. The number of features will grow rapidly with the dimensionality d and the degree of the polynomial kernel p , which is 2 in our example, making the closed-form representation of Ψ such as that shown here practically impossible

for the dual problem:

$$\min_{\alpha_i, \delta_i, y_i, \beta} \max_{w_k, b, \xi_i} L_D. \quad (4.5)$$

Here, for a linear SVM,

$$w(\alpha)^\top w(\alpha) = \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j x_i^\top x_j. \quad (4.6)$$

A more general form is applicable in order to obtain nonlinear classifying functions in the data space:

$$w(\alpha)^\top w(\alpha) = \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j K(x_i, x_j). \quad (4.7)$$

The function $K(x_i, x_j)$ is called a kernel function. Since it has a closed form representation, the kernel is a convenient way of mapping low-dimensional data into a highly dimensional (often infinitely dimensional) space of features. It must satisfy the Mercer conditions (Mercer, 1909), i.e. it must be symmetric and semipositive definite; in other words it must represent a scalar product in some Hilbert space (Weyl, 1928).

In our study, we applied an SVM with an anisotropic Gaussian kernel

$$K(x_i, x_j) = \exp \left\{ -(x_i - x_j)^\top r^{-2} \Sigma^{-1} (x_i - x_j) / 2 \right\}, \quad (4.8)$$

where r is a coefficient and Σ is a variance–covariance matrix. The coefficient r is related to the classifying function complexity: as r increases, the complexity drops. If

kernel functions allow for sufficiently rich feature spaces, the performances of SVMs are comparable in terms of out-of-sample forecasting accuracy (Vapnik, 1995).

4.3

Company Score Evaluation

The company score is computed as:

$$f(x) = x^\top w + b, \quad (4.9)$$

where $w = \sum_{i=1}^n \alpha_i y_i x_i$ and $b = \frac{1}{2}(x_+ + x_-)^\top w$; x_+ and x_- are the observations from the opposite classes for which constraint (4.1) becomes equality. By substituting the scalar product with a kernel function, we will derive a nonlinear score function:

$$f(x) = \sum_{i=1}^n K(x_i, x) \alpha_i y_i + b. \quad (4.10)$$

The nonparametric score function (4.10) does not have a compact closed form representation. This means that graphical tools are required to visualise it.

4.4

Variable Selection

In this section we describe the procedure and the graphical tools for selecting the variables of the SVM model used in forecasts. We have two very important model accuracy criteria: the accuracy ratio (AR), which will be used here as a criterion for model selection (Fig. 4.6), and the percentage of correctly classified out-of-sample observations. Higher values indicate better model accuracy.

Model selection proceeds from the simplest (i.e. univariate) models to the one with the highest AR. The problem that arises is: how do we determine the variable that provides the highest AR across possible data samples? For a parametric model, we would need to estimate the distribution of the coefficients at the variables and therefore their confidence intervals. This approach, however, is practically irrelevant for nonparametric models.

Instead we can compare models using an accuracy measure, in our case AR. We first estimate the AR distributions for different models. This can be done using bootstrapping (Horowitz, 2001). We randomly select training and validation sets, each of which is a subsample of 500 solvent and 500 insolvent companies. We use a 50/50 ratio since this is the worst case with the minimum AR. The two sets do not overlap – they do not contain common observations. For each of these sets we apply the SVM with parameters that provide the highest AR for bivariate models (Fig. 4.7) and estimate the ARs. Then we perform a Monte Carlo experiment: we repeat this process of generating subsamples and computing ARs 100 times. Each time we will record the ARs and then estimate their distribution.

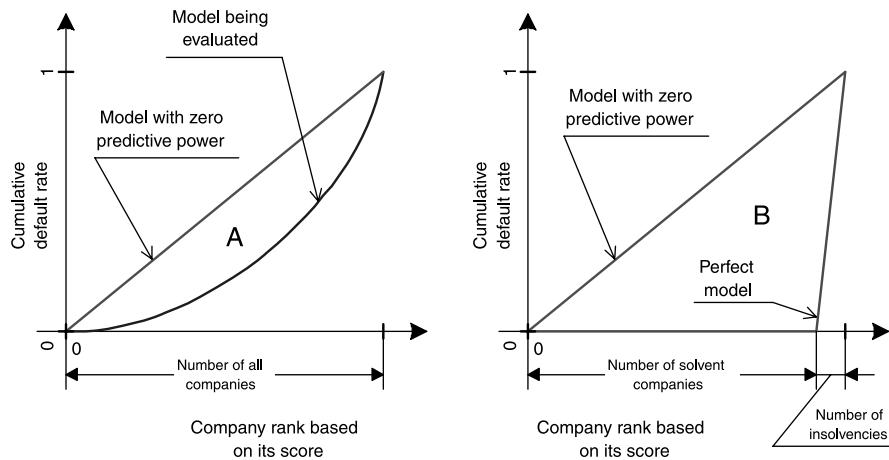


Figure 4.6. The power curves for a perfect model, a random model, and some real classification models. AR is the ratio of the two areas (A/B). It lies between 0 (a random model with no predictive power) and 1 (a perfect model)

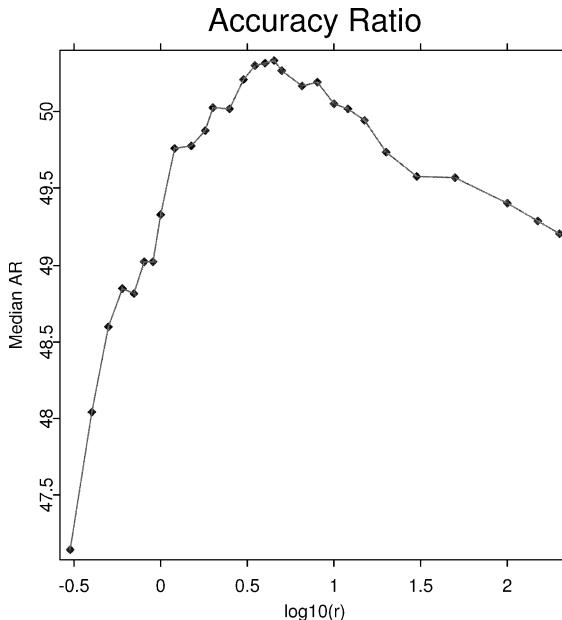


Figure 4.7. The relationship between an accuracy measure (AR) and the coefficient r in the SVM formulation. Higher r values correspond to less complex models. The median ARs were estimated for training and validation sets of 100 bootstrapped subsamples each of 500 solvent and 500 insolvent companies. A bivariate SVM was used with the variables K5 and K29. We will be using $r = 4$ in all SVMs discussed in this chapter

At the end of this procedure, we obtain an empirically estimated AR distribution for the bootstrapped subsamples. The median AR provides a robust measure that can be used to compare different variables used as predictors. The same approach can be used to compare SVM with DA and logit regression in terms of predictive power. We compute AR for the same subsamples with the SVM, DA, and logit models. The median improvements in AR for the SVM over DA and the SVM over the logistic regression are also reported below.

We will start this procedure with univariate models for 33 variables, K1–K9 and K11–K33 (as they are denoted by the Bundesbank) and variable K10, which is a standard normal random variable used as a reference (Table 4.1). For each model, the resulting distribution of ARs will be represented as box plots (Fig. 4.8). The broken

Table 4.1. Summary statistics for the Bundesbank data. q_α is an α quantile. IQR is the interquartile range

Var.	Name	Group	$q_{0.01}$	Median	$q_{0.99}$	IQR
K1	Pre-tax profit margin	Profitability	-26.9	2.3	78.5	5.9
K2	Operating profit margin	Profitability	-24.6	3.8	64.8	6.3
K3	Cash flow ratio	Liquidity	-22.6	5.0	120.7	9.4
K4	Capital recovery ratio	Liquidity	-24.4	11.0	85.1	17.1
K5	Debt cover	Liquidity	-42.0	17.1	507.8	34.8
K6	Days receivable	Activity	0.0	31.1	184.0	32.7
K7	Days payable	Activity	0.0	23.2	248.2	33.2
K8	Equity ratio	Financing	0.3	14.2	82.0	21.4
K9	Equity ratio (adj.)	Financing	0.5	19.3	86.0	26.2
K10	Random Variable	Test	-2.3	0.0	2.3	1.4
K11	Net income ratio	Profitability	-29.2	2.3	76.5	5.9
K12	Leverage ratio	Leverage	0.0	0.0	164.3	4.1
K13	Debt ratio	Liquidity	-54.8	1.0	80.5	21.6
K14	Liquidity ratio	Liquidity	0.0	2.0	47.9	7.1
K15	Liquidity 1	Liquidity	0.0	3.8	184.4	14.8
K16	Liquidity 2	Liquidity	2.7	63.5	503.2	58.3
K17	Liquidity 3	Liquidity	8.4	116.9	696.2	60.8
K18	Short term debt ratio	Financing	2.4	47.8	95.3	38.4
K19	Inventories ratio	Investment	0.0	28.0	83.3	34.3
K20	Fixed assets ownership r.	Leverage	1.1	60.6	3750.0	110.3
K21	Net income change	Growth	-50.6	3.9	165.6	20.1
K22	Own funds yield	Profitability	-510.5	32.7	1998.5	81.9
K23	Capital yield	Profitability	-16.7	8.4	63.1	11.0
K24	Net interest ratio	Cost struct.	-3.7	1.1	36.0	1.9
K25	Own funds/pension prov. r.	Financing	0.4	17.6	84.0	25.4
K26	Tangible asset growth	Growth	0.0	24.2	108.5	32.6
K27	Own funds/provisions ratio	Financing	1.7	24.7	89.6	30.0
K28	Tangible asset retirement	Growth	1.0	21.8	77.8	18.1
K29	Interest coverage ratio	Cost struct.	-1338.6	159.0	34350.0	563.2
K30	Cash flow ratio	Liquidity	-14.1	5.2	116.4	8.9
K31	Days of inventories	Activity	0.0	42.9	342.0	55.8
K32	Current liabilities ratio	Financing	0.3	58.4	98.5	48.4
K33	Log of total assets	Other	4.9	7.9	13.0	2.1

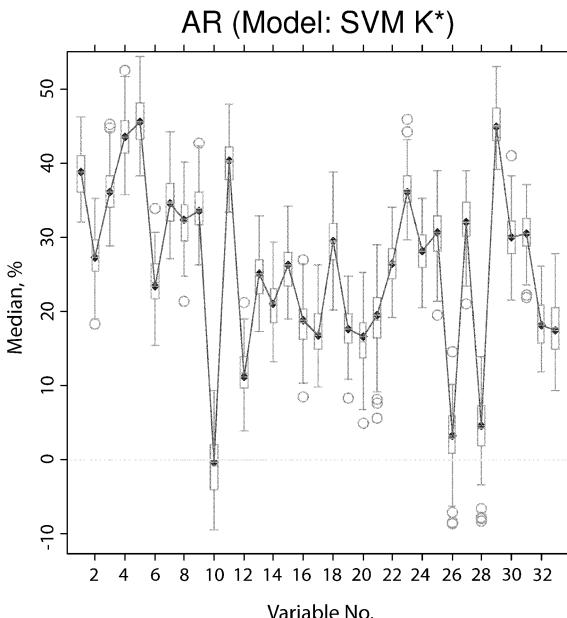


Figure 4.8. Accuracy ratios for univariate SVM models. Box plots are estimated based on 100 random subsamples. The AR for the model containing only the random variable K10 is zero

line depicts medians. The box within each box plot shows the interquartile range (IQR), while the whiskers span to the distance of $3/2$ IQR in each direction from the median. Outliers beyond that range are denoted by circles.

Based on Fig. 4.8, we can conclude that variables K5 (Debt Cover) and K29 (Interest Coverage Ratio) provide the highest median AR, of around 50 %. We also notice that the variables K12, K26, and K28 yield very low accuracy: their median ARs do not exceed 11.5 %. The model based on the random variable K10 has an AR of zero; in other words it has no predictive power whatsoever. For the next step we will select variable K5, which was included in the best univariate model.

For bivariate models, we will select the best predictor from the univariate models (K5) and one of the rest that delivers the highest AR (K29) (Fig. 4.9). This procedure will be repeated for each new variable added. The AR grows until the model has eight variables, and then it slowly declines. Median ARs for the models with eight variables are shown in Fig. 4.10. The forward selection procedure cannot guarantee that the variables selected will provide the highest accuracy. However, since many of them are highly correlated, we can expect that the selected variables capture most of the information.

We have also conducted experiments with subsamples of 5000 observations. The change in the median was extremely small (1–2 orders of magnitude smaller than the interquartile range). As expected, the interquartile range narrowed, i.e. the difference between the models with more samples became more statistically significant. Thus, if

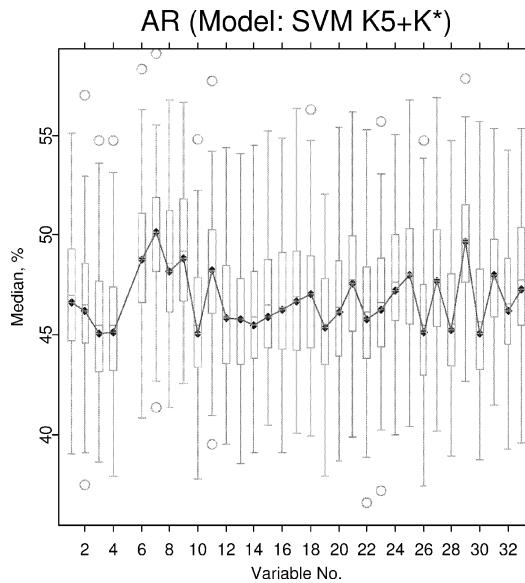


Figure 4.9. Accuracy ratios for bivariate SVM models. Each model includes variable K5 and one of the remaining variables. Box plots are estimated based on 100 random subsamples

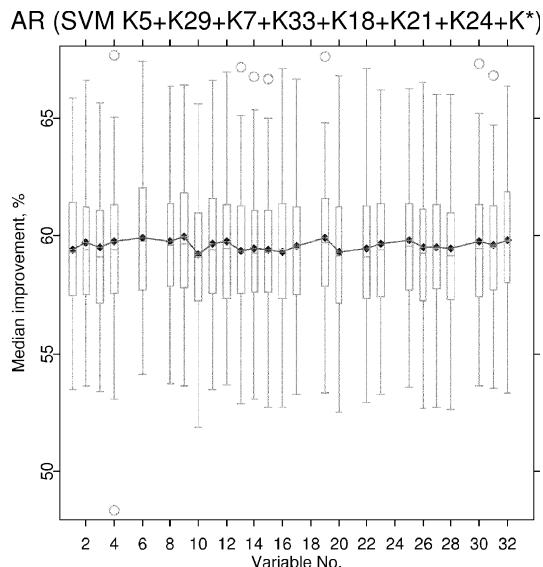


Figure 4.10. Accuracy ratios for SVM models with eight variables. Each model includes the variables K5, K29, K7, K33, K18, K21, K24, and one of the remaining variables. Box plots are estimated based on 100 random subsamples

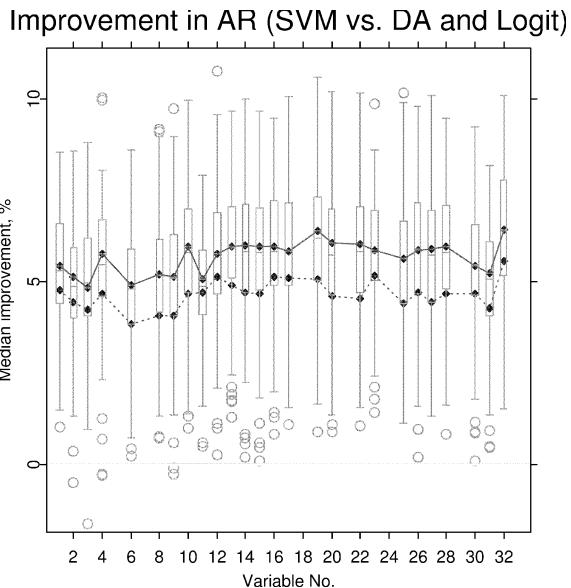


Figure 4.11. Median improvement in AR. SVM vs. DA (*upper line*) and SVM vs. logit regression (*lower line*). Box plots are estimated based on 100 random subsamples for the case of DA. Each model includes variables K5, K29, K7, K33, K18, K21, K24, and one of the remaining variables

the SVM significantly outperforms DA and the logit regression for a sample of 1000 observations, it will also do so for larger samples.

The SVM based on the variables K5, K29, K7, K33, K18, K21, K24, K33, and K9 attains the highest median AR, around 60.0 %. For comparison we plot the improvement in AR for the SVM vs. DA and logit regression for the same 100 subsamples. The data used in the DA and logit models were processed as follows: if $x_i < q_{0.05}(x_i)$ then $x_i = q_{0.05}(x)$, and if $x_i > q_{0.95}(x_i)$ then $x_i = q_{0.95}(x_i)$; $i = 1, 2, \dots, 8$; $q_\alpha(x_i)$ is an α quantile of x_i . Thus, the DA and logit regressions applied were robust versions that were not sensitive to outliers. Without such a procedure the improvement would be much higher.

Figure 4.11 represents the absolute improvement obtained by SVM over DA (upper line) and SVM over logit regression (lower line). We can see that, for all models containing the variables K5, K29, K7, K33, K18, K21, K24 and one of the remaining variables, the median AR was always higher for the SVM. This means that the SVM model always outperforms DA and logit regressions with regard to AR.

Conversion of Scores into PDs

There is another way to look at the company score. It defines the distance between companies in terms of the distance to the boundary between the classes. The lower

the score, the farther the company is from the class of bankrupt companies, and so (we can assume) the lower its PD is. This means that the dependence between scores and PDs is assumed to be monotonous. This is the only kind of dependence that was assumed in all rating models mentioned in this chapter, and the only one we use for the PD calibration.

The conversion procedure consists of the estimation of PDs for the observations in the training set with a subsequent monotonisation (steps one and two) and then the computation of the PD for a new company (step three).

Step one is the estimation of PDs for the companies in the training set. We used kernel techniques to perform a preliminary evaluation of the PD for observation i from the training set, $i = 1, 2, \dots, n$:

$$\widetilde{PD}(x_i) = \frac{\sum_{j=1}^n K_h(x_i, x_j) I_{\{y_j=1\}}}{\sum_{j=1}^n K_h(x_i, x_j)}. \quad (4.11)$$

A k nearest neighbor Gaussian kernel was used here. h is the kernel bandwidth.

The preliminary PDs evaluated in this way are not necessarily a monotonic function of the score. The monotonisation of \widetilde{PD}_i , $i = 1, 2, \dots, n$ is achieved at step two using the pool adjacent violators (PAV) algorithm (Barlow et al. (1972) and Mam-

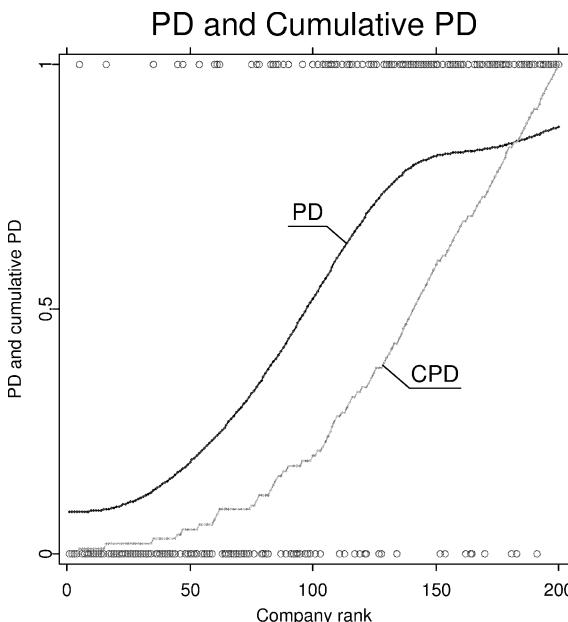


Figure 4.12. PD and cumulative PD estimated with the SVM for a subsample of 200 observations from the Bundesbank data. The variables included in the model were those for which the higher AR achieved: K5, K29, K7, K33, K18, K21, K24 and K9. The higher the score, the higher the rank of the company

men (1991)). As a result we obtain monotonised probabilities of default $PD(x_i)$ for the observations in the training set.

Finally, at step three the PDs are computed for any observation described by x as an interpolation between the two PDs of neighboring (in terms of the score) observations in the training set, x_i and x_{i-1} , $i = 2, 3, \dots, n$:

$$PD(x) = PD(x_i) + \frac{f(x) - f(x_{i-1})}{f(x_i) - f(x_{i-1})} \{PD(x_i) - PD(x_{i-1})\}. \quad (4.12)$$

If the score for an observation x lies beyond the range of scores for the training set, then $PD(x)$ equals the score of the first neighboring observation of the training set.

Figure 4.12 is an example of the cumulative PD curve (power curve) and estimated PDs for a subsample of 200 companies. The PD curve has a plateau area for observations with a high score. Default probabilities can change from 15 % to 80 % depending on the score.

Colour Coding

4.6

The RGB colour space is based on three primary colours, red, green and blue, that are mixed to produce others. This is the colour coding scheme that is used in monitors and TVs. It is, however, an inconvenient colour coding scheme here since we only wish to adjust the channel responsible for colour while keeping lightness and saturation constant. This can be achieved with the HLS colour space.

We will represent the probability of default (PD) estimated with the SVM using two-dimensional plots where each colour denotes a specific PD. The PD is a number that can be represented on a greyscale. For example, in the RGB encoding as (i, i, i) where i is in the range 0 to 255 (e.g. the colour $R=255, G=0, B=0$ corresponds to red, and $R=255, G=0, B=255$ to violet, etc.).

HLS stands for hue, lightness (or luminance) and saturation. By adjusting only the hue and keeping the luminance and saturation fixed, we can generate simulated colours from the range shown in Fig. 4.14. A pure red colour corresponds to $H=0$ or 360, a pure green to $H=120$, and a pure blue to $H=240$.

Red is often used in finance to highlight negative information, while green and blue are used to convey positive information. Therefore, we would like to code PDs with colours that range from red for the highest PD to blue-green for the most solvent company. We therefore normalise the PDs such that the lowest PD has a hue of 180 (green-blue) or 120 (green) while the highest PD has a hue of 0 (red). The resulting graphs that show the data and PDs in the dimensions of variables K33 and K29 are shown in Figs. 4.15–4.17.

The SVM with the radial basis parameter $r = 4$ provides the highest AR (Fig. 4.16). The near-linear SVM, $r = 100$, only uses almost linear classification functions and has a lower classification power due to this limitation (Fig. 4.17). The SVM with an excessively high complexity, $r = 0.5$, suffers from overfitting and also has a lower prediction accuracy (Fig. 4.15).

The three figures therefore correspond to three SVMs with high, average and high complexity. The saturation was fixed at 0.85 to make the colours look more natural and soft, and the luminance was fixed at 0.46, the maximum possible value for the chosen saturation. The HLS colours obtained in this way were transformed into RGB ones and plotted by XploRe (Härdle et al., 2000b,a) as a contour plot. The outliers that lie beyond the 5 % and 95 % quantiles are plotted at the edge.

To produce the plot, a grid was generated with 101 steps in both the horizontal and vertical directions. For each point of the grid, a PD was estimated and represented in the HLS colour space. Then the HLS colour was converted into an RGB colour

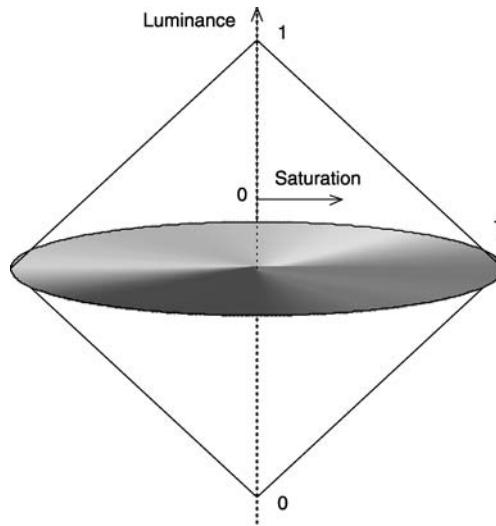


Figure 4.13. [This figure also appears in the color insert.] The luminance and saturation dimensions of the HLS colour space. We will keep luminance and saturation constant and encode PD information with the hue

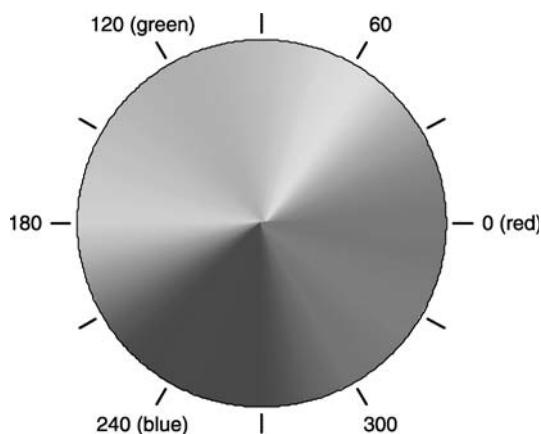


Figure 4.14. [This figure also appears in the color insert.] The hue dimension of the HLS colour space

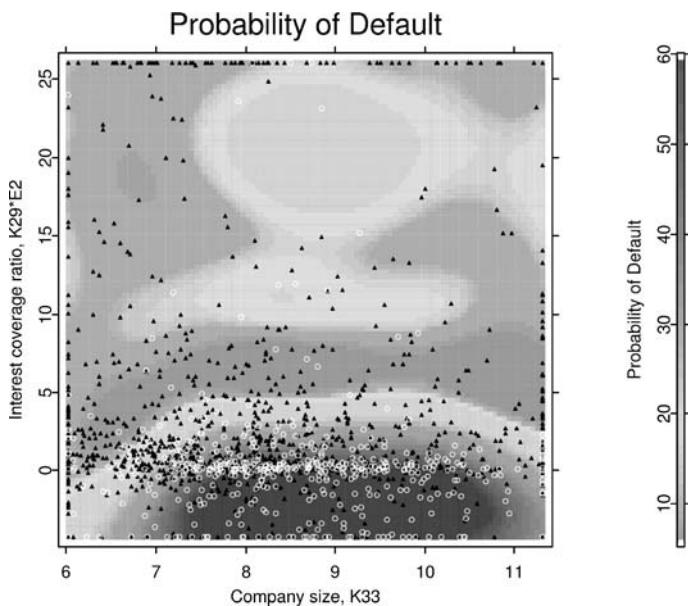


Figure 4.15. Probability of default, estimated for a random subsample of 500 failing and 500 surviving companies, and plotted for the variables K33 and K29. An SVM of high complexity with a radial basis kernel of $0.5\Sigma^{1/2}$ was used

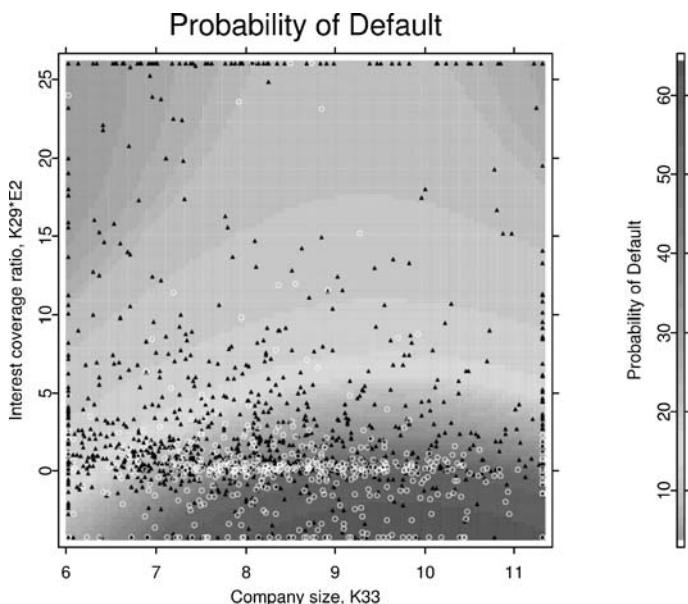


Figure 4.16. Probability of default, estimated for a random subsample of 500 failing and 500 surviving companies, and plotted for the variables K33 and K29. An SVM of average complexity with a radial basis kernel of $4\Sigma^{1/2}$ was used

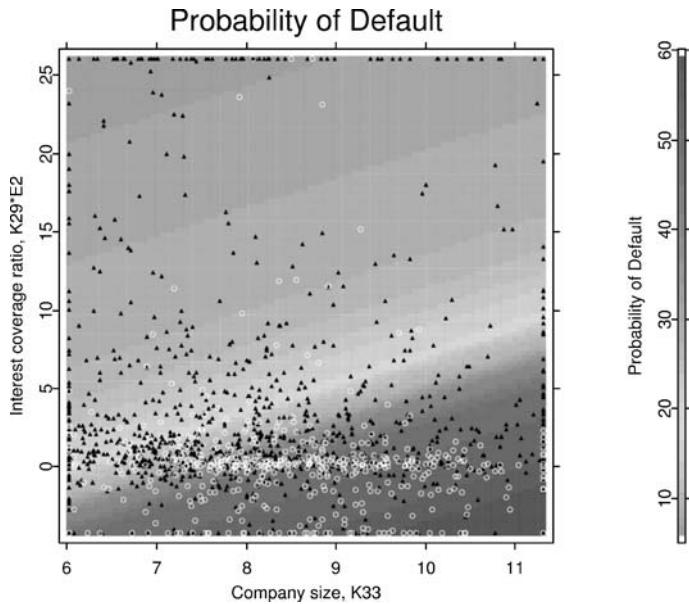


Figure 4.17. Probability of default, estimated for a random subsample of 500 failing and 500 surviving companies, and plotted for the variables K33 and K29. An SVM of low complexity with a radial basis kernel of $100\Sigma^{1/2}$ was used

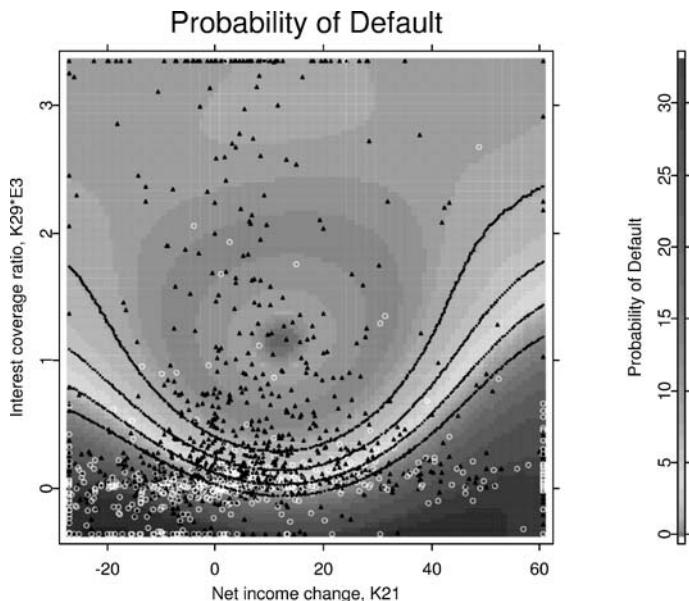


Figure 4.18. Probability of default, plotted for the variables K21 and K29. The boundaries of five risk classes are shown, which correspond to the rating classes BBB and above (investment grade), BB, B+, B, B- and lower

and plotted by XploRe as a small filled quadrangle. The quadrangles evenly cover the whole area, giving a continuous PD representation. Contour lines can also be added to the graph, as illustrated in Fig. 4.18.

Conclusion

In this chapter we demonstrated the application of graphical tools to variable selection, data visualisation and financial information representation, and we discussed essential aspects of graphical analysis, such as colour coding. We believe that graphical analysis will become increasingly important as nonparametric models, such as SVM, become more and more popular. On the other hand, graphical representation can aid the acceptance of nonparametric models in various areas, such as finance, medicine, and sound and image processing. The application of graphical representation will contribute to the development of these areas, since nonlinear nonparametric models are better at representing reality and provide higher forecasting accuracy.

Acknowledgement. The authors are grateful to the Deutsche Bundesbank for providing access to the unique database of the financial statements of German companies. The data analysis took place on the premises of the Deutsche Bundesbank in Frankfurt. The work of R.A. Moro was financially supported by the German Academic Exchange Service (DAAD) and Deutsche Bundesbank. We also acknowledge the support of the German Research Foundation (DFG) through the SFB 649 of Humboldt-Universität, Berlin, in writing this paper.

References

- Altman, E. (1968). Financial Ratios, Discriminant Analysis and the Prediction of Corporate Bankruptcy. *The Journal of Finance*, 23(4):589–609.
- Barlow, R.E., Bartholomew, J.M., Bremmer, J.M. and Brunk, H.D. (1972). *Statistical Inference Under Order Restrictions*. Wiley, New York, NY.
- Beaver, W. (1966). Financial Ratios as Predictors of Failures. Empirical Research in Accounting: Selected Studies. *Journal of Accounting Research*, 5(Suppl.):71–111.
- Breiman, L. (1984). *Classification and Regression Trees*. Chapman & Hall/CRC, New York.
- Friedman, H., Altman, E. and Kao, D.-L. (1985). Introducing Recursive Partitioning for Financial Classification: The Case of Financial Distress. *The Journal of Finance*, 40(1):269–291.
- Füser, K. (2002). *Basel II – was muß der Mittelstand tun?* A presentation at Ernst & Young.
- Gale, D., Kuhn, H.W. and Tucker, A.W. (1951). In: Koopmans, T.C. (ed) *Linear Programming and the Theory of Games, in Activity Analysis of Production and Allocation*. Wiley, New York, NY.
- Härdle, W., Hlavka, Z. and Klinke, S. (2000a). *XploRe Application Guide*. Springer, Berlin.

- Härdle, W., Klinke, S. and Müller, M. (2000b). *XploRe Learning Guide*. Springer, Berlin.
- Härdle, W., Moro, R.A. and Schäfer, D. (2004a). *Predicting Bankruptcy with Support Vector Machines*. Springer, Berlin.
- Härdle, W., Müller, M., Sperlich, S. and Werwatz, A. (2004b). *Nonparametric and Semiparametric Models*. Springer, Berlin.
- Härdle, W. and Simar, L. (2003). *Applied Multivariate Statistical Analysis*. Springer, Berlin.
- Horowitz, J.L. (2001). In: Heckman, J.J. and Leamer, E.E. (eds). *The Bootstrap*, Vol. 5. Elsevier, Amsterdam.
- Mammen, E. (1991). Estimating a Smooth Monotone Regression Function. *Annals of Statistics*, 19:724–740.
- Martin, D. (1977). Early Warning of Bank Failure: A Logit Regression Approach. *Journal of Banking and Finance*, 1:249–276.
- Mercer, J. (1909). Functions of Positive and Negative Type and Their Connection with the Theory of Integral Equations. *Philosophical Transactions of the Royal Society of London*, 209:415–446.
- Minsky, M. and Papert, S. (1969). *Perceptrons, An Introduction to Computational Geometry*. MIT Press, Cambridge, MA.
- Ohlson, J. (1980). Financial Ratios and the Probabilistic Prediction of Bankruptcy. *Journal of Accounting Research*, 18:109–131.
- Rosenblatt, F. (1962). *Principles of Neurodynamics*. Spartan, New York.
- Tam, K. and Kiang, M. (1992). Managerial Application of Neural Networks: the Case of Bank Failure Prediction. *Management Science*, 38(7):926–947.
- Vapnik, V.N. (1995). *The Nature of Statistical Learning Theory*. Springer, New York.
- Weyl, H. (1928). *Gruppentheorie und Quantenmechanik*. Hirzel, Leipzig.

Visualizing Functional Data with an Application to eBay's Online Auctions

IV.5

Wolfgang Jank, Galit Shmueli, Catherine Plaisant, Ben Shneiderman

5.1	<i>Introduction</i>	874
5.2	<i>Online Auction Data from eBay</i>	876
5.3	<i>Visualization at the Object Recovery Stage</i>	877
5.4	<i>Visualizing Functional Observations</i>	882
	Visualizing Individual Objects and Their Dynamics	882
	Visualizing Relationships Among Functional Data	886
	Visualizing Functional and Cross-sectional Information.....	887
5.5	<i>Interactive Information Visualization of Functional and Cross-sectional Information via TimeSearcher</i>	890
	Capabilities of TimeSearcher	891
	Forecasting with TimeSearcher.....	894
5.6	<i>Further Challenges and Future Directions</i>	895
	Concurrency of Functional Events.....	897
	Dimensionality of Functional Data	897
	Complex Functional Relationships	897

5.1 Introduction

Technological advances in the measurement, collection, and storage of data have led to more and more complex data structures. Examples of such structures include measurements of the behavior of individuals over time, digitized two- or three-dimensional images of the brain, and recordings of three- or even four-dimensional movements of objects traveling through space and time. Such data, although recorded in a discrete fashion, are usually thought of as continuous objects that are represented by functional relationships. This gave rise to functional data analysis (FDA), which was made popular by the monographs of Ramsay and Silverman (1997, 2002), where the center of interest is a set of curves, shapes, objects, or, more generally, a set of *functional observations*, in contrast to classical statistics where interest centers on a set of data vectors. In that sense, functional data is not only different from the data structure studied in classical statistics, but it actually generalizes it. Many of these new data structures require new statistical methods to unveil the information that they carry.

There are many examples of functional data. The year-round temperature at a weather station can be thought of as a continuous curve, starting in January and ending in December, where the amplitude of the curve signifies the temperature level at each day or at each hour. A collection of temperature curves from different weather stations is then a set of functional data. Similarly, the price during an online auction of a certain product can be represented by a curve, and a sample of multiple auction price curves for the same product is then a set of functional objects. Alternatively, the digitized image of a car passing through a highway toll booth can be described by a two-dimensional curve measuring the pixel color or intensity of that image. A collection of image curves from all of the cars passing through the toll booth during a single day can then be considered to be a set of functional data. Lastly, the movement of a person through time and space can be described by a four- (or even higher) dimensional hyperplane in x -, y -, z - and time coordinates. The collection of all such hyperplanes from people passing through the same space is again a set of functional data.

Data visualization is an important part of any statistical analysis and it serves many different purposes. Visualization is useful for understanding the general structure and nature of the data, such as the types of variables contained in the data (categorical, numerical, text, etc.), their value ranges, and the balance between them. Visualization is useful for detecting missing data, and it can also aid in pinpointing extreme observations and outliers. Moreover, unknown trends and patterns in the data are often uncovered with the help of visualization. After identifying such patterns, they can then be investigated more formally using statistical models. The exact nature of these models (e.g., linear vs. log-linear) is again often based on insight learned from visualization. Finally, model assumptions are typically verified through the visualization of residuals and other model-related variables.

While visualization is an important step in comprehending any set of data, different types of data require different types of visualization. Take for instance the example

of cross-sectional data vs. time series data. While the information in cross-sectional data can often be displayed satisfactorily with the help of standard bar charts, box-plots, histograms or scatter plots, time series data require special graphs that can also capture the temporal information. The methods used to display time-series data range from rather simple time-series plots, to streaming video clips for discrete time series (Mills et al., 2005), to cluster- and calendar-based visualization for more complex representations (van Wijk and van Selow, 1999).

Functional data are different to ordinary data in both structure and concept, and thus require special visualization methods. While the reasons for visualizing functional data are similar to those for ordinary data, functional data entail additional challenges that require extra attention. One such challenge is the creation of functional observations. Functional data are typically obtained by recovering the continuous functional object from the discrete observed data via data-smoothing. The implication of this is that there are two levels to the study of functional data. The first level uses the discrete observed data to recover the continuous functional object. Visualizing data at this level is important for detecting anomalies that are related to the data generation process, such as data collection and data entry, as well as for assessing the fit of the smoothed curves to the discrete observed data. This is illustrated and discussed further in Sect. 5.3. The second and higher level of study operates on the functional objects themselves. Since at this level the functional objects are the observations of interest, visualization is now used for the same reasons mentioned previously for ordinary data: to detect patterns and trends, possible relationships, and also anomalies. In Sect. 5.4 we describe different visualizations that enhance data comprehension and support more formal analyses.

The visualization of functional data has not received much attention in the literature to date. Most of the work in this area has focused on the derivation of mathematical models for functional data, with visualization playing a minor role and typically appearing only as a side product of the analysis. Some noteworthy exceptions include the display of summary statistics, such as the mean and the variability of a set of functional objects, the use of phase-plane plots to understand the interplay of dynamics, and the graphing of functional principal components to study sources of variability within functional data (Ramsay and Silverman, 2002). Another exception is the work of Shmueli and Jank (2005) and Hyde et al. (2005), which is focused directly on the visualization of functional data, and which suggests a few novel ideas for the display of functional data, such as *calendar plots* and *rug plots*.

Most of the visualizations currently used for functional data are static in nature. By “static” we mean a graph that can no longer be modified by the user without re-running a piece of software code after it has been generated. A static approach is useful for differentiating subsets of curves by attributes (e.g., by using color), or for spotting outliers. A static approach, however, does not permit an interactive exploration of the data. By “interactive” we mean that the user can perform operations such as zooming in and out, filtering the data, and obtaining details about the filtered data, all within the environment of the graphical interface. Interactive visualizations that can be used for the special structure of functional data are not straightforward to devise, and so solutions have only recently begun to receive consideration (Aris et al.,

2005; Shmueli et al., 2006). In Sect. 5.5 we describe an interactive visualization tool designed for the display and exploration of functional data. We illustrate its features and benefits using the example of price curves, which capture the price evolution in online auctions.

The insightful display of functional data comes with many, many different challenges, and we are only scraping the tip of the iceberg in this essay. Functional data is challenging due to its high object dimensionality, complex functional relationships and the concurrency present among the functional objects. We discuss some of these extra challenges in Sect. 5.6.

5.2

Online Auction Data from eBay

eBay (www.eBay.com) is one of the major online marketplaces and currently the biggest consumer-to-consumer online auction site. eBay offers a vast amount of rich bidding data. Besides the time and amount of each bid placed, eBay also records plenty of information about the bidders, the seller, and the product being auctioned. On any given day, several million auctions take place on eBay, and all closed auctions from the last 15 days are made publicly available on eBay's website. This huge amount of information can be quite overwhelming and confusing to the user (i.e., either the seller, the potential buyer, or the auction house) that wishes to incorporate this information into his/her decision-making process. Data visualization can help to alleviate this confusion.

Online auctions lend themselves naturally to the use of functional data for a variety of reasons. Online auctions can be conceptualized as a series of bids placed over time. The finite time horizon of the auction allows the study of the price evolution between the start and the end of the auction. By “price evolution” we mean the changes in the price due to new bids as the auction approaches its end. Conceptualizing the price evolution as a continuous price curve allows the researcher to investigate price dynamics via the price curve's first and second derivatives.

It is worth noting that empirical research into online auctions has largely ignored the temporal dimension of the bidding data, and has instead only considered a condensed snapshot of the auction. That is, most research has only considered the end of the auction by, for example, concentrating only on the final price rather than on the entire price curve, or by only looking at the total number of bidders rather than the function describing the bidder arrival process. Considering only the end of the auction results in information loss, since such an approach entirely ignores the way in which that end-point was reached. Functional data analysis is a natural solution that allows us to avoid this information loss. In a recent series of papers, the first two authors have taken a functional approach and shown that pairing the price evolution with its dynamics leads to a better understanding of different auction profiles (Jank and Shmueli, 2005) or to more accurate forecasts of the final auction price (Wang et al., 2005).

Visualization at the Object Recovery Stage

Any functional data set consists of a collection of continuous functional objects, such as a set of continuous curves describing the temperature changes over the course of a year, or the price increases in an online auction. Despite their continuous nature, limitations in human perception and measurement capabilities allow us to observe these curves only at discrete time points. Moreover, the presence of human and measurement error results in discrete observations that are noisy realizations of the underlying continuous curve. Thus, the first step in every functional data analysis is to recover the underlying continuous functional object from the observed data. This is typically done with the help of smoothing techniques.

A variety of different smoothers exist. One very flexible and computationally efficient choice is the penalized smoothing spline (Ruppert et al., 2003). Let τ_1, \dots, τ_L be a set of knots. Then, a polynomial spline of order p is given by

$$f(t) = \beta_0 + \beta_1 t + \beta_2 t^2 + \dots + \beta_p t^p + \sum_{l=1}^L \beta_{pl} (t - \tau_l)_+^p, \quad (5.1)$$

where $u_+ = u I_{[u \geq 0]}$ denotes the positive part of the function u . Define the roughness penalty

$$\text{PEN}_m(t) = \int \{D^m f(t)\}^2 dt, \quad (5.2)$$

where $D^m f$, $m = 1, 2, 3, \dots$, denotes the m th derivative of the function f . The penalized smoothing spline f minimizes the penalized squared error

$$\text{PENSS}_{\lambda, m} = \int \{y(t) - f(t)\}^2 dt + \lambda \text{PEN}_m(t), \quad (5.3)$$

where $y(t)$ denotes the observed data at time t and the smoothing parameter λ controls the trade-off between the data fit and the smoothness of the function f . Using $m = 2$ in (5.3) leads to the commonly encountered cubic smoothing spline. Other possible smoothers include the use of B-splines or radial basis functions (Ruppert et al., 2003).

We want to emphasize that we use a common set of smoothing parameters across all functional objects. For instance, for the penalized smoothing splines, we pick a common set of knots τ_1, \dots, τ_L , a common spline order p , and a common penalizing term λ , and apply this common set of smoothing parameters to all functional objects, $1 \leq i \leq n$. The rationale behind using a common set is that it allows us to make comparisons among the individual functional objects. Conversely, if one were to use, say, a large value of λ for object i but a small value for object i' , then it is not quite clear whether an observed difference between i and i' is attributable to a difference in the underlying population or instead to the difference in the smoothing parameters.

The actual choice of the smoothing parameters is often driven by the context. In our application, we pick the location and number of knots to reflect the bid-arrival distribution, which is densest for the last day, and in particular for the last few moments of the auction. The choice of p depends, among other things, on whether higher order derivatives of the curve are also desired. The value of the penalty term λ is chosen by inspecting the resulting functional objects in order to ensure satisfactory results (Ramsay and Silverman, 2002). An alternative approach is to pick λ so as to balance the smoothness and the data fit (Wang et al., 2005). In particular, one can measure the degree of smoothness of the spline via its distance to the smoothest possible fit, a straight line through the data. The data fit, on the other hand, can be measured as the distance between the spline and the actual data points. One then chooses a value of λ that balances the two. We investigate and compare different smoothing parameters for our dataset in what follows.

The process of moving from observed data to functional data is then as follows. For a set of n functional objects, let t_{ij} denote the time of the j th observation ($1 \leq j \leq n_i$) of the i th object ($1 \leq i \leq n$), and let $y_{ij} = y(t_{ij})$ denote the corresponding measurements. Let $f_i(t)$ denote the penalized smoothing spline fitted to y_{i1}, \dots, y_{in_i} . Then, functional data analysis is performed on the continuous curves $f_i(t)$ rather than on the noisy observations y_{i1}, \dots, y_{in_i} . That is, after creating the functional objects $f_i(t)$, the observed data y_{i1}, \dots, y_{in_i} are discarded and subsequent modeling, estimation and inference are based on the functional objects only.

One important implication of this practice is that any error or inaccuracy in the smoothing step will propagate into the inferences and conclusions made based on the functional model. To make matters worse, the observed data are discarded after the functional data are created and are therefore often hard to retrieve, and any violation of the functional model is confounded with the error at the smoothing step. That is, it is hard to know whether a model violation is due to model misspecification or due to anomalies at the smoothing step. For this reason, it is important to carefully monitor the functional object recovery process and to detect inaccuracies early in the process using appropriate tools. Although measures for evaluating the goodness of fit of the functional object to the observed data are available (such as those based on the residual sums of squares, or criteria that include the roughness penalty), it is unwise to rely on these measures alone, and visualization becomes an indispensable tool in the process.

Consider Figs. 5.1–5.3 for illustration. The figures compare the functional objects recovered for three different smoothing scenarios. Specifically, for bidding data from 16 different eBay online auctions, Fig. 5.1 shows the functional objects obtained from penalized smoothing splines using a spline order $p = 2$ and a small smoothing parameter $\lambda = 0.001$. Figure 5.2 on the other hand corresponds to the same spline order ($p = 2$) but a larger smoothing parameter ($\lambda = 1$). In Fig. 5.3 we use a spline order $p = 4$, a smoothing parameter $\lambda = 10$, and a data preprocessing step via interpolation. The exact details of the smoothing are not of interest here and can be found elsewhere Jank and Shmueli (2005). What is of interest here though is the fact that Figs. 5.1–5.3 correspond to three *different* approaches to recovering functional objects from the *same* data. The researcher could have taken either one of these three approaches and

used the resulting functional objects for subsequent analysis. However, as we will explain next, two of the three approaches lead to very unrepresentative functional objects and therefore probably to erroneous conclusions.

Statistical conclusions typically make sense only in the context of their application, and ignorance thereof will lead to wrong conclusions. This is no different for visualizations. As mentioned earlier, Figs. 5.1-5.3 show bidding data from 16 eBay auctions. All auctions are for the same item (a *Palm PDA M515* personal digital assistant), all lasted seven days, and all auctions were collected during the same time period (March to June, 2003) and had a retail value of about \$250 at the time of collection. In that sense, all 16 auctions are comparable. The circles correspond to the observed bids (i.e., the times and sizes of the bids), while the solid lines correspond to the resulting functional objects via penalized smoothing splines. The objective at this stage is to recover, from the observed bidding data, the underlying price curve. The price curve describes the price evolution during an auction, and its derivatives measure the price dynamics. In that sense, the objective is to create a functional object that is representative of the evolution of price between the start and end of the seven-day auction. The process of bidding on eBay follows an ascending format and the price curve should naturally reflect that. This goal is somewhat complicated by

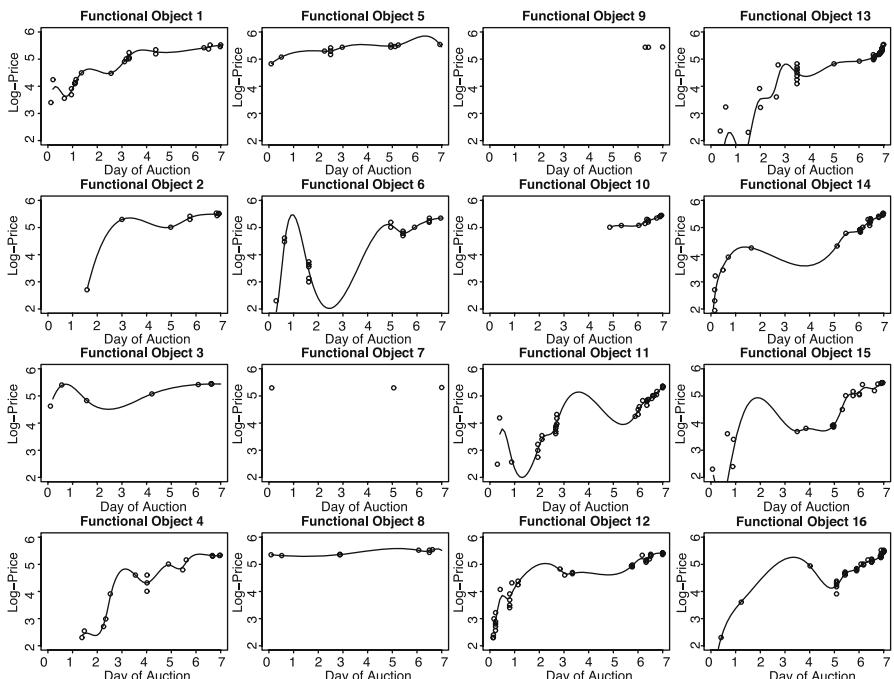


Figure 5.1. Creating functional objects: price curves using penalized smoothing splines with $p = 2$ and $\lambda = 0.001$. Note that the x -axis denotes the day of the auction, which is between 0 and 7, and the y -axis denotes the auction price on a log scale

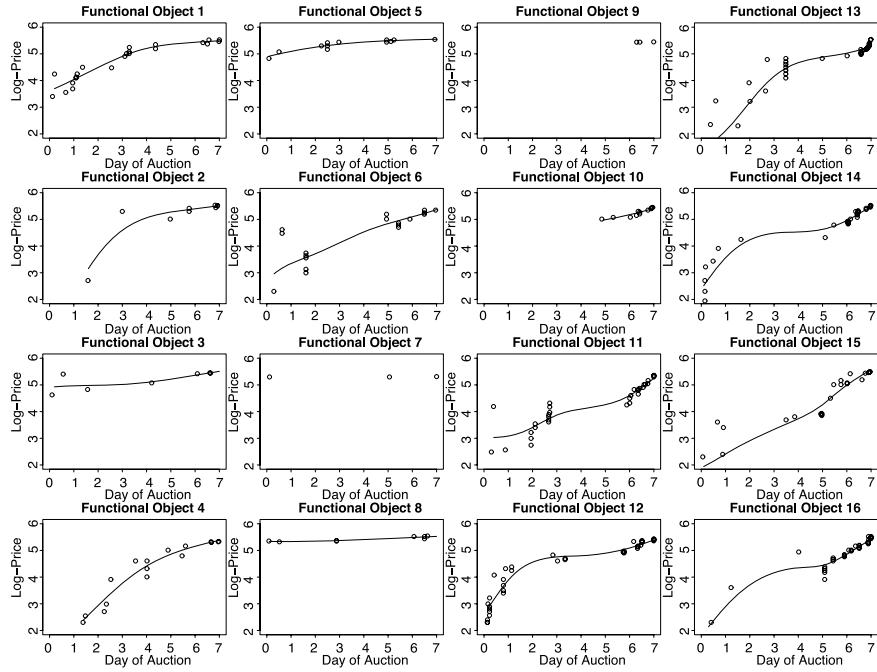


Figure 5.2. Creating functional objects: price curves using penalized smoothing splines with $p = 2$ and $\lambda = 1$

the fact that observed bids do *not* increase monotonically due to eBay's proxy bidding system (Jank and Shmueli, 2005). Thus, creating representative functional objects is not a straightforward task.

Consider Fig. 5.1. We can see that the functional objects are very "wiggly" and certainly do not do a good job of representing the monotonic price increase in the auction. Moreover, we also notice that some of the objects (e.g., #2 and #10) only *partially* cover the seven-day period and thus do not represent the price evolution over the *entire* auction. The reason for this is the software: the penalized spline module *pspline* in R, by default, returns a function that is defined only on the range of the input data. Hence, since the bids for #2 and #10 cover only a small part of the duration of the auction, so does the resulting functional object. Lastly, we notice that there are no functional objects for #7 and #9. The reason for this is that the *pspline* module requires at least $2p + 1$ data points to estimate a smoothing spline of order p . This means that for a second-order smoothing spline we need at least $(2)(2) + 1 = 5$ points. However, both #7 and #9 only have three bids and thus no functional object is created. This loss of information is quite disturbing from a conceptual point of view, since data are in fact available for these two auctions and the missing (functional) data are a consequence of the functional object generation process. In summary, if the researcher were to use the smoothing approach from Fig. 5.1 "blindly" (i.e., without

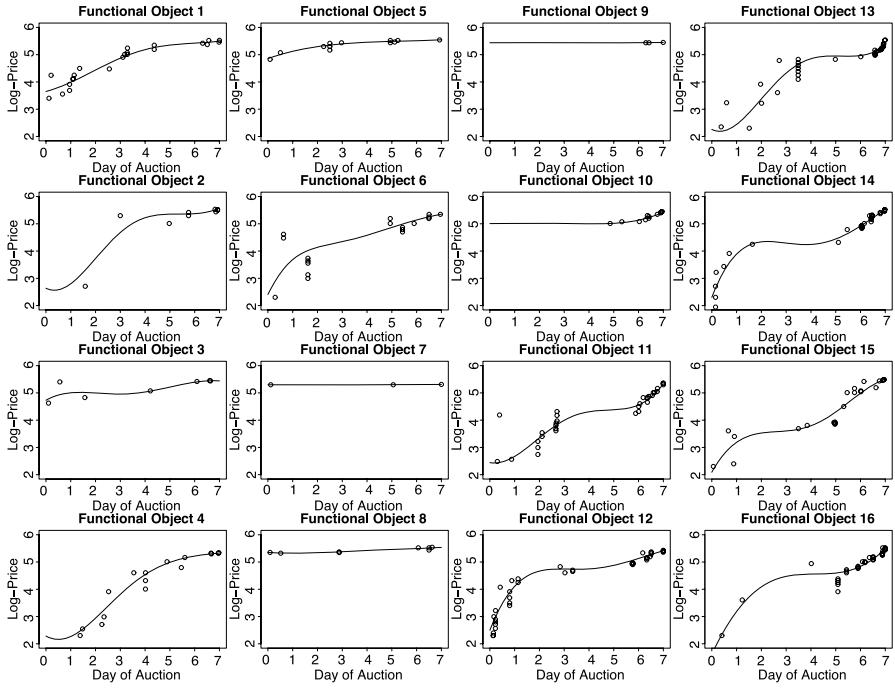


Figure 5.3. Creating functional objects: price curves using data preprocessing via interpolation of the bids and penalized smoothing splines with $p = 4$ and $\lambda = 10$

carefully checking the results), then very unrepresentative functional objects would be obtained and valuable information would be lost.

One reason for the poor representativeness of the objects in Fig. 5.1 is the low value of the smoothing parameter. Increasing λ to 1 (Fig. 5.2) results in much smoother (i.e., less wiggly) price curves. However, there are still some partial functional objects (#2, #10) and missing functional objects (#7, #9). Moreover, while the higher value of λ results in curves that are much less wiggly, some of the functional objects now appear to be too inflexible (e.g., #15 may be too similar to a straight line).

We can achieve a better fit (i.e., one with more flexibility, but only a little extra wigginess) by increasing the order of the spline together with the magnitude of the smoothing parameter. We can also solve the problem of partial and missing functional objects by using a preprocessing step via interpolation. That is, we interpolate the observed bidding data and fit the smoothing spline to a discretized grid of the interpolating function. Specifically, let \tilde{t}_{ij} denote the time that the j th bid is placed in auction i , and let \tilde{y}_{ij} denote the corresponding bid amount. We interpolate the \tilde{y}_{ij} values linearly to obtain the interpolating function $\tilde{y}_i(t)$. Let now $0 \leq t_j \leq 7$ be a common grid of time points. We evaluate all $\tilde{y}_i(t)$ values at the common grid points t_j to obtain $y_{ij} := \tilde{y}_i(t_j)$ and fit the smoothing spline to the y_{ij} values. In this way, we can ensure that we estimate the smoothing spline based on a sufficient number of

points that cover the entire range of the seven-day auction. The results obtained from doing this can be seen in Fig. 5.3. Now the functional objects appear to be very representative of the price evolution, much better than in the previous two approaches. Equally importantly, there are now no missing or partial functional objects. Inference based on the objects in Fig. 5.3 is likely to yield the most reliable insights about the price evolution in online auctions.

The previous examples illustrate the importance of visualization at the object recovery stage. Although the causes of problems at this stage may often be quite trivial (e.g., unfortunate software default settings or poor parameter choices), they are typically hard to diagnose without the use of proper visualizations.

5.4 Visualizing Functional Observations

5.4.1 Visualizing Individual Objects and Their Dynamics

The first step in statistical analysis is typically to scrutinize data summaries and graphs. Data summaries include measures of central tendency, variability, skewness, etc. Traditionally, summary statistics are presented in numerical form. However, in the functional setting, each summary statistic is actually a functional object, such as the *mean function* or the *standard deviation function*. Since there are usually no analytical, closed-form representations of these functions, one resorts to graphical representations of the summary measures. The left panel in Fig. 5.4 shows the (pointwise) mean price curve (solid thick line) together with the 95 % pointwise upper and lower confidence curves (broken thick lines) for the 16 auctions from Sect. 5.3. We compute these pointwise measures in the following way. For an equally spaced grid $t_i \in [0, 7]$, we compute the mean and standard deviation for the 16 auction prices at each grid point t_i . We use these two measures to construct 95 % upper and lower confidence bounds at each grid point. By interpolating the results, we obtain the mean and confidence curves in Fig. 5.4. Notice that since we only consider 16 auctions in this example, one can easily identify the minimum and maximum prices of all curves. In larger data sets, one may also want to add a curve for the (pointwise) minimum and maximum, respectively.

One of the main advantages of functional data analysis is that it allows for an estimation of derivatives. The nonparametric approach to the recovery of the functional object guarantees that local changes in the data are well-reflected, and yet the object's smoothness properties also allow for a reliable estimation of partial derivatives. For instance, setting $m = 4$ in the penalty term in (5.2) guarantees smooth first and second derivatives. Knowledge of the derivatives can result in an important advantage, especially for applications that experience change. Take the online auction setting as an example. While the price curve $f(t)$ describes the exact *position* of the price at any time point t , it does not reveal how fast the price is *moving*. Attributes that we typically associate with a moving object are its *velocity* (or its *speed*) and its *acceleration*. Velocity and acceleration can be computed via the first and second derivatives

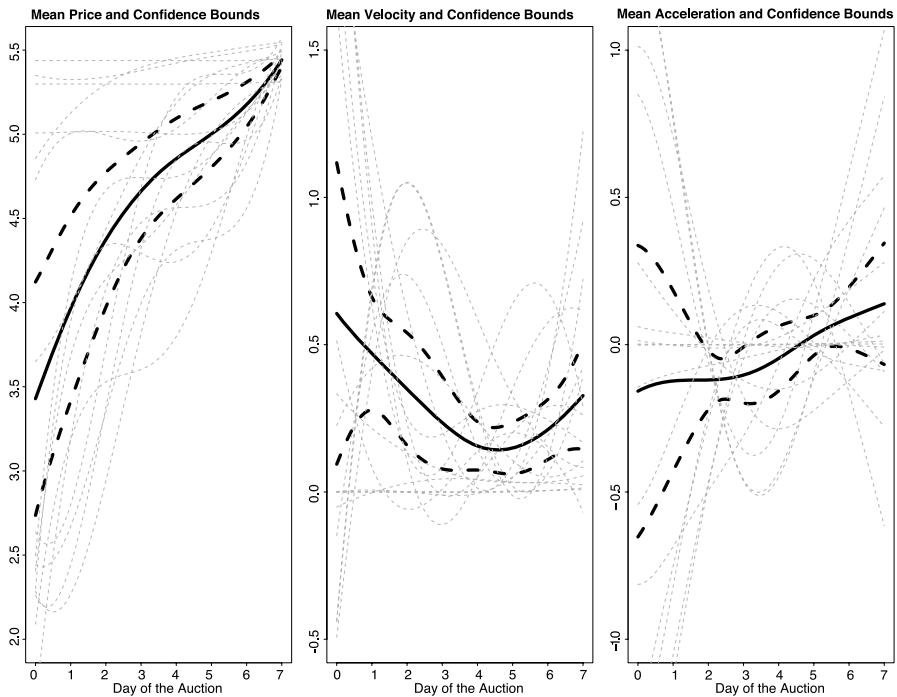


Figure 5.4. Summaries for functional objects: pointwise mean and 95 % pointwise confidence bounds for the price evolution, price velocity, and price acceleration of the 16 eBay online auctions

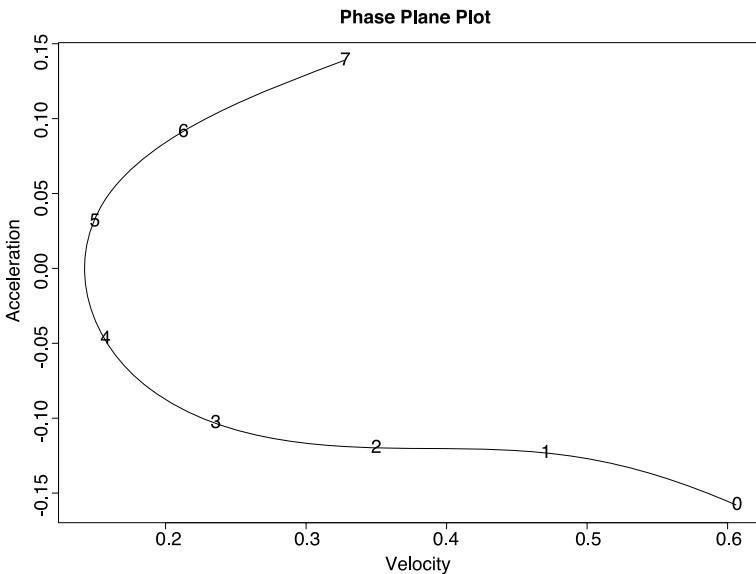


Figure 5.5. Phase-plane plot of the mean velocity vs. the mean acceleration. Each number on the curve indicates a particular day of the auction

of $f(t)$, respectively. Knowledge of the dynamics can be important for pinpointing the periods during which the auction price experiences only minor change, which in turn is important for forecasting the final price (Wang et al., 2005). The middle and right panels of Fig. 5.4 show the velocity and acceleration for the 16 eBay auctions together with the pointwise mean and confidence bounds.

Another way of investigating the interplay of dynamics is with the aid of so-called *phase-plane plots*. Phase-plane plots graph dynamics against one another. For instance, Fig. 5.5 shows a graph of mean velocity versus mean acceleration. The numbers on the curve indicate the day of the auction. We can see that at the start (day 0), high velocity is accompanied by low, negative acceleration (=deceleration). Acceleration precedes velocity, so deceleration *now* results in a lower velocity *tomorrow*, and consequently the velocity decreases to below 0.5 on day 1. This trend continues until the acceleration turns positive (between days 4 and 5), causing the velocity to pick up towards the end of the auction end. Phase-plane plots are useful for diagnosing whether the interplay of dynamics suggests a system that could be modeled by a suitable differential equation.

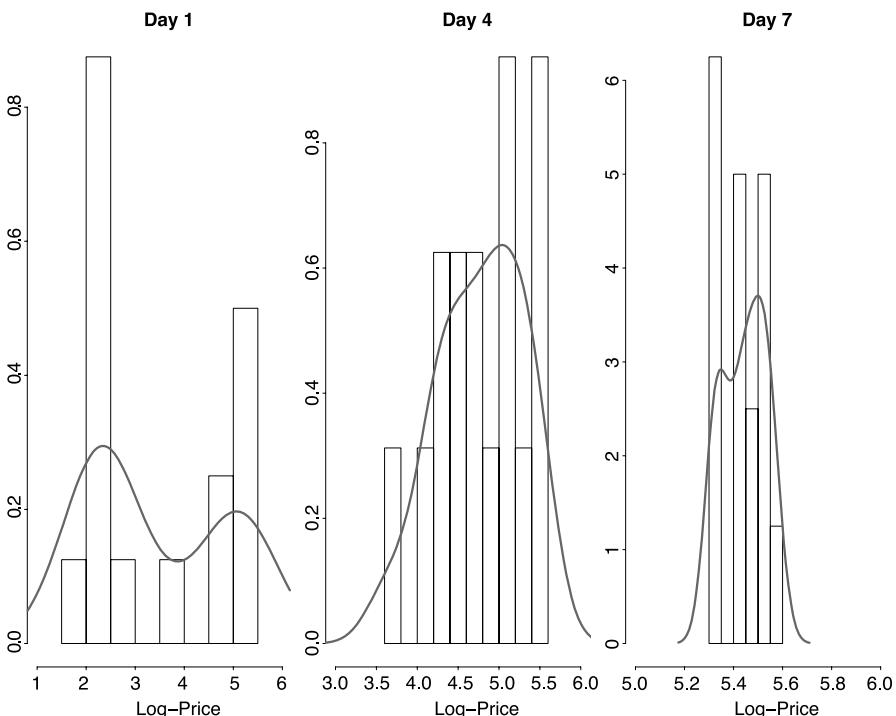


Figure 5.6. Distribution of functional objects: histograms of price (plotted on a log scale) at days 1, 4 and 7 of the 16 eBay online auctions. The gray line corresponds to a kernel density estimate with a Gaussian kernel

Another reason to explore the data is to investigate the distributions of individual variables. Since most parametric models require the response to follow a certain distribution (typically the normal distribution), this step is important for selecting the right model and for ensuring the appropriateness of the selected model. One standard tool for investigating the distribution of a numerical variable is the histogram. However, generalizing the idea of a histogram to the functional context is a challenging task, since the input variable is a continuous function. One solution is to graph the distribution of the functional object at only a few select snapshots in time. This can be done by discretizing the object and graphing pointwise histograms (or similar plots such as probability plots) at each time point. Figure 5.6 shows snapshots of the distributions of the 16 eBay price curves at days 1, 4 and 7. These snapshots allow conclusions to be drawn about the distribution of the entire functional object. Notice that Fig. 5.6 also shows kernel density estimates of the distributions and thus allows conclusions to be drawn about the evolution of the functional density over days 1–7. One can generalize this idea to obtain the density continuously over the entire functional object (rather than only at discrete time points). Specifically, Fig. 5.7 shows the density estimates evaluated over a fine grid and subsequently interpolated. We can see that the distribution is very flat at the beginning of the auction and it starts to peak towards the end.

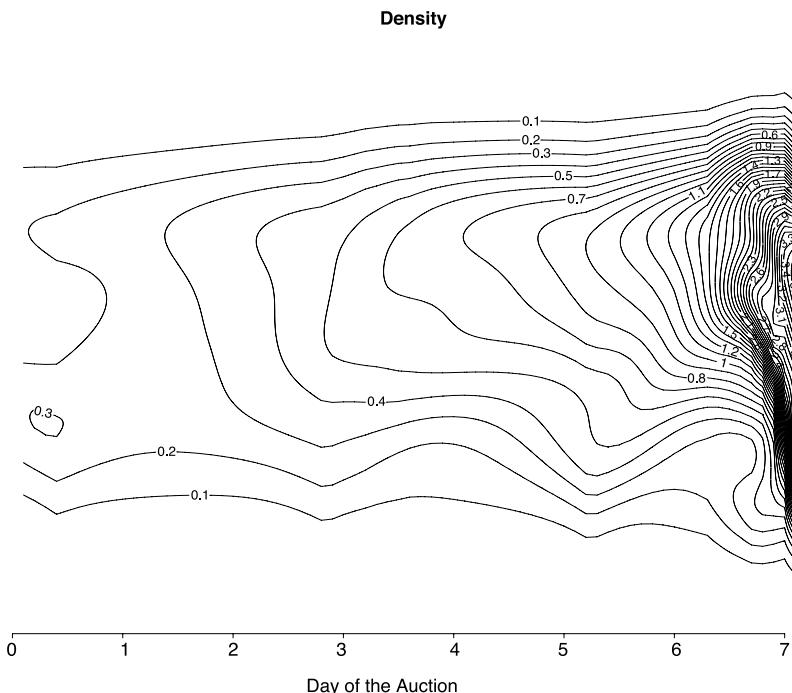


Figure 5.7. Contour plot of the density of the functional objects over the seven-day auction. The contour plot is obtained by calculating kernel density estimates (as done in Fig. 5.6) over a fine grid and subsequent interpolation over the seven-day period

Visualizing Relationships Among Functional Data

After examining each variable individually, the next step in exploratory data analysis is typically to investigate relationships across several variables. For two numerical variables, this is often accomplished with the help of scatterplots. One way of generalizing the traditional scatterplot to the functional setting is, again, to draw a sequence of *pointwise* scatterplots. Figure 5.8 shows scatterplots at days 1, 4 and 7 for the auction price versus the opening bid (on a log scale). We can see that the relationship between the two variables changes over the course of time. While there is a strong positive effect at the beginning of the auction (left panel), the magnitude of the effect decreases at day 4 (middle panel), and there is barely any effect at all (possibly even a slightly negative effect) at the end of the auction (right panel). This suggests that the relationship between the opening bid and the auction price can be modeled well using a time-varying coefficient model. Of course, one aspect that remains unexplored in this pointwise approach is a possible three-way interaction between the opening bid, the price and the time. Such an interaction could be detected using a three-dimensional scatterplot. However, as the left panel in Fig. 5.9 illustrates, three-dimensional graphs have the disadvantage that they are often cluttered and difficult to read. We can improve the interpretability by using smoothing. The right panel in

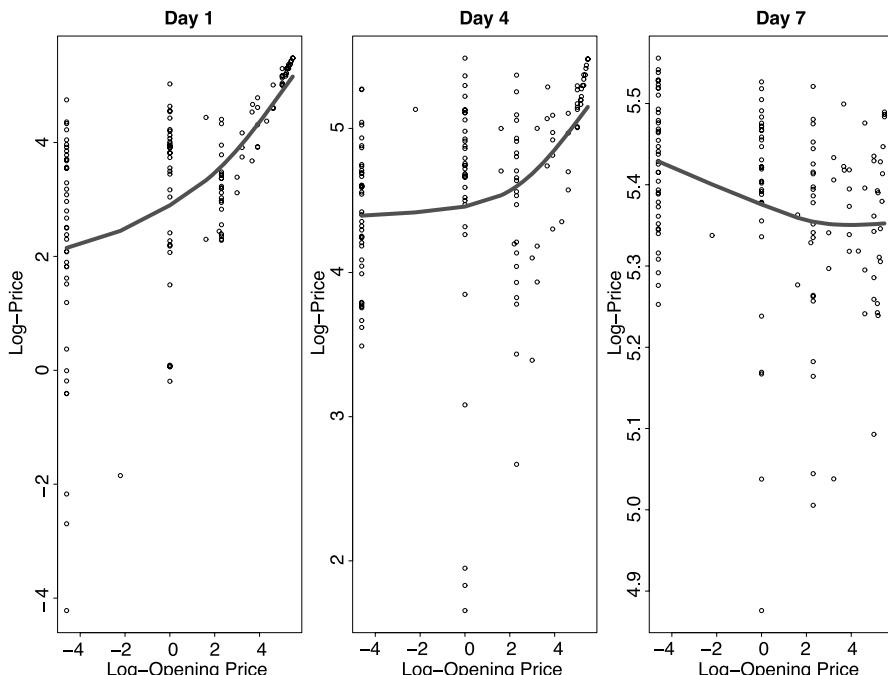


Figure 5.8. Relationships among functional objects: scatterplots of (log) price vs. (log) opening bid at days 1, 4 and 7 for a sample of eBay online auctions. The *solid gray line* corresponds to a cubic smoothing spline with three degrees of freedom

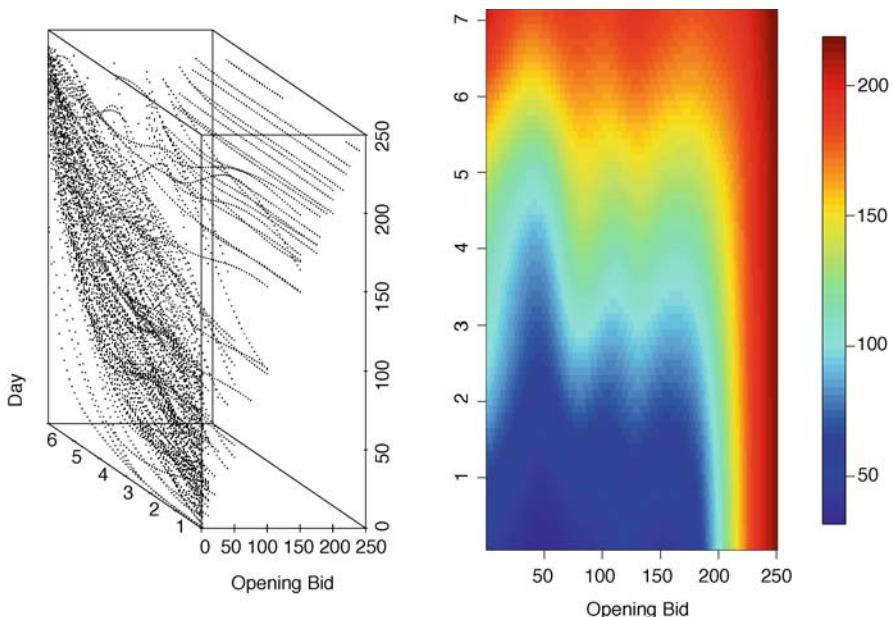


Figure 5.9. [This figure also appears in the color insert.] Relationships among functional objects: the left panel shows a 3-D scatterplot of opening bid (x), day of the auction (y) and price (z). The right panel shows a smoother version of the price surface, obtained using a Nadaraya–Watson smoother. In that plot, the x -axis is the opening bid and the y -axis is the day of the auction

Fig. 5.9 shows a smoother image of the price surface obtained using a Nadaraya–Watson smoother. The three-way relationship between opening bid, price and time is now much easier to see.

Visualizing Functional and Cross-sectional Information

5.4.3

As illustrated above, it is more challenging to visualize functional data than classical data. The visualization process is often further complicated by the coupling of functional observations with cross-sectional attribute data. For example, online auction data include not only the bid history (i.e., the times and sizes of bids), but also auction-specific attributes corresponding to auction design (e.g., length of the auction, magnitude of the opening bid, use of a secret reserve price, use of the “Buy It Now” option, etc.), bidder characteristics (e.g., bidder IDs and ratings), seller characteristics (e.g., seller ID and rating, seller location, whether or not a seller is a “Powerseller,” etc.), and product characteristics (e.g., product category, product quality and quantity, product description, etc.). All of these characteristics correspond to cross-sectional information in that they do not change during the auction. The coupling of time series with cross-sectional information is important because the relationship between the two could be the main aim or at least one of the aims of the analysis.

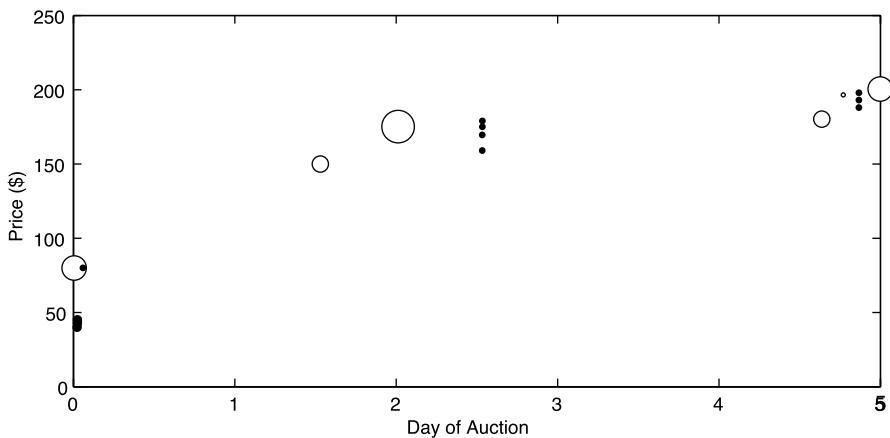


Figure 5.10. Profile plot of a single five-day auction. The *circles* represent bids, with circle size proportional to the bidder's eBay rating

Standard visualization tools are geared towards the display of either time-series data alone or cross-sectional data alone – almost never both.

The combination of time-series and cross-sectional data into one display is rare and requires careful, application-specific modifications of standard methods. Shmueli & Jank (2005) propose the use of *profile plots* for displaying the temporal sequence of bids together with additional auction attributes (such as a bidder's rating) in the same graph. This is illustrated in Fig. 5.10, which describes the sequence of bids in a five-day eBay auction. The circle size is proportional to the bidder's eBay rating. However, profile plots are more suitable for visualizing single auctions, and do not scale well.

Another type of plot that is suitable for visualizing functional data is the *rug plot* (Hyde et al., 2005). A rug plot displays curves (i.e., functional objects) over calendar time in order to explore the effects of event concurrency. Figure 5.11 shows a rug plot displaying the price curves of 217 eBay auctions of a *Palm M515 PDA* that took place over a three-month period. Each colored line represents an individual auction, and it is estimated via monotone smoothing (Ramsay and Silverman, 2005). Monotone smoothing is computationally more expensive than penalized smoothing, but it ensures that the resulting line increases monotonically. The black line represents the average daily closing price. We can see that daily prices vary quite significantly, and so does the daily variation in price (gray bands). Furthermore, we can see that there are time periods with many similar, almost parallel price curves for the same auction durations (e.g., seven-day auctions – green curves – at around 4/3 and also around 4/23). Moreover, the closing prices after 4/3 appear to be relatively low, and so does the associated price variability. Most auctions closing at that time are seven-day auctions with similar shapes. It would be interesting to see if one could establish a more formal relationship between similar price patterns (i.e., parallel price curves) and their effect on the price and its uncertainty.

The rug plot in this example combines functional data with attribute data via the time axis (calendar time on the *x*-axis takes into account the start and the end of

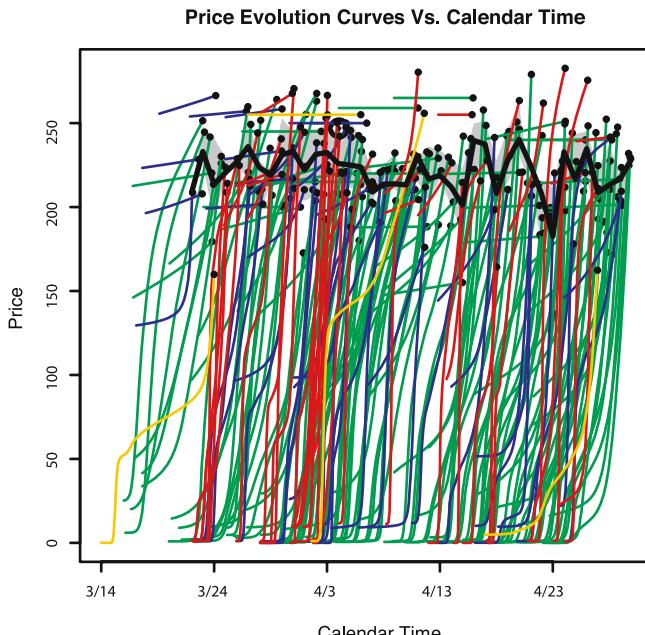


Figure 5.11. [This figure also appears in the color insert.] Rug plot displaying the price evolution (*y*-axis) of 217 online auctions over calendar time (*x*-axis) during a three-month period. The colored lines show the price path of each auction, with color indicating auction length (yellow, three days; blue, five days; green, seven days; red, ten days). The *dot* at the end of each line indicates the final price of the auction. The *black line* represents the average of the daily closing price, and the gray band is the interquartile range

the curve) and via color (different colors for different auction durations). Notice that the plot scales well for a large number of auctions, but it is limited in the number of attributes that can be coupled within the visualization.

Finally, trellis displays (Cleveland et al., 1996) are another method that supports the visualization of relationships between functional data and an attribute of interest. This is achieved by displaying a series of panels where the functional objects are displayed at different levels (or categories) of the attribute of interest (see for instance Shmueli and Jank, 2005). In general, while static graphs can capture some of the relationships between time series and cross-sectional information, they become less and less insightful as the dimensionality and complexity of the data increase. One of the reasons for this is that they have to accomplish meaningful visualizations at several data levels: relationships within cross-sectional data (e.g., find relationships between the opening bid and a seller's rating) and within time-series data (e.g., find an association between the bid magnitudes, which is a sequence over time, and the number of bids, which is yet another sequence over time). To complicate matters, these graphs also need to portray relationships across the different data types; for example, between the opening bid and the bid magnitudes. In short, the graphs have

to be very flexible to accommodate all of these different criteria. Ideally, one would want to literally “dive” into the data and explore it interactively.

Information visualization tools apply several common strategies that enable user control over data displays (see Shneiderman and Plaisant (2004), Card et al. (1999) or Plaisant (2005)). A primary strategy is to allow the user to manipulate a set of widgets, such as dynamic query sliders that allow the user to select the ranges of the desired variables, in a process often called *conditioning*. The power of interaction is that users can rapidly (100 ms) and incrementally change the ranges and explore the effect of these changes on the display. For example, users can move a slider to gradually eliminate auctions with low starting prices and see if that removes time series plots that end with low, middle, or high closing prices. A second strategy is to have multiple views of the data, such as scattergram, histogram, tabular, or parallel coordinate views. The users can then select a single or multiple items in one view and see the results in another view (“brushing”). For example, users can select the time series with sharp increases near the finish in order to see if these had relatively few previous bids.

Selectivity and user control are essential, as they support exploration (to confirm hypotheses) and discovery (to generate new hypotheses) (Chen, 2004). The large number of possibly interesting features in high-dimensional data means that static displays and a fixed set of data-mining algorithms may not be enough. Users can quickly spot unusual outliers, bimodal distributions, spikes, long or short tails on one side of a distribution, and surprising clusters or gaps. Users may also detect strong or weak relationships, which can be positive or negative, linear, quadratic, sinusoidal, exponential, etc.

The strongest tools are likely to be those that combine data-mining algorithms with potent user interfaces (Shneiderman, 2002). These have the potential to provide thorough coverage through a systematic process of exploration in which users can decompose a complex problem domain into a series of simpler explorations with ranking criteria, and they guide user attention to potentially interesting features and relationships (Seo and Shneiderman, 2005).

5.5 Interactive Information Visualization of Functional and Cross-sectional Information via TimeSearcher

TimeSearcher is a time series visualization tool developed at the Human–Computer Interaction Laboratory (HCIL) of the University of Maryland. *TimeSearcher* enables users to see an overview of long time series ($> 10\,000$ points), to view multivariate time series, to select data with rectangular time boxes, and to search for a selected pattern. Its main strength comes from its interactivity, which allows users to explore time series data in an active way. Unlike static graphs, an interactive approach can be more powerful and can lead to a better understanding of the data.

TimeSearcher can be used to visualize functional data if a discretized version of the curves is used as input. The level of discretization is chosen by the user, and is generally selected such that the interpolated points result in apparently continuous curves. In a collaborative project, the authors (two statisticians and two computer scientists from HCIL) further developed the tool to accommodate a particular type of functional data, namely price curves from online auctions. As described in Sect. 5.2, auction data include bid histories, which we convert to smooth curves, and additional attributes. To illustrate the enhanced features of *TimeSearcher* that support functional data exploration, we use a dataset of 34 magazine auctions on eBay that took place during the fall of 2004. The data include the bid histories (converted to curves) and the attributes for each auction.

The first step involves aligning the auctions of different durations that took place at different times. We chose to align the time scale so that in *TimeSearcher* the x -axis shows the proportion of the full duration of the auction. We then added the auction duration and the additional lost temporal information (day and time at which the auction commenced and finished) to the list of attributes.

Capabilities of TimeSearcher

5.5.1

TimeSearcher was extended for the analysis of online auction data to include attribute data browsing with tabular views and filtering by attribute values and ranges (e.g., start date or seller), which were both tightly coupled to the time series visualization. The application is available for download from <http://www.cs.umd.edu/hcil/timesearcher>. Figure 5.12 shows the main screen of the visualization tool with a dataset of 34 eBay auctions for magazines. The time series are displayed in the left panel, with three series (i.e., three variables) for each auction: "Price" (top), "Velocity" (middle), and "Acceleration" (bottom), which correspond to the price curves and their first and second derivatives, as explained in the previous section. At the bottom of the screen, an overview of the entire time period covered by the auctions is provided to allow users to specify time periods of interest to be displayed in more detail in the left panel. On the right, the attribute panel shows a table of auction attributes. Each row corresponds to an auction, and each column to an attribute, starting with the auction ID number. In this dataset there are 21 attributes, and scrolling provides access to attributes that do not fit into the available space. Users can choose how much screen space is allocated for the different panels by dragging the separators between the panels, enlarging some panels and reducing others. All three panels are tightly coupled so that an interaction in one of the panels is immediately reflected in the other panels. Attributes are matched with time series using the auction ID number as a link.

The interactive visualization operations can be divided into time series operations (functional data) and attribute operations. We describe these next.

Functional Object Operations

TimeSearcher treats each time series, represented by a curve, as a single observation, and allows operations on the complete curve or on subsets of it. The following operations can be applied to the functional data (curves).

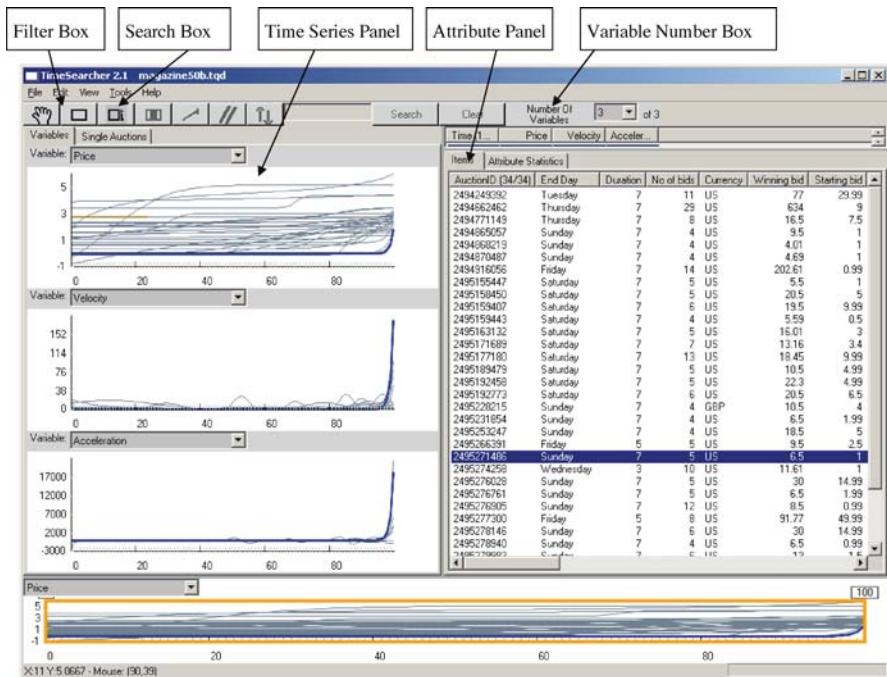


Figure 5.12. The main screen of *TimeSearcher*, showing price curves and dynamics curves (left) coupled with attribute data (right) for 34 online auctions

Curve selection: A particular curve (or a set of curves) is selected by mouse-clicking on any point in that curve. The selected curve is then highlighted in blue (see Fig. 5.12). Hovering over a curve will highlight it in orange, thereby simplifying the task of mouse coordination.

Zooming: The overview panel at the bottom of the screen displays the time series for one of the variables and allows users to specify the part of the time series they want to zoom in on. The orange field of the view box determines the time range that is displayed on the upper left panels. Any one of the panels can be used for the display in the orange field. To zoom, users drag the sides of the box. By zooming in, the user can focus on a specific period in the data and see more details. In many cases zooming also results in better separation between the curves, enabling easier selection and deselection of lines. The box can also be dragged right and left to pan the display and show a different time period. Regardless of the range of the detail view, the overview always displays the entire time series and provides context for the detail view.

Focusing on a variable: To focus on a certain variable (price, velocity, or acceleration curves), users can choose to view only the panel on the left, which provides a larger view of those curves. This results in clearer separation between curves, which can be especially useful when there are many auctions. Users can specify the number of variables to be shown (here one, two or three) and select which

of the variables should be displayed. This allows extra flexibility in terms of the choice of derivatives to display.

Filtering curves: Users can filter the curves to see only auctions of interest by using filter widgets called TimeBoxes. One can click on the TimeBox icon of the toolbar and draw a box on the time series panel of interest. Every curve that passes through the box (between the bottom and top edges of the box for the duration that the box occupies) is kept, while all of the other curves are filtered by graying them out. The corresponding auctions are also removed from the attribute panel on the right. Figure 5.13 shows a typical filter TimeBox used to view only the auctions that end with high price velocity. In the attributes panel, users can see that all of these auctions finished around the weekend. They can apply multiple TimeBoxes to the same or separate variables in order to form conjunctive queries (i.e., a logical AND combination of individual TimeBox queries). For example, users can search for auctions that end with low prices and high velocities.

Searching for patterns in curves: When comparing price curves and (especially) price dynamics, one useful tool is the pattern search. This is achieved by drawing a SearchBox on a selected curve for a certain time duration. The pattern is the part of the series that the SearchBox covers horizontally, and this SearchBox is searched simultaneously for all other curves at any time point in the auction. There is a tolerance handle on the right of the SearchBox that allows the similarity to be specified. For example, users can search for auctions that have price curves with steep escalations at any time during the auction. TimeBoxes and SearchBoxes can be combined into multistep interactive searches.

Functional summaries: One can obtain numerical summaries for a set of functional objects using the *riverplot* in Fig. 5.14. The riverplot is a continuous form of the boxplot and displays the (pointwise) median together with the 25 % and 75 % confidence bounds. The riverplot allows for a condensed display of the average behavior of all curves together with the uncertainty around this average.

Attribute Operations

Manipulating the attribute data and observing the coupled functional data are useful ways of learning about relationships within the data across the different data types. The following operations support such explorations (in addition to more standard explorations of attribute data alone).

Sorting auctions: Users can sort the auctions by any attribute by clicking on the attribute name in the first row. A click sorts in ascending order, while the next click sorts in descending order. Sorting can be performed on numerical as well as text attributes. The sorting also recognizes day-of-the-week and time formats. The sorting is useful for learning about the ranges of the values for the different attributes, the existence of outliers, the absence of certain values, and possible errors and duplications in the data. Furthermore, sorting can allow users to visually spot patterns of "similar" auctions, by making auctions with similar attribute values appear consecutively in the auction list. Users can sort according to more

than one column. In addition, the order of the attribute columns can be changed by clicking and dragging the attribute names to the right or left.

Highlighting groups of auctions: After the attribute/s of interest have been sorted, groups of auctions can be selected and their corresponding time series in the left panels are highlighted. For example, if the attributes table is sorted by the end day of the auction, it is easy to select all auctions that ended on a weekday from the table, and see the corresponding time series highlighted, which reveals that they are the auctions that tend to end with the highest prices (Fig. 5.12).

Summary statistics: The summary statistics tab shows the mean, standard deviation, minimum, max, median, and the quartiles for each attribute for the selected auctions. This is updated interactively when the auctions are filtered with TimeBoxes, or when users select a subset of auctions manually. For example, while the median seller rating of all auctions is 615, when users apply a TimeBox to select the auctions that started with a low price, the median seller rating jumps to 1487. Moving the TimeBox to select auctions that started with a high price results in a median seller rating of 243, which may imply that starting the auction with a low starting price is a strategy that tends to be employed by experienced sellers.

The array of interactive operations described above support data exploration, and Shmueli et al. (2006) describe how these operations can be used for the purpose of decision-making, through a semi-structured exploration. Exploration can be guided by a set of hypotheses, and the results can then help the user to find support for and direct the user towards suitable formal statistical models. In particular, they show how insights gained from the visual exploration can improve seller, bidder, auction house, and other vendors' understanding of the market, thereby assisting their decision-making processes.

5.5.2 Forecasting with TimeSearcher

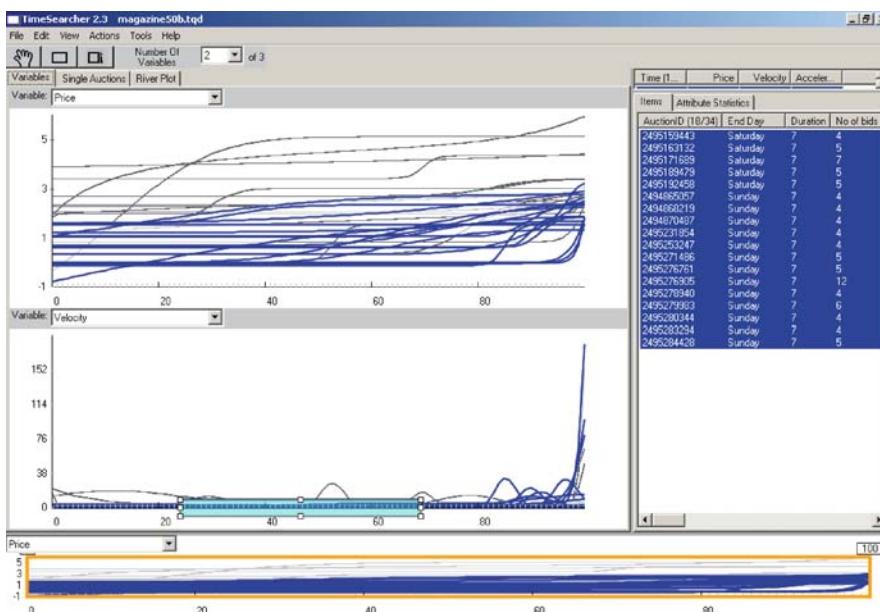
Functional object value forecasting is an area of functional data analysis that has not received much attention so far. It involves forecasting the value of a curve at a particular time t (either a particular curve in the data or the average curve), based on information contained in the functional data and the attribute data. We propose the following general forecasting procedure:

Select similar items: For a partial curve (e.g., an ongoing auction that has not closed), we select the subset of curves that are closest to the curve of interest in the sense of similar attributes and curve evolutions and dynamics. For the attribute criterion, this can be achieved either by sorting by attributes and selecting items with similar values for the relevant attributes (e.g., auctions of the same duration and with the same opening price), or directly by using a filtering facility that allows the user to specify limits on the values of each of the attributes of interest (this facility is currently not available in the public version of *TimeSearcher*). When curve-matching, TimeBoxes can be used to find curves that have similar structures during the time periods of interest (e.g., auctions with

high price velocities on day 1 and high prices on day 3). We are currently working on developing a facility for "curve-matching" that is more automated. For instance, consider the case of forecasting the closing price of a seven-day auction that is scheduled to close on a Sunday, with an opening price of \$0.99, and that has displayed very low dynamics so far. Let us assume that we observe this auction until day 6 (85 % of the auction duration). Figure 5.13 illustrates a selection of auctions that all have similar attributes to the above auction (all are seven days long, have an opening price of less than \$5, and close on a weekend), and also have similar curve structures during the first six days of the auction (low velocities, as shown by the filtering box placed on the velocity curves).

similar set: Make a forecast based on the We then use the selected "similar" set of curves to make a prediction for time t by examining their riverplots. The median at time t is then the forecast of interest, and the quartiles at that time can serve as a confidence interval. Although this is a very crude method, it is similar in concept to collaborative filtering. The key is to have a large enough dataset, so that the "similar" subset is large enough. To continue our illustration, Fig. 5.14 shows the riverplot of the subset of "similar" auctions. The forecasted closing price is then the median of the closing prices of the subset of auctions, and we can learn about the variability in these values from the percentile curves on the riverplot.

The forecasting module is still under development, with the goal being a more automated process. However, the underlying concept is that interactive visualization can support more advanced operations (including forecasting) than static visualization.



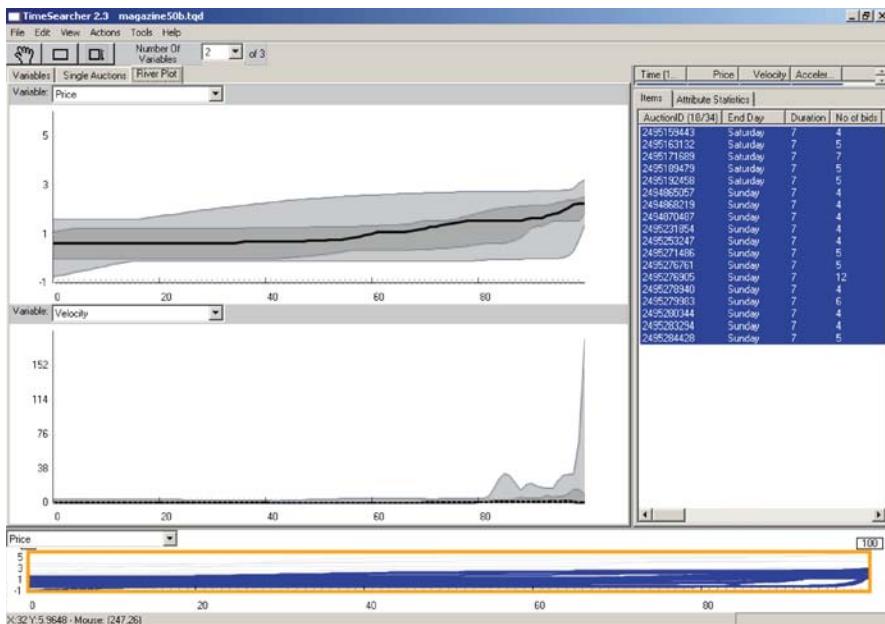


Figure 5.14. Riverplot of the subset of “similar” auctions. The *thick black line* is the pointwise median used for forecasting. The *dark gray bands* around the median show the 25 and 75 % percentile range and the *light gray bands* show the envelope for all similar auctions. This can be seen as a continuous form of box plot

Further Challenges and Future Directions

5.6

Functional data analysis is an area of statistical research that is receiving a growing amount of interest. To date, most of this interest has centered around developing new functional models and techniques for estimating them, while little effort has been expended on exploratory techniques, especially visualization. Classical statistics has become very popular due to both the availability of a wide array of models and the ability to check the appropriateness of these models. The results obtained by applying a particular model will only be wholeheartedly supported if the model is shown to be appropriate. However, this requires that the data can be compared to the model. In this sense, the widespread acceptance and usage of functional models is only going to happen when a range of visualization tools that perform similar tasks to their counterparts in classical statistics are made available.

In this paper, we have outlined a variety of functional visualizations that are available. However, significant challenges remain. These challenges range from concurrency of functional objects, to high dimensionality, to complex functional relationships.

Concurrency of Functional Events

5.6.1

The standard assumption in functional data analysis is independence of the functional observations in the data set. This assumption may not, however, always be plausible. For instance, if the functional object represents the evolution of the price in an online auction, then it is quite possible that the price in *one* auction is affected by the price of an object in *another* auction. That is, if the price in one auction jumps to an unexpectedly high level, then this may cause some bidders to leave that auction and move on to another auction of a similar item. This results in a dependence in price between the two auctions. Or more generally, the result is a dependence between the two functional objects. Capturing this kind of dependence in a mathematical model is not a straightforward task. For a start, how can we unveil such a concurrency in graphical fashion? One promising attempt in this direction is the work of Hyde et al. (2005), which suggests that *rug plots* can be used for the functional objects and their derivatives.

Dimensionality of Functional Data

5.6.2

Another challenge when visualizing functional data is the dimensionality of the data. As pointed out earlier, it is not uncommon for functional data to have three, four or even more dimensions. Most standard visualization techniques work well for two dimensions at most, which is the number of dimensions of the paper that we write on and the computer screen that we look at. Moving beyond two dimensions is a challenge in any kind of visualization task, including that of visualizing functional data.

Complex Functional Relationships

5.6.3

In addition to the high dimensionality, functional data is also often characterized by complex functional relationships. Take for instance the movement of a object through time and space. This movement may be well characterized by a three- or four-dimensional *differential equation* (Ramsay and Silverman, 2002). However, how should we visualize a differential equation? One way is to use phase-plane plots like that in Fig. 5.5. Other approaches have been proposed in Schwalbe (1996).

References

- Aris, A., Shneiderman, B., Plaisant, C., Shmueli, G. and Jank, W. (2005). Representing unevenly-spaced time series data for visualization and interactive exploration. In: *International Conference on Human Computer Interaction (INTERACT 2005)*, 12–16 Sept 2005, Rome, Italy.
- Card, S., Mackinlay, J. and Shneiderman, B. (1999). *Readings in Information Visualization: Using Vision to Think*. Morgan Kaufmann, San Francisco, CA.
- Chen, C. (2004). *Information Visualization: Beyond the Horizon*. Springer, Berlin.
- Cleveland, W.S., Shyu, M. and Becker, R. (1996). The visual design and control of trellis display. *Journal of Computational and Graphical Statistics*, 5:123–155.

- Hyde, V., Jank, W. and Shmueli, G. (2005). Investigating concurrency in online auctions through visualization. *The American Statistician*, 34(3):241–250.
- Jank, W. and Shmueli, G. (2005). *Profiling price dynamics in online auctions using curve clustering*. Technical report, Smith School of Business, University of Maryland, College Park, MD.
- Mills, K., Norminton, T. and Mills, S. (2005). Visualization of network scanning (poster presentation). In: *National Defense and Homeland Security Kickoff Workshop of the Statistical and Applied Mathematical Sciences Institute (SAMSI)*, 11–15 Sept 2005, Research Triangle Park, NC.
- Plaisant, C. (2005). Information Visualization and the Challenge of Universal Access. In: *Exploring Geovisualization*. Elsevier, Oxford.
- Ramsay, J.O. and Silverman, B.W. (2005). *Functional data analysis*, 2nd edn. Springer, New York.
- Ramsay, J.O. and Silverman, B.W. (2002). *Applied functional data analysis: methods and case studies*. Springer, New York.
- Ruppert, D., Wand, M.P. and Carroll, R.J. (2003). *Semiparametric Regression*. Cambridge University Press, Cambridge.
- Schwalbe, D. (1996). *VisualDSolve: Visualizing Differential Equations with Mathematica*. TELOS/Springer, Berlin.
- Seo, J. and Shneiderman, B. (2005). A rank-by-feature framework for interactive exploration of multidimensional data. *Information Visualization*, 4:99–113.
- Shmueli, G. and Jank, W. (2005). Visualizing online auctions. *Journal of Computational and Graphical Statistics*, 14(2):299–319.
- Shmueli, G., Jank, W., Aris, A., Plaisant, C. and Shneiderman, B. (2006). Exploring auction databases through interactive visualization. *Decision Support Systems*, 42(3):1521–1538.
- Shneiderman, B. (2002). Inventing discovery tools: Combining information visualization with data mining. *Information Visualization*, 1:5–12.
- Shneiderman, B. and Plaisant, C. (2004). *Designing the User Interface: Strategies for Effective Human-Computer Interaction*, 4th edn. Addison-Wesley, Reading, MA.
- van Wijk, J.J. and van Selow, E. (1999). Cluster and calendar-based visualization of time series data. In Wills, G. and Keim, D. (eds), *Proceedings 1999 IEEE Symposium on Information Visualization (InfoVis'99)*. IEEE, Los Alamitos, CA, pp. 4–9.
- Wang, S., Jank, W. and Shmueli, G. (2007). Explaining and Forecasting Online Auction Prices and their Dynamics using Functional Data Analysis. Forthcoming at the *Journal of Business and Economic Statistics*.

Visualization Tools for Insurance Risk Processes

Krzysztof Burnecki, Rafał Weron

6.1	<i>Introduction</i>	900
6.2	<i>Software</i>	902
6.3	<i>Fitting Loss and Waiting Time Distributions</i>	902
	Mean Excess Function	902
	Limited Expected Value Function.....	907
	Probability Plot	908
6.4	<i>Risk Process and its Visualization</i>	912
	Ruin Probability Plots	912
	Density Evolution	915
	Quantile Lines	916
	Probability Gates	919

Introduction

This chapter concerns risk processes, which may be the most suitable for computer visualization of all insurance objects. At the same time, risk processes are basic instruments for any non-life actuary – they are needed to calculate the amount of loss that an insurance company may incur. They also appear naturally in rating-triggered step-up bonds, where the interest rate is bound to random changes in company ratings, and catastrophe bonds, where the size of the coupon payment depends on the severity of catastrophic events.

A typical model of insurance risk, the so-called collective risk model, has two main components: one characterizing the frequency (or incidence) of events and another describing the severity (or size or amount) of the gain or loss resulting from the occurrence of an event (Klugman et al., 1998; Panjer and Willmot, 1992; Teugels and Sundt, 2004). Incidence and severity are both generally assumed to be stochastic and independent of each other. Together they form the backbone of a realistic risk process. Consequently, both must be calibrated to the available historical data. All three visualization techniques discussed in Sect. 6.3:

- mean excess function
- limited expected value function
- probability plot

are relatively simple, but at the same time they provide valuable assistance during the estimation process.

Once the stochastic models governing the incidence and severity of claims have been identified, they can be combined into the so-called aggregate claim process,

$$S_t = \sum_{k=1}^{N_t} X_k , \quad (6.1)$$

where the claim severities are described by the random sequence $\{X_k\}$ with a finite mean, and the number of claims in the interval $(0, t]$ is modeled by a counting process N_t , often called the claim arrival process.

The risk process $\{R_t\}_{t \geq 0}$ describing the capital of an insurance company is then defined as

$$R_t = u + c(t) - S_t , \quad (6.2)$$

where the non-negative constant u stands for the initial capital of the company and $c(t)$ is the premium the company receives for sold insurance policies.

The simplicity of the risk process (6.2) is only illusionary. In most cases no analytical conclusions regarding the time evolution of the process can be drawn. However, it is this evolution that is important to practitioners, who have to calculate functionals of the risk process, like the expected time to ruin and the ruin probability. This calls for efficient numerical simulation schemes (Burnecki et al., 2004) and powerful inference tools. In Sect. 6.4 we will present four such tools:

- ruin probability plot
- density evolution plot
- quantile lines
- probability gates.

All four of these techniques permit an immediate evaluation of model adequacy and the risks faced by the company based on visual inspection of the generated plots. As such they are especially useful for high-level managers who are interested in a general overview of the situation and do not need to study all of the computational details underlying the final results.

To illustrate the usefulness of the presented visualization tools, throughout this chapter we will apply them to two datasets. The first one, studied in detail, is the Property Claim Services (PCS) dataset, which covers losses resulting from catastrophic events in the USA. The data include 1990–1999 market loss amounts in US dollars (USD), adjusted for inflation using the Consumer Price Index. Only natural events that caused damage exceeding five million dollars were taken into account, see Fig. 6.1. The second dataset, used here solely to illustrate the risk process inference tools, concerns major inflation-adjusted Danish fire losses of profit (in Danish Krone, DKK) that occurred between 1980 and 1990 and were recorded by Copenhagen Re.

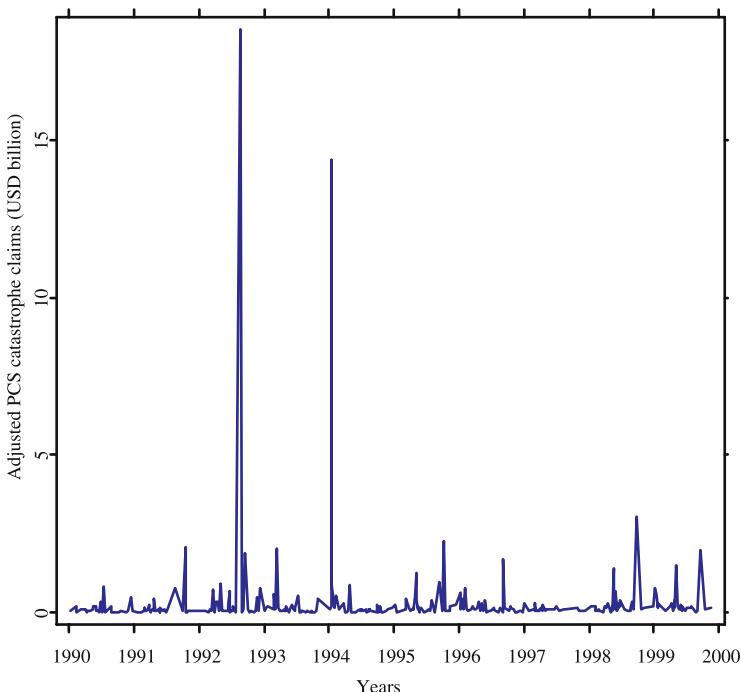


Figure 6.1. Graph of the PCS catastrophe loss data, 1990–1999. The two largest losses in this period were caused by Hurricane Andrew (24 August 1992) and the Northridge Earthquake (17 January 1994). From XploRe

Software

Visualization tools would not be very useful without adequate software. All of methods discussed in this chapter are demonstrated using two software libraries and one standalone application. The *Insurance Library* of Xplore (www.xplore-stat.de) is a collection of quantlets that illustrate various topics related to insurance (Čižek et al., 2005). It is accompanied by online, hyperlinked web tutorials that are freely downloadable from the Web (www.xplore-stat.de/tutorials/_Xpl_Tutorials.html).

The *Ruin Probabilities* toolbox for MATLAB is a set of m-functions with a graphical user interface (GUI) that is primarily created to visualize risk processes and evaluate ruin probabilities (Miśta, 2002). It can be downloaded from www.im.pwr.wroc.pl/~hugo/stronaHSC/Podstrony/Programy.html.

The *SDE-Solver* is a standalone application for the Windows environment. It enables the construction and visualization of solutions of stochastic differential equations (SDEs) with Gaussian, Poisson, and stable random measures (Janicki et al., 2003); for SDE modeling concepts, the reader should consult (Kloeden and Platen, 1995). The graphics make use of quantile lines and density evolution techniques and introduce the interesting concept of interactive probability gates, which give the probability that the simulated process passes through a specified interval at a specified point in time (for details see Sect. 6.4.4). More information about the software can be found at www.math.uni.wroc.pl/~janicki/solver.html.

6.3 Fitting Loss and Waiting Time Distributions

The derivation of loss and waiting times (interarrival) distributions from insurance data is not an easy task. There are three basic approaches: empirical, analytical, and moment-based. The analytical approach is probably the one most often used in practice and certainly the one most frequently adopted in the actuarial literature. It reduces to finding an analytical expression that fits the observed data well and is easy to handle (Daykin et al., 1994).

Having a large collection of distributions to choose from, we need to narrow our selection to a single model and a unique parameter estimate. The type of objective loss distribution (the waiting time distribution can be analyzed analogously) can easily be selected by comparing the shapes of the empirical and theoretical mean excess functions. Goodness-of-fit can be verified by plotting the corresponding limited expected value functions or drawing probability plots. Finally, the hypothesis that the modeled random event is governed by a certain loss distribution can be statistically tested (but is not discussed in this chapter; for a recent review of goodness-of-fit hypothesis testing see Burnecki et al. (2005)).

In the following subsections we will apply the abovementioned visual inference tools to PCS data. They will narrow our search for the optimal analytical model of

the loss and waiting time distributions to one or two probability laws. Moreover, they will allow for a visual assessment of the goodness-of-fit.

Mean Excess Function

6.3.1

For a random claim amount variable X , the mean excess function or mean residual life function is the expected payment per claim on a policy with a fixed amount deductible of x , where claims of less than or equal to x are completely ignored:

$$e(x) = E(X - x | X > x) = \frac{\int_x^\infty \{1 - F(u)\} du}{1 - F(x)}. \quad (6.3)$$

In practice, the mean excess function e is estimated by \hat{e}_n , based on a representative sample x_1, \dots, x_n :

$$\hat{e}_n(x) = \frac{\sum_{x_i > x} x_i}{\#\{i : x_i > x\}} - x. \quad (6.4)$$

Note that in a financial risk management context, switching from the right tail to the left tail, $e(x)$ is referred to as the expected shortfall (Weron, 2004).

When considering the shapes of mean excess functions, the exponential distribution with the cumulative distribution function (cdf) $F(x) = 1 - \exp(-\beta x)$ plays a central role. It has the memoryless property, meaning that whether or not the information $X > x$ is given, the expected value of $X - x$ is the same as if one started at $x = 0$ and calculated $E(X)$. The mean excess function for the exponential distribution is therefore constant. One can in fact easily calculate that $e(x) = 1/\beta$ for all $x > 0$ in this case.

If the distribution of X has a heavier tail than the exponential distribution, we find that the mean excess function ultimately increases, and when it has a lighter tail $e(x)$ ultimately decreases. Hence, the shape of $e(x)$ provides important information on the sub-exponential or super-exponential nature of the tail of the distribution at hand. That is why, in practice, this tool is used not only to discover the relevant class of the claim size distribution, but also to investigate all kinds of phenomena. We will apply it to data on both PCS loss and waiting times.

Mean excess functions of the well known and widely used distributional classes are given by the following formulae (selected shapes are sketched in Fig. 6.2, while the empirical mean excess functions $\hat{e}_n(x)$ for the PCS catastrophe data are plotted in Fig. 6.3):

- log-normal distribution with cdf $F(x) = \Phi\{(\ln x - \mu)/\sigma\}$:

$$e(x) = \frac{\exp\left(\mu + \frac{\sigma^2}{2}\right) \left\{1 - \Phi\left(\frac{\ln x - \mu - \sigma^2}{\sigma}\right)\right\}}{\left\{1 - \Phi\left(\frac{\ln x - \mu}{\sigma}\right)\right\}} - x,$$

where $\Phi(\cdot)$ is the standard normal (with mean 0 and variance 1) distribution function;

- Pareto distribution with cdf $F(x) = 1 - \{\lambda/(\lambda + x)\}^\alpha$:

$$e(x) = \frac{\lambda + x}{\alpha - 1}, \quad \alpha > 1;$$

- Burr distribution with cdf $F(x) = 1 - \{\lambda/(\lambda + x^\tau)\}^\alpha$:

$$e(x) = \frac{\lambda^{1/\tau} \Gamma(\alpha - \frac{1}{\tau}) \Gamma(1 + \frac{1}{\tau})}{\Gamma(\alpha)} \cdot \left(\frac{\lambda}{\lambda + x^\tau} \right)^{-\alpha} \\ \cdot \left\{ 1 - B\left(1 + \frac{1}{\tau}, \alpha - \frac{1}{\tau}, \frac{x^\tau}{\lambda + x^\tau}\right) \right\} - x,$$

where $\Gamma(\cdot)$ is the standard gamma function and $B(\cdot, \cdot, \cdot)$ is the beta function;

- Weibull distribution with cdf $F(x) = 1 - \exp(-\beta x^\tau)$:

$$e(x) = \frac{\Gamma(1 + 1/\tau)}{\beta^{1/\tau}} \left\{ 1 - \Gamma\left(1 + \frac{1}{\tau}, \beta x^\tau\right) \right\} \exp(-\beta x^\tau) - x,$$

where $\Gamma(\cdot, \cdot)$ is the incomplete gamma function;

- gamma distribution with cdf $F(x) = \int_0^x \beta(\beta s)^{\alpha-1} \exp(-\beta s)/\Gamma(\alpha) ds$:

$$e(x) = \frac{\alpha}{\beta} \cdot \frac{1 - F(x, \alpha + 1, \beta)}{1 - F(x, \alpha, \beta)} - x,$$

where $F(x, \alpha, \beta)$ is the gamma cdf;

- mixture of two exponential distributions with cdf $F(x) = a \{1 - \exp(-\beta_1 x)\} + (1 - a)\{1 - \exp(-\beta_2 x)\}$:

$$e(x) = \frac{\frac{a}{\beta_1} \exp(-\beta_1 x) + \frac{1-a}{\beta_2} \exp(-\beta_2 x)}{a \exp(-\beta_1 x) + (1 - a) \exp(-\beta_2 x)}.$$

A comparison of Figs. 6.2 and 6.3 suggests that log-normal, Pareto, and Burr distributions should provide a good fit for the loss amounts. The maximum likelihood estimates of the parameters of these three distributions are as follows: $\mu = 18.4478$ and $\sigma = 1.1554$ (log-normal), $\alpha = 1.9975$ and $\lambda = 2.3248 \times 10^8$ (Pareto), and $\alpha = 0.4135$, $\lambda = 3.235 \times 10^{18}$ and $\tau = 2.4135$ (Burr). Unfortunately, the parameters of the Burr distribution imply that the first moment is infinite, which contradicts the assumption that the random sequence of claim amounts $\{X_k\}$ has a finite mean. This assumption seems natural in the insurance world, since any premium formula usually includes the expected value of X_k . Therefore, we exclude the Burr distribution from the analysis of claim severities.

The classification of waiting time data is not this straightforward. If we discard large observations, then the log-normal and Burr laws should yield a good fit. However, if all of the waiting times are taken into account, then the empirical mean excess function $\hat{e}_n(x)$ approximates a straight line (although one that oscillates somewhat), which suggests that the exponential law could be a reasonable alternative.

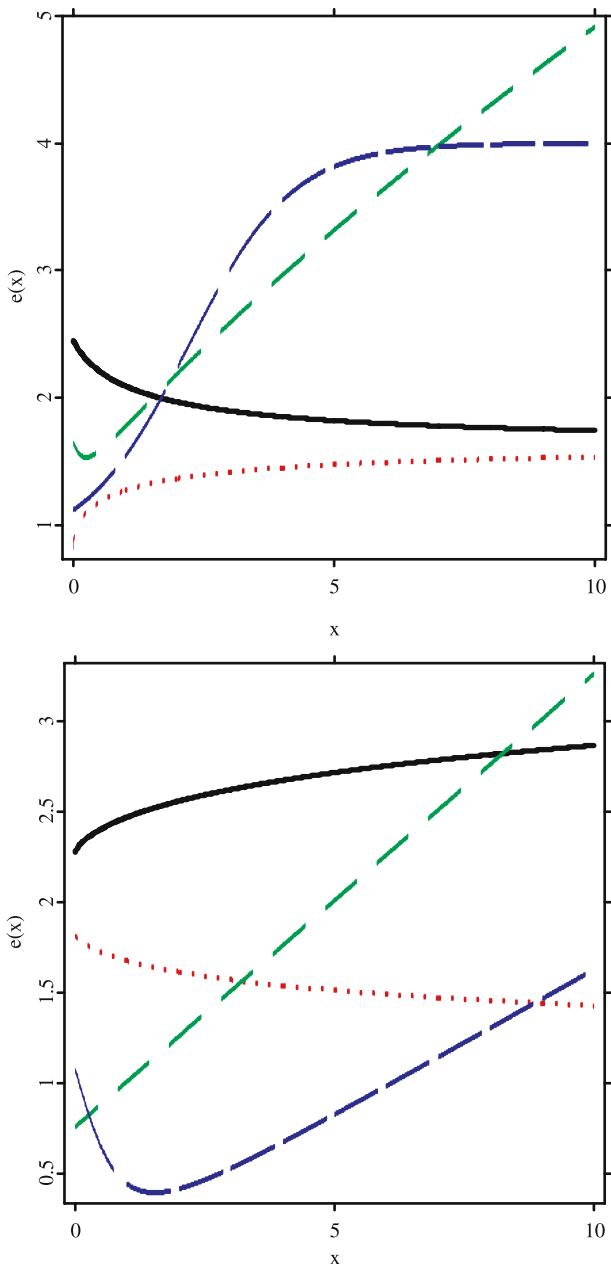


Figure 6.2. Top panel: shapes of the mean excess function $e(x)$ for the log-normal (dashed line), gamma with $\alpha < 1$ (dotted line), gamma with $\alpha > 1$ (solid line) and a mixture of two exponential distributions (long-dashed line). Bottom panel: shapes of the mean excess function $e(x)$ for the Pareto (dashed line), Burr (long-dashed line), Weibull with $\tau < 1$ (solid line) and Weibull with $\tau > 1$ (dotted line) distributions. From XploRe

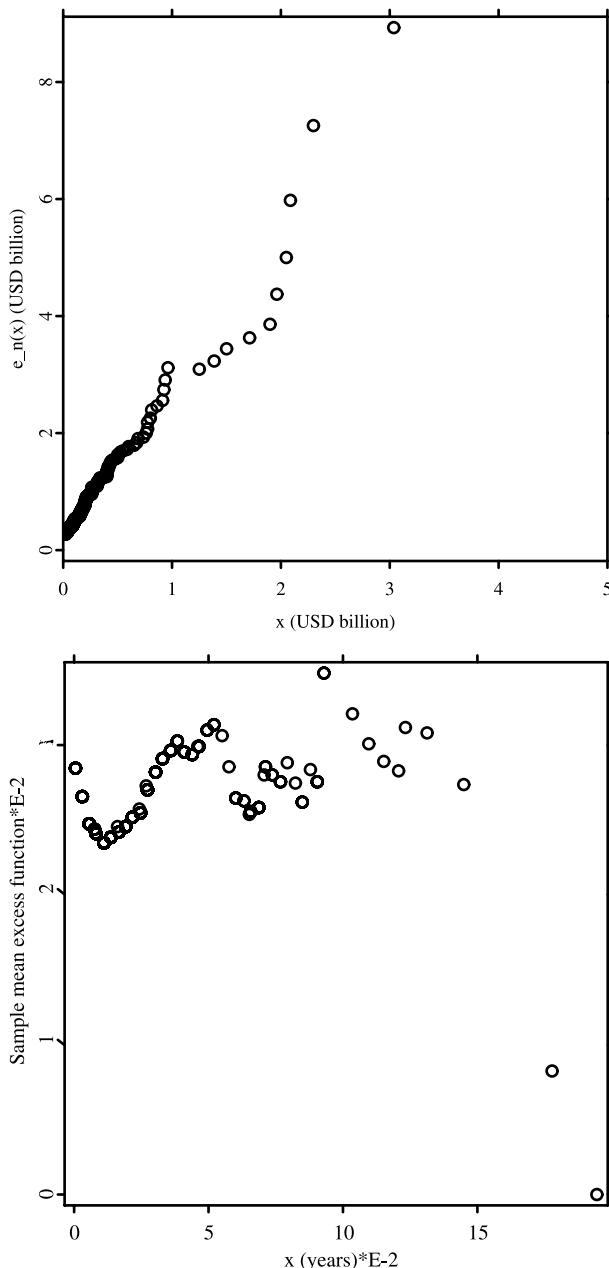


Figure 6.3. The empirical mean excess function $\hat{e}_n(x)$ for the PCS catastrophe loss amounts in billions of USD (top panel) and waiting times in years (bottom panel). Comparison with Fig. 6.2 suggests that log-normal, Pareto, and Burr distributions should provide a good fit for loss amounts, while log-normal, Burr, and exponential laws work well for the waiting times. From: XploRe

Limited Expected Value Function

6.3.2

The limited expected value function L of a claim size variable X , or of the corresponding cdf $F(x)$, is defined as

$$L(x) = E\{\min(X, x)\} = \int_0^x y dF(y) + x \{1 - F(x)\}, \quad x > 0. \quad (6.5)$$

The value of the function L at point x is equal to the expectation of the cdf $F(x)$ truncated at this point. In other words, it represents the expected amount per claim retained by the insured on a policy with a fixed amount deductible of x . The empirical estimate is defined as follows:

$$\hat{L}_n(x) = \frac{1}{n} \left(\sum_{x_j < x} x_j + \sum_{x_j \geq x} x \right). \quad (6.6)$$

In order to fit the limited expected value function L of an analytical distribution to the observed data, the estimate \hat{L}_n is first constructed. Thereafter, one attempts to find a suitable analytical cdf F such that the corresponding limited expected value function L is as close to the observed \hat{L}_n as possible.

The limited expected value function (LEV) has the following important properties:

- (i) the graph of L is concave, continuous and increasing
- (ii) $L(x) \rightarrow E(X)$ as $x \rightarrow \infty$
- (iii) $F(x) = 1 - L'(x)$, where $L'(x)$ is the derivative of the function L at point x ; if F is discontinuous at x , then the equality holds true for the right-hand derivative $L'(x+)$.

The limited expected value function is a particularly suitable tool for our purposes because it represents the claim size distribution in the monetary dimension. For example, we have $L(\infty) = E(X)$ if it exists. The cdf F , on the other hand, operates on the probability scale, i.e., it takes values of between 0 and 1. Therefore, it is usually difficult to work out how sensitive the price of the insurance – the premium – is to changes in the values of F by looking only at $F(x)$, while the LEV immediately shows how different parts of the claim size cdf contribute to the premium. Aside from its applicability to curve-fitting, the function L also turns out to be a very useful concept when dealing with deductibles Burnecki et al. (2005). It is also worth mentioning that there is a connection between the limited expected value function and the mean excess function:

$$E(X) = L(x) + P(X > x)e(x). \quad (6.7)$$

The limited expected value functions (LEVs) for all of the distributions considered in this chapter are given by the following formulae:

- exponential distribution:

$$L(x) = \frac{1}{\beta} \{1 - \exp(-\beta x)\};$$

- log-normal distribution:

$$L(x) = \exp\left(\mu + \frac{\sigma^2}{2}\right) \Phi\left(\frac{\ln x - \mu - \sigma^2}{\sigma}\right) + x \left\{1 - \Phi\left(\frac{\ln x - \mu}{\sigma}\right)\right\};$$

- Pareto distribution:

$$L(x) = \frac{\lambda - \lambda^\alpha (\lambda + x)^{1-\alpha}}{\alpha - 1};$$

- Burr distribution:

$$L(x) = \frac{\lambda^{1/\tau} \Gamma(\alpha - \frac{1}{\tau}) \Gamma(1 + \frac{1}{\tau})}{\Gamma(\alpha)} B\left(1 + \frac{1}{\tau}, \alpha - \frac{1}{\tau}; \frac{x^\tau}{\lambda + x^\tau}\right) + x \left(\frac{\lambda}{\lambda + x^\tau}\right)^\alpha;$$

- Weibull distribution:

$$L(x) = \frac{\Gamma(1 + 1/\tau)}{\beta^{1/\tau}} \Gamma\left(1 + \frac{1}{\tau}, \beta x^\alpha\right) + x e^{-\beta x^\alpha};$$

- gamma distribution:

$$L(x) = \frac{\alpha}{\beta} F(x, \alpha + 1, \beta) + x \{1 - F(x, \alpha, \beta)\};$$

- mixture of two exponential distributions:

$$L(x) = \frac{a}{\beta_1} \{1 - \exp(-\beta_1 x)\} + \frac{1-a}{\beta_2} \{1 - \exp(-\beta_2 x)\}.$$

From a curve-fitting point of view, the advantage of using the LEVF_s rather than the cdfs is that both the analytical function and the corresponding observed function \hat{L}_n , based on the observed discrete cdf, are continuous and concave, whereas the observed claim size cdf F_n is a discontinuous step function. Property (iii) implies that the limited expected value function determines the corresponding cdf uniquely. When the limited expected value functions of two distributions are close to each other, not only are the mean values of the distributions close to each other, but the whole distributions are too.

Since the LEVF represents the claim size distribution in the monetary dimension, it is usually used exclusively to analyze price data. In Fig. 6.4, we depict the empirical and analytical LEVF_s for the two distributions that best fit the PCS catastrophe loss amounts (as suggested by the mean excess function). We can see that the Pareto law is definitely superior to the log-normal one.

6.3.3 Probability Plot

The purpose of the probability plot is to graphically assess whether the data comes from a specific distribution. It can provide some assurance that this assumption is not being violated, or it can provide an early warning of a problem with our assumptions.

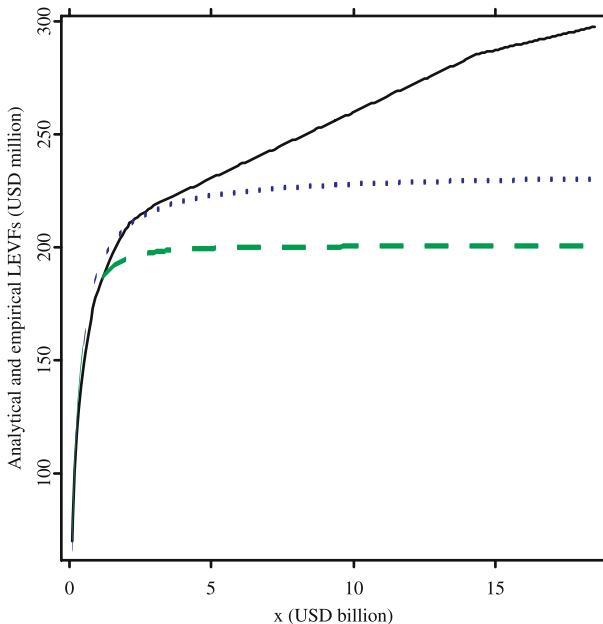


Figure 6.4. The empirical (solid line) and analytical limited expected value functions (LEVFs) for the log-normal (dashed line) and Pareto (dotted line) distributions for the PCS loss catastrophe data. From XploRe

The probability plot is constructed in the following way. First, the observations x_1, \dots, x_n are ordered from the smallest to the largest: $x_{(1)} \leq \dots \leq x_{(n)}$. Next, they are plotted against their observed cumulative frequency, i.e., the points (the crosses in Figs. 6.5–6.8) correspond to the pairs $(x_{(i)}, F^{-1}([i - 0.5]/n))$, for $i = 1, \dots, n$. If the hypothesized distribution F adequately describes the data, the plotted points fall approximately along a straight line. If the plotted points deviate significantly from a straight line, especially at the ends, then the hypothesized distribution is not appropriate.

Figure 6.5 shows a Pareto probability plot of the PCS loss data. Apart from the two very extreme observations (corresponding to Hurricane Andrew and Northridge Earthquake), the points more or less constitute a straight line, validating the choice of the Pareto distribution. Figures 6.6 and 6.7 present log-normal probability plots of the PCS data. To this end we applied the standard normal probability plots to the logarithms of the losses and waiting times, respectively. Figure 6.6 suggests that the log-normal distribution for the loss severity is not the best choice, whereas Fig. 6.7 justifies the use of that particular distribution for the waiting time data. Figure 6.8 depicts an exponential probability plot of the latter dataset. We can see that the exponential distribution is not a very good candidate for the underlying distribution of the waiting time data – the points deviate from a straight line at the far end. Nevertheless, the deviation is not that large, and the exponential law may be an acceptable model.

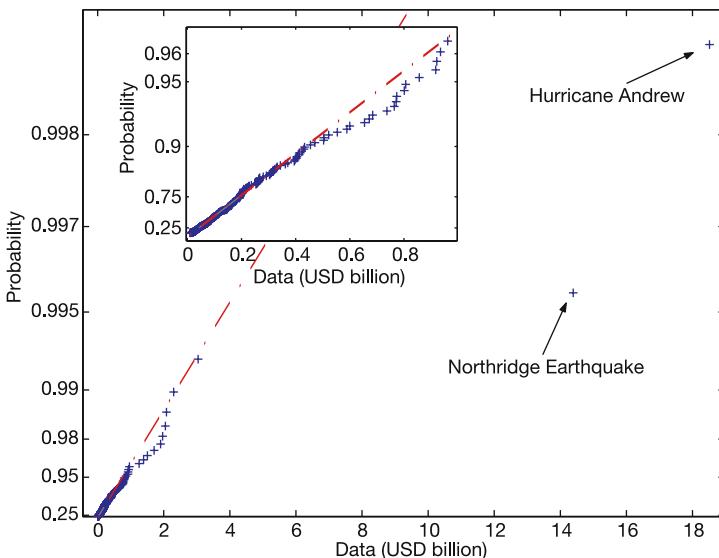


Figure 6.5. Pareto probability plot of the PCS loss data. Apart from the two very extreme observations (Hurricane Andrew and Northridge Earthquake), the points (crosses) more or less constitute a straight line, validating the choice of the Pareto distribution. The *inset* is a magnification of the bottom left part of the original plot. From the Ruin Probabilities Toolbox

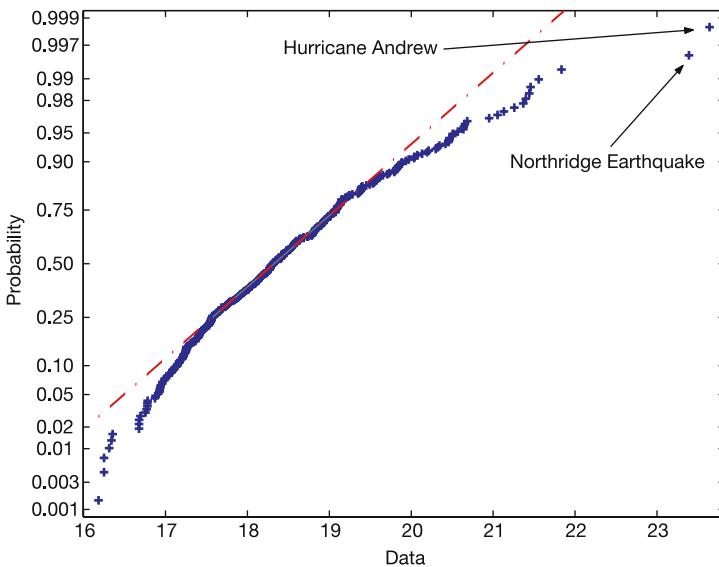


Figure 6.6. Log-normal probability plot of the PCS loss data. The x -axis corresponds to logarithms of the losses. The deviations from the straight line at both ends question the adequacy of the log-normal law. From the Ruin Probabilities Toolbox

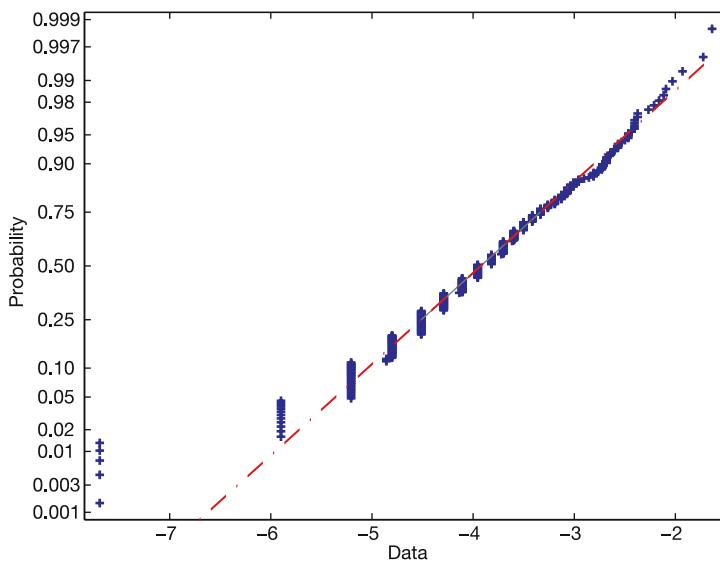


Figure 6.7. Log-normal probability plot of the PCS waiting time data. The x -axis corresponds to logarithms of the losses. From the Ruin Probabilities Toolbox

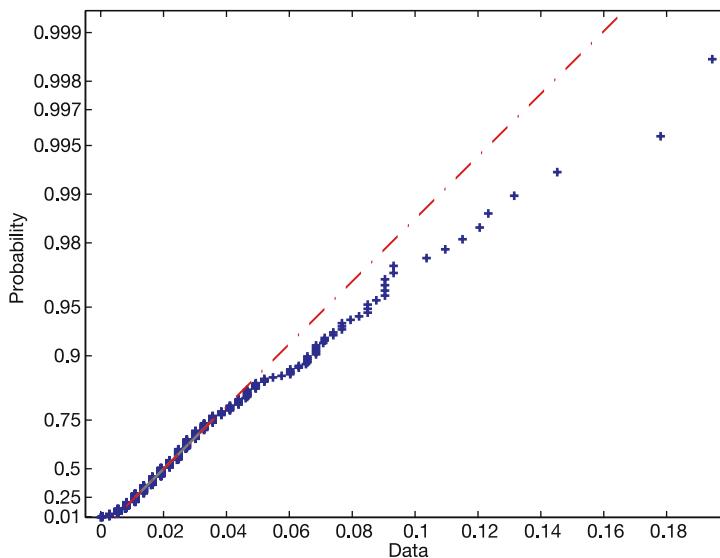


Figure 6.8. Exponential probability plot of the PCS waiting time data. The plot deviates from a straight line at the far end. From the Ruin Probabilities Toolbox

These probability plots suggest that, as far as the loss amounts are concerned, the Pareto law provides a much better fit than the log-normal distribution. In fact, apart from the two very extreme observations (Hurricane Andrew and Northridge Earth-

quake), it does provide a very good fit. Note that the procedure of trimming the top 1–5 % of the data before calibration is known as “robust estimation” and it leads to very similar conclusions (see a recent paper by Chernobai et al., 2006).

From the probability plots, we can also infer that the waiting time data can be described by the log-normal and – to some degree – the exponential distribution. The maximum likelihood estimates of the parameters of these two distributions are given by $\mu = -3.9431$ and $\sigma = 0.9685$ (log-normal) and $\beta = 34.572$ (exponential).

6.4

Risk Process and its Visualization

The risk process (6.2) is the sum of the initial capital and the premium function minus the aggregate claim process governed by two random phenomena – the severity and incidence of claims. In many practical situations, it is reasonable to consider the counting process N_t (responsible for the incidence of events) to be (i) a renewal process, i.e., a counting process with interarrival times that are i.i.d. positive random variables with a mean of $1/\lambda$, and (ii) independent of the claim severities $\{X_k\}$. In such a case, the premium function can be defined in a natural way as $c(t) = (1 + \theta)\mu\lambda t$, where μ is the expectation of X_k and $\theta > 0$ is the relative safety loading on the premium which “guarantees” the survival of the insurance company. The (homogeneous) Poisson process (HPP) is a special case of the renewal process with exponentially distributed waiting times.

Two standard ways of presenting sample trajectories of the risk process are displayed in Figs. 6.9 and 6.10. Here, we use the Danish fire losses dataset, which can be modeled by a log-normal claim amount distribution with parameters $\mu = 12.6795$ and $\sigma = 1.4241$ (obtained via maximum likelihood) and a HPP counting process with a monthly intensity $\lambda = 4.81$. The company’s initial capital is assumed to be $u = 10$ million DKK. In Fig. 6.9, five real (discontinuous) sample realizations of the resulting risk process are presented, whereas in Fig. 6.10 the trajectories are shown in a continuous fashion. The latter way of depicting sample realizations seems to be more illustrative. Also note that one of the trajectories falls below zero, indicating a scenario leading to bankruptcy of the insurance company.

6.4.1

Ruin Probability Plots

When examining the nature of the risk associated with a business portfolio, it is often interesting to assess how the portfolio may be expected to perform over an extended period of time. This is where the ruin theory (Grandell, 1991) comes in handy. Ruin theory is concerned with the excess of the income $c(t)$ (with respect to a business portfolio) over the outgoings, or claims paid, $S(t)$. This quantity, referred to as the insurer’s surplus, varies over time. Specifically, ruin is said to occur if the insurer’s surplus reaches a specified lower bound, e.g., minus the initial capital. This can be observed in Fig. 6.10, where the time of ruin is denoted by a star. One measure of risk

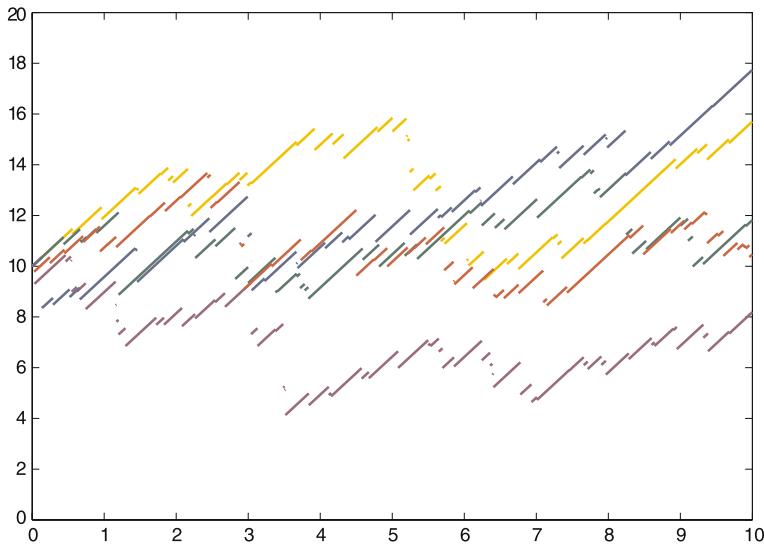


Figure 6.9. Discontinuous visualization of the trajectories of a risk process. The initial capital $u = 10$ million DKK, the relative safety loading $\theta = 0.05$, the claim size distribution is log-normal with parameters $\mu = 12.6795$ and $\sigma = 1.4241$, and the driving counting process is a HPP with monthly intensity $\lambda = 4.81$. From the Ruin Probabilities Toolbox

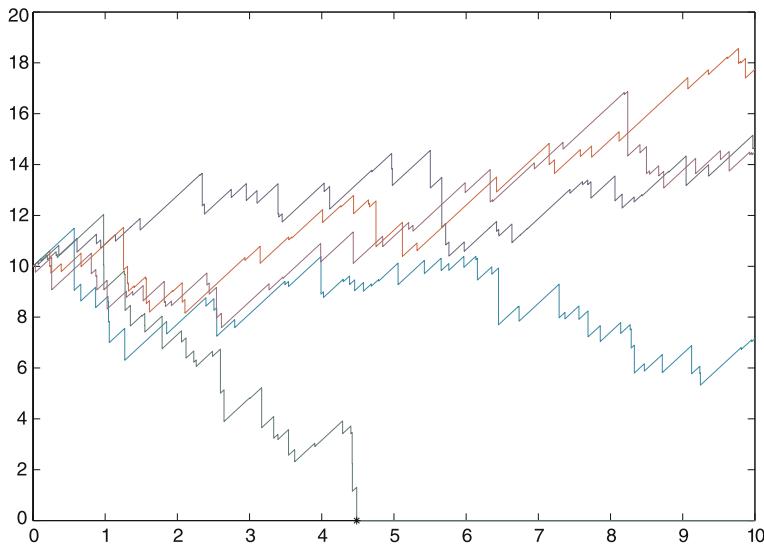


Figure 6.10. Alternative (continuous) visualization of the trajectories of a risk process. The bankruptcy time is denoted by a star. The parameters of the risk process are the same as in Fig. 6.9. From the Ruin Probabilities Toolbox

is the probability of such an event, which clearly reflects the volatility inherent in the business. In addition, it can serve as a useful tool when planning how the insurer's funds are to be used over the long term.

The ruin probability in a finite time T is given by

$$\psi(u, T) = P \left(\inf_{0 < t < T} \{R_t\} < 0 \right). \quad (6.8)$$

Most insurance managers will closely follow the development of the risk business and increase the premium if the business behaves badly. The planning horizon may be thought of as the sum of the following: the time until the risk business is found to behave "badly," the time until the management reacts, and the time until the decision to increase the premium takes effect (Grandell, 1991).

In Fig. 6.11, a three-dimensional (3-D) visualization of the ruin probability with respect to the initial capital u (varying from zero to eight million DKK) and the time horizon T (ranging from zero to five months) is depicted. The remaining parameters of the risk process are the same as those used in Figs. 6.9 and 6.10, except that the relative safety loading was raised from $\theta = 0.05$ to $\theta = 0.15$. We can see that the ruin probability increases with the time horizon and decreases as the initial capital grows.

The ruin probability in finite time can always be computed directly using Monte Carlo simulations. Naturally, the choice of the intensity function and the distribution

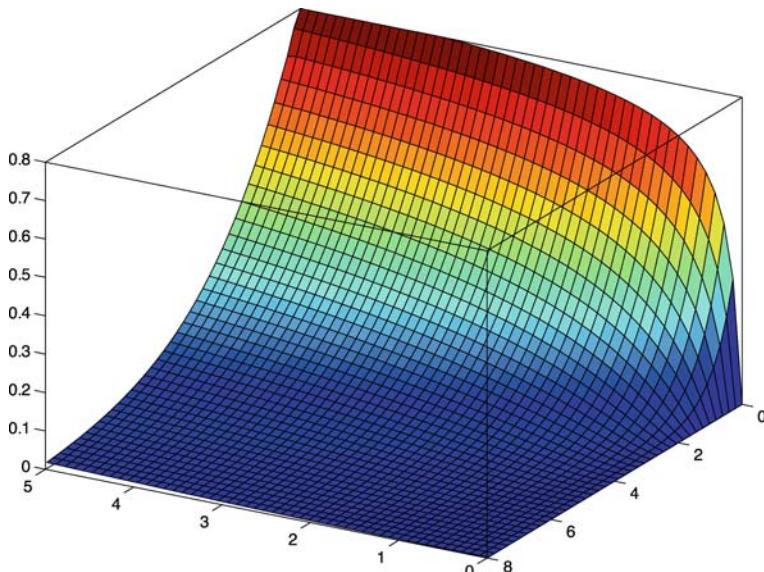


Figure 6.11. Ruin probability plot with respect to the time horizon T (left axis, in months) and the initial capital u (right axis, in million DKK). The relative safety loading $\theta = 0.15$; other parameters of the risk process are the same as in Fig. 6.9. From the Ruin Probabilities Toolbox

of claim severities heavily affects the simulated values and the ruin probability. The graphical tools presented below can help in assessing whether the choices made result in a reasonable risk process and, hence, greatly reduce the time needed to construct an adequate model.

Density Evolution

6.4.2

Density evolution plots (and their two-dimensional projections) are a visually attractive method of representing the time evolution of a process. At each time point $t = t_0, t_1, \dots, t_n$, a density estimate of the distribution of process values at this time point is evaluated; see Fig. 6.12 (the same parameters of the risk process as those in Fig. 6.11 were used). Then the densities are plotted on a grid of t values. The three-dimensional surface obtained can be further rotated to better present the behavior of the process over a particular interval.

A two-dimensional projection of the density evolution plot (see Fig. 6.13) reveals equivalent information to that represented by the quantile lines (see Sect. 6.4.3). However, in this case, the presented visual information is more attractive to the eye, but not that rigid. Depending on the discretization of the time and process value intervals and the kernel density estimate used, slightly different density evolution plots can be obtained (Janicki and Izydorczyk, 2001; Miśta, 2002).

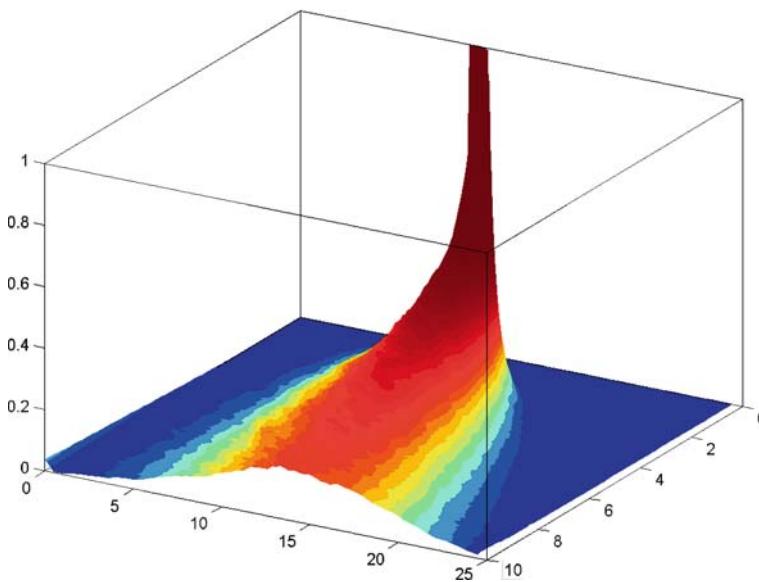


Figure 6.12. [This figure also appears in the color insert.] Three-dimensional visualization of the density evolution of a risk process with respect to the risk process value R_t (left axis) and time t (right axis). The parameters of the risk process are the same as in Fig. 6.11. From the Ruin Probabilities Toolbox

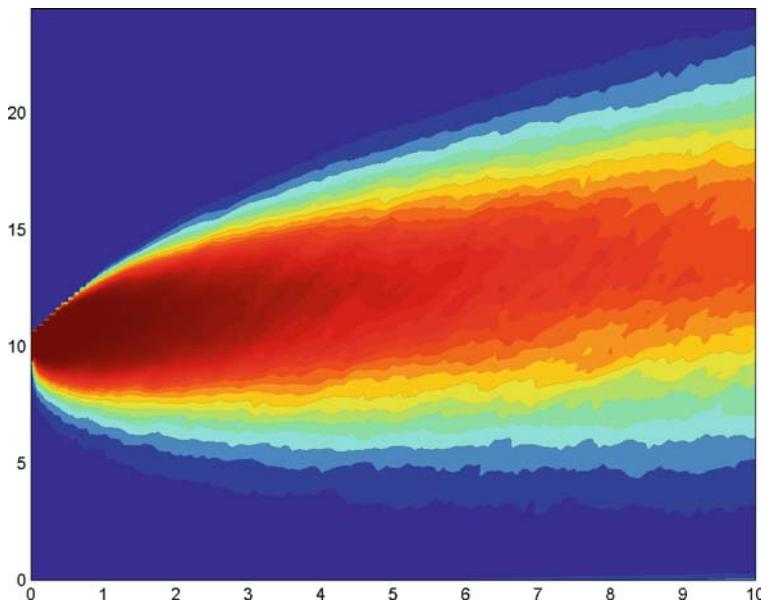


Figure 6.13. Two-dimensional projection of the density evolution depicted in Fig. 6.12. From the Ruin Probabilities Toolbox

6.4.3 Quantile Lines

The function $\hat{x}_p(t)$ is called a sample p -quantile line if for each $t \in [t_0, T]$, $\hat{x}_p(t)$ is the sample p -quantile, i.e., if it satisfies $F_n(x_{p-}) \leq p \leq F_n(x_p)$, where F_n is the empirical distribution function (edf). Recall that for a sample of observations $\{x_1, \dots, x_n\}$, the edf is defined as

$$F_n(x) = \frac{1}{n} \#\{i : x_i \leq x\}, \quad (6.9)$$

in other words, it is a piecewise constant function with jumps of size $1/n$ at points x_i ; Burnecki et al. (2005).

Quantile lines are a very helpful tool in the analysis of stochastic processes. For example, they can provide a simple justification of the stationarity of a process (or the lack of it); (see Janicki and Weron, 1994). In Figs. 6.14, 6.15, and 6.16, they visualize the evolution of the risk process.

Quantile lines can be also a useful tool for comparing two processes; see Fig. 6.14. It depicts quantile lines and two sample trajectories of the risk process and its diffusion approximation; consult Burnecki et al. (2005) for a discussion of different approximations in the context of ruin probability. The parameters of the risk process are the same as in Fig. 6.11. We can see that the quantile lines of the risk process and its approximation coincide. This justifies the use of the Brownian approximation for these parameters of the risk process.

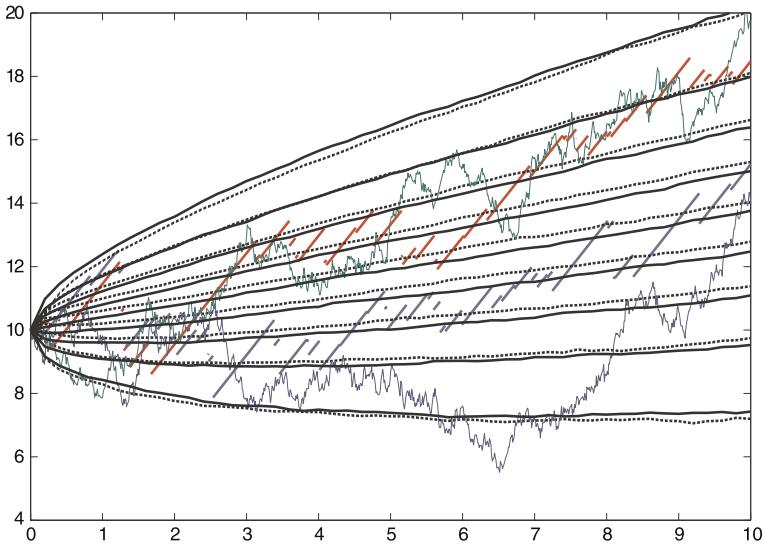


Figure 6.14. [This figure also appears in the color insert.] A Poisson-driven risk process (*discontinuous thin lines*) and its Brownian motion approximation (*continuous thin lines*). The quantile lines enable an easy and fast comparison of the processes. The *thick solid lines* represent the sample $0.1, \dots, 0.9$ -quantile lines based on 10 000 trajectories of the risk process, whereas the *thick dashed lines* correspond to their approximation counterparts. The parameters of the risk process are the same as in Fig. 6.11. From the Ruin Probabilities Toolbox

We now return to the PCS dataset. To study the evolution of the risk process we simulate sample trajectories and compute quantile lines. We consider a hypothetical scenario where the insurance company insures losses resulting from catastrophic events in the United States. The company's initial capital is assumed to be $u = 100$ billion USD and the relative safety loading used is $\theta = 0.5$. We choose two different models of the risk process based on the results from a statistical analysis (see Sect. 6.3): a homogeneous Poisson process (HPP) with log-normal claim sizes, and a renewal process with Pareto claim sizes and log-normal waiting times. The results are presented in Fig. 6.15. The thick solid line is the “real” risk process, i.e., a trajectory constructed from the historical arrival times and values of the losses. The different shapes of the “real” risk process in the subplots are due to the different forms of the premium function $c(t)$. The thin solid line is a sample trajectory. The dotted lines are the sample 0.001, 0.01, 0.05, 0.25, 0.50, 0.75, 0.95, 0.99, 0.999-quantile lines based on 3000 trajectories of the risk process. The quantile lines visualize the evolution of the density of the risk process. Clearly, if the claim severities are Pareto-distributed then extreme events are more likely to happen than in the log-normal case, where the historical trajectory falls outside even the 0.001-quantile line. Figure 6.15 suggests that the second model (Pareto-distributed claim sizes and log-normal waiting times) yields a reasonable model for the “real” risk process.

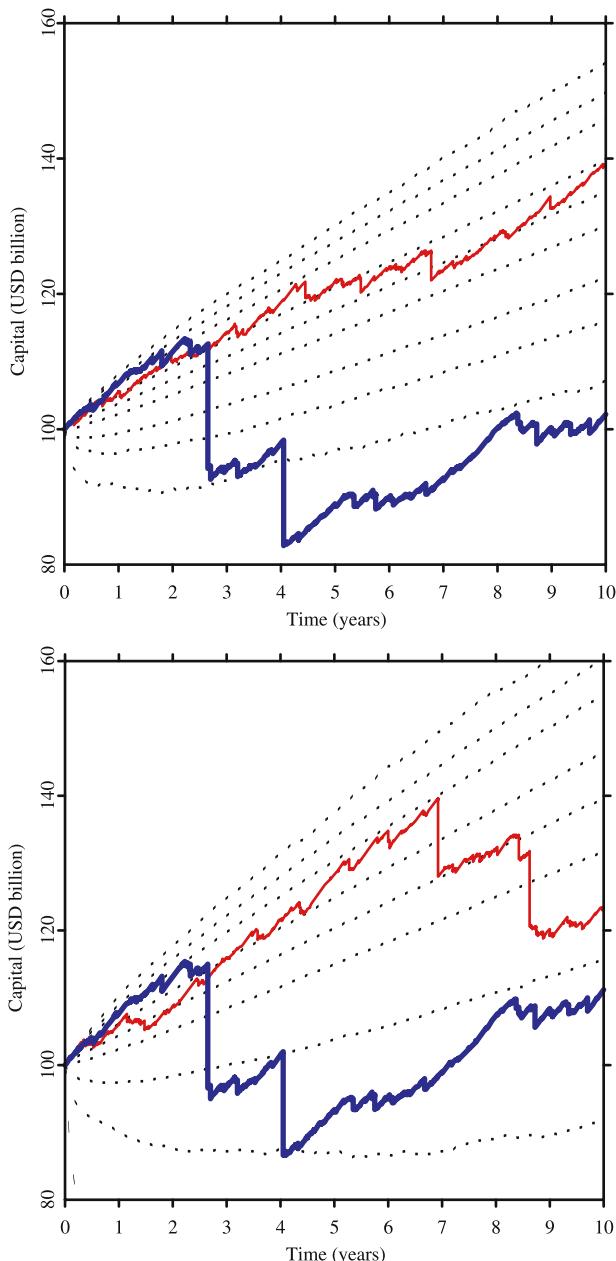


Figure 6.15. The PCS data simulation results for a homogeneous Poisson process with log-normal claim sizes (*top panel*) and a renewal process with Pareto claim sizes and log-normal waiting times (*bottom panel*). The dotted lines are the sample 0.001, 0.01, 0.05, 0.25, 0.50, 0.75, 0.95, 0.99, 0.999-quantile lines based on 3000 trajectories of the risk process. From XploRe

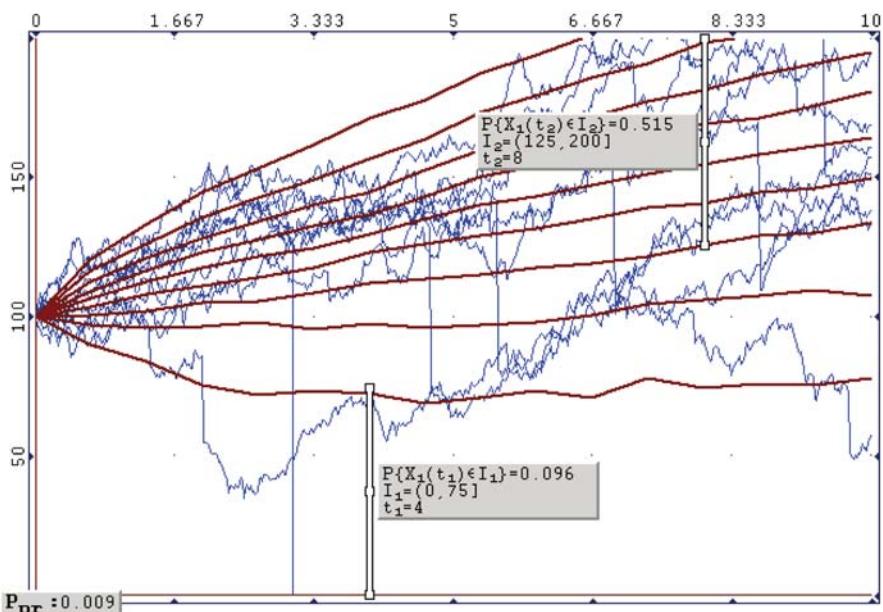


Figure 6.16. “Probability gates” are an interactive graphical tool used for determining the probability that the process passes through a specified interval. The $0, \dots, 0.9$ -quantile lines (thick lines) are based on 1000 simulated trajectories (thin lines) of the risk process originating at $u = 100$ billion USD. The parameters of the α -stable Lévy motion approximation of the risk process were chosen to comply with PCS data. From the SDE-Solver

Probability Gates

6.4.4

“Probability gates” are an interactive graphical tool implemented in SDE-Solver. They can provide invaluable assistance in the real-time analysis of the risk process and its models. A “probability gate” gives the so-called cylindrical probability $P\{X_{t_0} \in (a, b]\}$ that the simulated process X_t passes through a specified interval $(a, b]$ at a specified point in time t_0 (Janicki and Izdyorczyk, 2001; Janicki et al., 2003). Two probability gates are defined in Fig. 6.16; Monte Carlo simulations can be used to obtain the probability estimates. One yields the probability of the risk process falling below 75 billion after four years, i.e., $P\{R_4 \in (0, 75]\} = 0.096$. The other yields the probability of the risk process being in the range $(125, 200]$ billion after eight years, i.e., $P\{R_8 \in (125, 200]\} = 0.515$. Additionally, product probabilities of the process passing through all of the defined gates are provided. In the above example, the product probability of the risk process first falling below 75 billion (after four years) and then recovering to over 125 billion but less than 200 billion (after eight years) is equal to $P\{R_4 \in (0, 75], R_8 \in (125, 200]\} = 0.009$. The parameters of the α -stable Lévy motion approximation (Furrer et al., 1997) of the risk process were chosen to comply with the catastrophic data example.

References

- Burnecki, K., Härdle, W. and Weron, R. (2004). Simulation of risk processes, in Teugels, J., Sundt, B. (eds) *Encyclopedia of Actuarial Science*, Wiley, Chichester, UK 1564–1570.
- Burnecki, K., Misiorek, A. and Weron, R. (2005). Loss Distributions, in Cizek, P., Härdle, W., Weron, R. (eds) *Statistical Tools for Finance and Insurance*, Springer, Berlin, 289–317.
- Burnecki, K., Miśta, P. and Weron, A. (2005). Ruin Probabilities in Finite and Infinite Time, in Cizek, P., Härdle, W., Weron, R. (eds) *Statistical Tools for Finance and Insurance*, Springer, Berlin, 341–379.
- Burnecki, K., Nowicka-Zagrajek, J. and Wyłomańska, A. (2005). Pure Risk Premiums under Deductibles, in Cizek, P., Härdle, W., Weron, R. (eds) *Statistical Tools for Finance and Insurance*, Springer, Berlin, 427–452.
- Čížek, P., Härdle, W. and Weron, R. (eds) (2005). *Statistical Tools for Finance and Insurance*, Springer, Berlin.
- Chernobai, A., Burnecki, K., Rachev, S.T., Trück, S. and Weron, R. (2006). Modelling catastrophe claims with left-truncated severity distributions, *Computational Statistics*, 21:537–555.
- Daykin, C.D., Pentikainen, T. and Pesonen, M. (1994). *Practical Risk Theory for Actuaries*, Chapman, London.
- Grandell, J. (1991). *Aspects of Risk Theory*, Springer, New York.
- Furrer, H., Michna, Z. and Weron, A. (1997). Stable Lévy motion approximation in collective risk theory, *Insurance: Mathematics and Economics*, 20:97–114.
- Janicki, A. and Izydorczyk, A. (2001). *Computer Methods in Stochastic Modeling*, WNT, Warszawa (in Polish).
- Janicki, A., Izydorczyk, A. and Gradalski, P. (2003). Computer Simulation of Stochastic Models with SDE-Solver Software Package, *Lecture Notes in Computer Science*, 2657:361–370.
- Janicki, A. and Weron, A. (1994). *Simulation and Chaotic Behavior of α -Stable Stochastic Processes*, Marcel Dekker, New York.
- Kloeden, P.E. and Platen, E. (1995). *Numerical Solution of Stochastic Differential Equations*, Springer, Berlin.
- Klugman, S.A., Panjer, H.H. and Willmot, G.E. (1998). *Loss Models: From Data to Decisions*, Wiley, New York.
- Miśta, P. (2002). *Computer methods of ruin probability approximation and its control via optimal reinsurance policies*, Fundacja "Warta", Warsaw (in Polish).
- Panjer, H.H. and Willmot, G.E. (1992). *Insurance Risk Models*, Society of Actuaries, Chicago, IL.
- Teugels, J. and Sundt, B. (2004). *Encyclopedia of Actuarial Science*, Wiley, Chichester, UK.
- Weron, R. (2004). Computationally Intensive Value at Risk Calculations, in Gentle, J.E., Härdle, W., Mori, Y. (eds) *Handbook of Computational Statistics*, Springer, Berlin, 911–950.

Subject Index

- α -blending, 70, 839
- 3D plot, 777, 778
- Abbott, Edwin, 30
- acceleration, 884
- accuracy ratio, 860
- active contour model, 815
- active plot, 203
- acyclic graph, 123, 245
- adaptive smoothing, 472, 489
- adaptive weights smoothing, 476
- additive models, 507–511
- additive tree, 124
- adjunct variables, 158
- adverse events, 442–444
- age pyramid, 34
- aggregate claim process, 900
- aggregated view, 222
- aggregated views, linking, 227
- AIC, *see* Akaike's information criterion
- Akaike's information criterion, 448, 497, 502, 503
- algebraic linking, 208
- algebraic reconstruction technique, 817
- alignment, 64
- analyse des données, 351
- analysis
 - discriminant, 41
 - principal component, 41
- analysis of covariance, 431, 432
- Anderberg, 318
- angular separation, 317
- anomaly, 875, 878
- anti-Robinson, 688
- approximate degrees of freedom, 497, 498, 502, 503, 508, 509
- arithmetic
 - political, 20, 21
- ART, *see* algebraic reconstruction technique
- aspect ratio, 68, 75
- association plot, 590, 596
- association rules, 779
- atlas
 - cartographic, 19
 - statistical, 33, 36
- atomic number, 38
- attribute operations, 891
- attributes, 891
- auction, 879, 888
- auction profiles, 876
- average linkage, 693, 703
- averaged shifted histogram, 393, 409
- axis variables, 157
- B-spline, 877
- backfitting algorithm, 507, 509
- background variable, 576
- bagging, 257
- Baker, Robert, 27
- balloon plot, 577
- bandwidth, 394, 396, 404, 407, 408
 - smoothing parameter, 494
 - variable, 399
- bankruptcy analysis, 854

-
- bar chart, 24, 36, 43, 63, 77, 875
 - divided, 28
 - Barbeu-Dubourg, Jacques, 22
 - barplot, 574
 - baseball data, 239
 - Bayesian data analysis, 710
 - Bayesian estimate, 817
 - Bayesian exploratory data analysis, 712
 - Bayesian information criterion, 581
 - Bayesian networks, 794
 - Bayesian statistics, 710
 - Bell Laboratories, 40
 - Benzécri, Jean-Paul, 39, 351
 - Bertin plot, 629
 - Bertin, Jacques, 39, 372
 - Bhattacharya distance, 317
 - bias, 183, 195, 498–500
 - grouping-based, 184
 - perception-based, 183
 - proximity-based, 184
 - bias-optimized frequency polygon, 393
 - BIC, *see* Bayesian information criterion
 - bid histories, 891
 - bid-arrival, 878
 - bidding data, 876, 879
 - bidirectional color, 685
 - bidirectional linking, 213
 - bilateral frequency polygon, 34
 - bills of mortality, 21
 - binary data, 700, 704
 - binary response model, 472, 481
 - binwidth, 66
 - biplot, 40
 - bivariate normal surface, 47
 - block clustering, 682
 - Boolean networks, 794
 - bootstrap, 41, 257–259, 860
 - Boston housing, 449
 - Bowley, Arthur, 37, 38
 - Box–Cox transformation, 427
 - boxplot, 39, 63, 434, 437–442, 444, 691, 728, 736, 875, 893
 - Brahe, Tycho, 19–21, 32
 - Braun, Blanque, 318
 - Bray–Curtis, 317
 - brush mode, 225
 - brushing, 40, 161, 222, 890
 - Buache, Philippe, 22
 - Bundesbank, 855, 856, 862, 866
 - C (programming language), 40, 98
 - Caesarean data set, 155
 - calendar plots, 875
 - calendar time, 888
 - calendar-based, 875
 - camera obscura, 19
 - Canberra metric, 317
 - cancer atlas, 75
 - canonical correlation analysis, 540, 551, 553
 - kernel, 542, 543, 551–553
 - reduced kernel, 552
 - caption, 68
 - cars data set, 157, 166
 - CART, 254, 257, 258, 451, 455
 - cartesian product, 417–419, 422, 425, 431, 433, 435, 437
 - cartesian space, 352–354
 - case-by-variables matrices, 629
 - catastrophic event, 901
 - categorical data, 590
 - CCmaps, 288
 - cDNA, 795, 802, 807
 - cell-cycle, 696
 - censored zooming, 246
 - centroid, 568
 - centroid-based cluster analysis, *see* cluster analysis
 - chart
 - bar, 24, 28, 36, 43
 - control, 37
 - Gantt, 37
 - isochronic, 32
 - pie, 24
 - time-series, 42
 - Cheysson, Émile, 31, 33
 - chi-square distance, 338
 - chi-squared test, 454
 - choropleth, 75
 - choropleth maps, 270
 - city block metric, 317

-
- claim arrival process, 900
 - class diagram, 734
 - classical scaling, 319
 - classification, 761, 771
 - classification in data, 664–670
 - classification tree, 244
 - cluster analysis, 362, 540–542, 554–556, 562, 797
 - centroid-based, 568
 - computational issues, 366
 - furthest neighbor, 363
 - hierarchical, 362, 555, 556, 564
 - mixed strategy, 366, 370
 - – large datasets, 368
 - model-based, 580
 - nearest neighbor, 363
 - nested partitions, 363
 - nonhierarchical, 366
 - partitioning, 362, 366, 568
 - – large dataset, 368
 - single linkage, 363
 - tree, *see* dendrogram
 - – alternative views, 374
 - – structure, 365
 - cluster tree, 127
 - cluster views linked to statistical charts, 236
 - clustering, 40, 762, 771, 778, 779, 797
 - collinearity, 449
 - color choice, 425, 435, 437, 438, 440, 441, 443
 - color histogram, 683
 - color images, 473, 483
 - color palette
 - diverging, 599
 - qualitative, 599
 - color projection, 704
 - color space
 - CIELAB, 600
 - CIELUV, 600
 - HCL, 600
 - HSV, 599
 - RGB, 599
 - color spectrum, 684, 685
 - Colorbrewer, 68, 425
 - colour, 68
 - colour coding, 854, 867
 - comma separated value (CSV) format, 728
 - command line, 84, 92, 98
 - common scale, 162
 - common scaling, 66, 73
 - company rating, 854
 - complete graph, 122
 - compositing operator, 86
 - composition, 733
 - Comprehensive R Archive Network, 564
 - computational
 - complexity, 366
 - – clustering, 366
 - time, 366
 - concordance measures, 148
 - concurrency, 876, 888, 897
 - conditional correlation, 705
 - conditional independence, 608, 797
 - conditional inference, 604
 - conditioned choropleth maps, 288
 - conditioning variables, 158
 - confidence bounds, 882, 893
 - confidence curves, 882
 - confidence region, 582
 - consistency, 59, 795
 - contingency table, 577, 590
 - continuous distribution function, 21
 - contour plot, 404, 405, 408, 458, 467, 868
 - trivariate, 408
 - contour shell, 408
 - convex hull, 138, 569
 - Cook's distance, 429
 - coordinate
 - parallel, 31
 - coordinate paper, 24
 - coordinate system, 21, 90, 93, 96
 - coplot, 408
 - correlation, 32, 317, 449, 797, 799–801
 - marginal, 452
 - correlation matrix, 582, 689
 - correspondence analysis, 117, 338
 - corrgrams, 683
 - cost function, 807–809
 - counting process, 900
 - coupling, 887
 - covariance matrix, 582, 797

- covariance operator, 545
covariate adjustment, 704
coxcomb, 30
CRAN, *see* Comprehensive R Archive Network
crash tests, 459
creation of functional observations, 875
critical path method, 145
Crome, August F.W., 22
cross-sectional, 875, 887
cross-validation, 396, 448, 455, 497, 508, 509
cubic smoothing spline, 877
curse of dimensionality, 540
curve, 874
 - contour, 47
 - cumulative frequency, 37
 - fitting, 24
 - freehand, 38
 - frequency, 37
 - historical, 43
 - isodic, 32
 - isogen, 32
 - loess, 47
 - selection, 892Cusa, Nicolas of, 19
cut point, 244
cylindrical probability, 919
Czekanowski, Sørensen, Dice, 318
- DAG, *see* directed acyclic graph
data
 - analysis, 351
 - arthritis, 604
 - Danish corporal punishment, 611
 - display, 203
 - financial, 655–664
 - fit, 877
 - GIS, 649–652
 - hospital, 591
 - multivariate, 40
 - scaling, 310
 - smoothing, 875
 - sphering, 310
 - transformation, 310
 - UCB admissions, 607
 - visualization, 540, 542, 874, 876
- – high dimensional, 152
data set
 - alpha, 795–797, 804, 806–809
 - Caesarean, 155
 - cars, 157, 166
 - cdcl5, 795–797
 - cdc28, 795–797
 - dentitio, 563
 - detergent, 153
 - elu, 795, 796
 - German election, 563
 - Pollen, 167
 - Tour de France, 169de Fourcroy, Charles, 22
de Witt, Jan, 21
deceleration, 884
decision support, 668–670
decision tree, 245
deductible, 907
degree of interest, 221
Delaunay triangulation, 137
dendrogram, 354, 362, 565, 689, 693
 - cutting level, 374density estimation, 47
density evolution, 915
dentitio data, 563
dependency inversion principle, 732
depth shading, 251
derivative, 877, 878, 882, 891
Descartes, René, 20, 353
design patterns, 730, 732
 - decorator pattern, 741
 - factory method pattern, 743
 - mediator pattern, 751
 - observer pattern, 750
 - state pattern, 746
 - strategy pattern, 748details, 875
detergent data, 153
DFBETAS, 429, 430
dffits, 430
diagnostic, 801, 802
diagram
 - alignment, 31
 - butterfly, 38

- cartogram, 22, 36
- Chernoff faces, 40
- circle, 34
- contour, 30, 32
- coxcomb, 30
- divided circle, 30
- geometric, 18
- Hertzsprung–Russell, 38
- historical, 37
- nomogram, 31
- pie, 34
- rose, 30, 34
- sieve, 41
- statistical, 36
- stereogram, 30
- tree, 40
- differential equation, 884, 897
- dimension
 - high, 540
 - reduction, 540, 546, 551, 682, 698
 - three, 31
- dimension ordering, 184
 - correlation-driven, 185
 - data-driven, 185
 - symmetry-driven, 185
 - user-driven, 186
- dimensionality, 697, 876, 897
- direct clustering, 682
- directed acyclic graph, 245
- discretization, 885, 891
- disparity, 705
- display
 - fourfold, 41
 - trellis, 32
 - two-way table, 39
- dissimilarity, 684
- dissimilarity measure, 316
- distance, 353
 - in \mathbb{R}^p , 360
- distance measure, 565
 - Euclidean, 565
 - Manhattan, 565
- distance transfer function, 232
- distance-based linking, 231
- distortion, 189, 190
- distribution
 - Burr, 904, 908
 - exponential, 903, 907
 - gamma, 904, 908
 - log-normal, 903, 908
 - loss, 902
 - mixture of 2 exponentials, 904, 908
 - Pareto, 904, 908
 - tail, 903
 - Weibull, 904, 908
- divergence metric, 317
- domain-specific linking, 235
- domain-specific visualizations, 220
- dotplot, 442, 697
- doublededecker plot, 71, 607, 627
- drill-down, 225, 250
- du Carla-Boniface, Marcellin, 22
- dualities
 - 2-D, 652
- Dupin, Charles, 26
- dynamic binding, 733
- dynamic interaction, 801
- dynamic sliced inverse regression (DSIR), 824
- dynamics, 875
- e-learning, 758, 777
- eBay, 876, 879, 888, 891
- EDA, *see* exploratory data analysis
- edge frequency polygon, 393
- edge-preserving smoothing, 480
- effect
 - curvature, 468
 - interaction, 468
- effect-ordered data display, 683, 687
- Eigenvalue, 356
 - decomposition, 689
- Eigenvector, 356
- elliptical glyphs, 683
- elliptical seriation, 531, 685, 688, 702
- EM algorithm, 814
- empirical linking, 205, 208
- energy data, 336
- entropy, 254
- equally spaced grid, 882
- Euclid, 353

- “elements”, 353
- Euclidean distance, 317, 565, 684
- Euclidean geometry, 352, 353
- evaluation, 191, 195
 - perception-based, 194
 - performance-based, 194
- event-driven programming, 746
- evolution, 879
- expected shortfall, 903
- exploration, 152, 890
- exploratory data analysis, 164, 648, 649, 682, 705, 712, 832
 - atomic queries, 649–655
 - compound queries, 655–664
 - guidelines, 649–664
 - requirements, 649
 - visual cues, 651
- exploratory graphics, 59
- faceting, 224
- factor analysis, 355, 705
 - factorial plan, 357
 - output interpretation, 357
 - visualization, 371
- FDA, *see* functional data analysis
- feature map, 541–543, 545, 552
- feature space, 542
- Fermat, Pierre, 20
- figure
 - geometric, 22
- filtered kernel, 399
- filtered mode tree, 400
- filtering, 875, 891
 - curves, 893
- final auction price, 876
- financial data, 655–664
- finite mixture model, 580
- finite time horizon, 876
- flexclust, 564
- flexmix, 564
- flipping of intermediate nodes, 689
- flow map, 30, 34
- fluctuation diagram, 258, 628, 843
- focusing, 892
- fonts, 86
- forecast, 876, 894, 895
- forest, 256, 259
- FORTRAN, 40
- frequency polygon, 393
 - bias-optimized, 393
 - edge, 393
- Friendly, Michael, 41
- functional, 874
 - data, 875, 876
 - model, 878
 - object, 874, 877, 885
 - observation, 874
 - summary, 893
- functional data analysis, 874
- fuzzy logic algorithm, 801
- Gabor filter, 815
- Galilei, Galileo, 20
- Galton, Francis, 32, 33, 45–48
- Gannett, Henry, 36
- GAP, *see* generalized association plots
- Gauss, Carl Friedrich, 29
- Gaussian distribution, 581
- Gemma-Frisius, Reginer, 19
- general similarity coefficient, 319
- generalized association plots, 682, 683, 704
- genetic algorithms, 344
- German Bundesbank, 856
- German election data, 563
- GGobi, 296, 301, 303, 305, 312
- Giffen, Robert, 37
- Gini index, 254
- GIS, 779
- GIS data, 649–652
- glyph, 180, 401, 402
 - design, 181, 183, 184, 188
 - evaluation, 191
 - examples, 181, 182
 - limitations, 180
 - strengths, 180
- gold & currencies, 655–664
- Golden Age, 29, 44
- Goodman, Leo, 41
- goodness-of-fit, 804
- gradient fill, 86

-
- Grammar of Graphics, 80
 grand tour, 40, 152, 172, 698
 graph, 104, 122, 245, 882
 – age pyramid, 34
 – bilateral polygon, 36
 – circle, 24
 – hanging rootogram, 39
 – high-resolution, 40
 – line, 21, 24
 – model, 794
 – paper, 24
 – time-series, 18, 24, 43, 45
 graphical
 – Gaussian model, 797
 – interface, 875
 – layout, 96
 – parameter, 85
 – path, 90
 – primitive, 88, 93
 – test statistic, 720
 graphics
 – multivariate, 152
 Graunt, John, 20, 21
 Greenacre, Michael, 351
 Gresham's Law, 77
 grid
 – coordinate, 22
 – hexagonal, 31
 group stimulus space, 333
 Guerry, André-Michel, 26, 29
 GUI, 84, 92, 98
 GUIDE, 451
 Haberman, Shelly, 41
 Halley, Edmund, 22
 Hamman, 318
 Harrison, John, 21
 head injury, 459
 heatmap, 683
 Herschel, William, 25
 hexagonal bin, 402, 404
 Hidalgo stamps, 65
 hierarchical
 – clustering, 689, 696, 705
 – linking, 213
 – model, 715
 – parameter naming, 715
 – sources of variation, 715
 – tree, 123
 – view, 245
 hierarchical cluster analysis, *see* cluster analysis
 high-dimensional data visualization, 152
 high-dimensional visualizations, interacting with, 220
 higher-order kernel, 398
 highlighting, 602, 894
 – linked, 152
 histogram, 63, 390–393, 687, 697, 728, 736, 875,
 890
 – averaged shifted, 393, 409
 – bivariate, 402–404
 – density, 391
 – frequency, 390
 – percentage, 390
 histospline, 393
 HLS, 857, 867, 868
 Homals, 704
 homogeneous Poisson process (HPP), 912
 HTML, 96
 Human–Computer Interaction Laboratory, 890
 hundreds of variables, 658–664
 Huygens, Christiaan, 21
 icon, *see* glyph
 identity linking, 205
 image
 – analysis, 814
 – fusion, 827
 – plot, 402
 – reconstruction, 814
 – segmentation, 814
 impurity measure, 254
 independence, 897
 INDSCAL, 333
 information loss, 876
 inheritance, 733
 insurance, 900
 – automobile, 465

- company
- bankruptcy, 912
- premium, 900
- relative safety loading, 912
- insurer’s surplus, 912
- integrating data
 - Dynamic Signal Maps, 802
 - KnowledgeEditor, 802
 - PathFinder, 802
 - Pathway Assist, 802
 - PubGene Vector PathBlazer, 802
- interactive, 152, 375, 644, 696, 875, 890
 - graphics, 42, 761, 768, 771, 786
 - operation, 728
 - plot, 434–437
 - statistical graphic, 161
 - visualization, 875, 876
- interpolation, 24, 878, 881
- interquartile range, 805
- inverse problem, 814
- isoline, 32
- isomap, 684
- Italian Household Income and Wealth, 377
- Jaccard coefficient, 318, 701
- Jasp, 735
- Jasplot, 735, 754
- Java, 41, 98, 727, 729
 - Abstract Window Toolkit (AWT), 730
 - Java 2D, 730, 740
 - Java Foundation Classes (JFC), 730
 - Swing, 730
- Jevons, William Stanley, 28
- JPEG, 728
- juxtaposition, 211
- k-means, 554, 569
- Karush–Kuhn–Tucker conditions, 555
- KEGG, 700
- Kendall’s tau coefficient, 684
- kernel, 858
 - kernel machine, 540, 541
 - support vector clustering, 558
 - kernel canonical correlation analysis, 542, 543, 551–553
 - kernel principal component analysis, 540, 542–546, 550
 - smooth support vector machine, 543
 - smooth support vector regression, 543
 - support vector clustering, 541, 542, 554–556
- kernel method, 684
- keyword data, 338
- knots, 877
- Kulczyński, 318
- lack-of-fit, 801, 802
- Lallemand, Charles, 31
- Lambert, Johann, 23
- Laplace, Pierre Simon, 29
- large data sets, 341
- larger view, 892
- lassoing, 223
- lattice graphics, 157
- layer, 761, 770, 771
- layout, 188
 - data-driven, 188
 - manager, 96
 - structure-driven, 189
- le Blon, Jacob, 22
- leaf, 244
- legend, 68, 86
- Levasseur, Pierre Émile, 33
- leverage, 429
- life table, 21
- likelihood ratio test, 797
- limited expected value function, 907
- line
 - contour, 22
 - graph
 - 1-D, 21
 - isogon, 22
 - isoline, 22, 32
 - slope as rate, 24
 - timeline, 22
 - type, 85, 86

- linear
 - graph, 123
 - projection, 582
- linear regression
 - multiple, 452
- linkage methods, 565
- linked
 - brushing, 729, 750
 - clustering view, 236
 - data view, 218
 - highlighting, 152, 729, 750
 - map, 235
 - micromap plot, 268
 - network graph, 236
 - plot, 761, 781
 - view, 202, 218, 376, 729, 750
- linking, 40
 - with memory, 233
 - without memory, 233
- Liskov substitution principle, 732
- Lisp-Stat, 41, 42
- lithography, 22
- LM plots, 268
- local linear, 496, 500, 502, 511, 513
- loglinear model, 465
- longitude, 21
- loss distribution, 902
- loss of information, 880
- low-rank approximation, 543
- lowess, 67
- MacSpin, 42
- magnetic resonance images (MRI), 814
- Mahalanobis distance, 317
- majorization, 111, 345
- MANET, 41, 161
- Manhattan distance, 565
- map, 75, 764, 776, 779
 - anamorphic, 31
 - cartography, 22
 - chloropleth, 26
 - contour, 22
 - disease, 27
 - epidemiological, 31
 - flow, 30, 34
 - geological, 25
 - linked to statistical charts, 235
 - shaded, 36
 - thematic, 22, 30
 - topographic, 22
 - weather, 21, 33
- mappings, 181
 - many-to-one, 181
 - one-to-many, 181
 - one-to-one, 181
- Marey, Étienne Jules, 37
- marginal panels, 422
- marketplaces, 876
- Markov chain Monte Carlo (MCMC), 817
- Markov property, 797
- Marshall, Alfred, 43
- mathematical framework, 104
- matrix
 - scatterplot, 40
- matrix maps, 682, 684
- matrix visualization, 629, 682, 683, 690, 704
- Maunder, E.W., 38
- maximum likelihood estimate (MLE), 814
- mean
 - acceleration, 884
 - excess function, 903, 907
 - integrated squared error, 392
 - residual life function, 903
 - squared error, 801
 - velocity, 884
- measurement
 - physical, 20
- measurement error, 22
- medical image, 814
- metabolic networks, 794
- metric, *see* distance
- metric scaling, 316, 319
- metric space, 353
- Michael, 318
- microarray data, 794, 795, 802, 804, 805, 807
- microarrays, 693
- micromaps, 72, 268
- microplot, 441, 442
- Microsoft Excel format, 728
- Minard, Charles Joseph, 28, 30, 60

-
- minimal span loss, 689
 - Minkowski metric, 317
 - missing functional objects, 881
 - missing values, 693, 696, 699, 705
 - mode
 - forest, 400
 - surface, 806
 - tree, 400
 - – filtered, 400
 - model
 - checking, 710, 715
 - generalized linear, 41
 - loglinear, 465
 - misspecification, 878
 - mixed, 41
 - piecewise constant, 451
 - piecewise linear, 451
 - selection, 448
 - tree-structured, 451
 - understanding, 710
 - model–view–controller (MVC), 733, 744
 - model-based cluster analysis, *see* cluster analysis
 - modeling and simulation
 - GenePath, 802
 - Genetic Network Analyzer, 802
 - Mondrian, 41, 161
 - monitor, 165
 - monolithic visualizations, 220
 - monotone smoothing, 888
 - mosaic
 - mondrian display, 634
 - multiple bars, 632
 - reorder rows and columns, 631
 - same-bin-size display, 632
 - mosaicplot, 28, 41, 45, 71, 152, 153, 252, 527, 577, 579, 590, 592, 728, 736, 843
 - conditional probability, 624
 - construction, 619
 - fluctuation diagram, 161
 - multiple barcharts, 161
 - order of variables, 620
 - same bin size, 161
 - variations, 160
- Moseley, Henry, 38
- mountain plot, 254, 263
 - Mountford, 318
 - multidimensional scaling, 40, 41, 110, 705
 - multilevel models, 711
 - multiple correspondence analysis, 117
 - multiple linked views, 214
 - multiple testing, 804
 - multiscale visualization, 400–401
 - multivariate data, 562
 - multivariate Gaussian distribution, 581
 - multivariate graphics, 152
 - mussels
 - horse, 456
 - Nadaraya–Watson, 887
 - neighborhood graph, 571
 - network graphs linked to statistical charts, 236
 - neural networks, 343
 - Neurath, 60
 - NHTSA, 459
 - Nightingale, Florence, 30
 - node, 245
 - noisy realizations, 877
 - nominal data, 704
 - nominal variable, 576
 - nomogram, 31
 - non-metric scaling, 316, 322
 - nonlinear color mapping, 693
 - nonlinear models, 668–670
 - nonparametric density estimation, 805
 - nonparametric regression, 472
 - normalization, 795
 - object, 874
 - object-oriented programming (OOP), 729
 - observed bids, 879
 - occlusion, 180, 191
 - Ochiai, 318
 - odds ratio
 - conditional, 625
 - sufficient set, 626
 - definition, 624
 - high dimensional generalization, 626
 - online, 876
 - online auction, 874, 876

-
- open-closed principle, 732
 OpenGL, 98
 operational taxonomy units, 682
 ordering, 66
 Oresme, Nicole, 19
 Ortelius, Abraham, 19
 outlier, 460, 686, 692, 795, 837, 875
 outlier detection, 794
 overall fitting, 802
 overlay, 210
 overview, 164
- p*-values, 804, 806
 paint mode, 225
 pairs plot, 607
 PAM, *see* partitioning around medoids
 panel plot, 157
 parallax software, 649
 parallel coordinate, 70, 644–678, 890
 - axes permutations, 653–655
 - classifiers, 664–670
 - D’Ocagne, 645
 - decision support, 668–670
 - dualities
 - – 2-D, 652
 - exploration guidelines, 649–664
 - exploratory data analysis, 648–649
 - origins, 645–646
 - Parallax software, 649
 - plot, 40, 152, 164, 575, 683, 697, 699, 728, 736, 752, 837
 - projective plane, 652
 - proximate planes, 648
 - queries
 - – atomic, 649
 - – atomic – EDA, 649–655
 - – compound boolean – EDA, 655–664
 - – compound-boolean, 649
 - – exploration, 649
 - relations → patterns, 644, 648
 - review, 670–678
 - scatterplot matrix, 648
 - visual cues, 651
 parametric bootstrap, 711
 partial functional objects, 881
- particle swarm optimization, 807
 partition
 - recursive, 451
 partitioning around medoids, 569
 partitioning cluster analysis, *see* cluster analysis
 Pascal, Blaise, 20
 passive plot, 203
 Patfield algorithm, 605
 path, 122
 pathway, 700, 703
 pattern, 875, 890
 - fill, 86
 - recognition, 644, 806, 807
 - search, 893
 PCA, *see* principal component analysis
 PDF, 99
 Pearson correlation, 684, 693
 Pearson residuals, 596
 Pearson, Karl, 32
 penalized smoothing spline, 877
 penalized squared error, 877
 penalizing term, 877
 penalty term, 878
 permutation, 683, 686, 693
 permutation matrices, 629
 permutation test, 604
 Perozzo, Luigi, 30
 perspective plot, 404, 408
 Petermann, Augustus, 28
 Petty, William, 20, 21
 phase-plane plots, 875, 884
 Phi, 318
 pictogram, 60
 pie chart, 24, 581
 piecewise additive, 465
 piecewise constant, 451
 piecewise linear, 451
 piechart, 63
 Pima Indian data, 342
 pivot table, 638
 planar graph, 125
 Playfair, William, 23, 24, 43, 45, 58, 832
 plot
 - association, 41

- bivariate, 21
- Fourier function, 40
- log-log, 31, 38
- mosaic, 28, 41, 45
- parallel coordinate, 36
- probability, 42, 908
- star, 34, 40
- stem-leaf, 39
- PNG, 99
- pointwise histograms, 885
- Poisson regression, 472, 481
- political arithmetic, 21
- politician data, 332
- Pollen data set, 167
- polymorphism, 733
- polynomial spline, 877
- positive definite kernel, 541, 552
- positron emission tomography (PET), 814
- posterior predictive checking, 720
- posterior predictive distribution, 712
- posterior uncertainty, 719
- PostScript, 80, 81, 85, 90, 93, 96, 99
- power law, 798
- prediction shading, 251
- predictive distribution, 711, 712
- preprocessing, 878, 881
- presentation, 152
- presentation graphics, 59
- price
 - curve, 876, 879, 881
 - dynamic, 876, 893
 - evolution, 876
- Priestley, Joseph, 22, 42, 43
- PRIM-9, 42
- primary monotone least-squares regression, 323
- principal component analysis, 114, 355, 356, 540, 546, 564, 569, 698, 762, 778
 - correlation circle, 357
 - kernel, 540, 542–546, 550
 - principal components, 357
 - – correlations, 357
 - – orthogonality, 357
 - reduced kernel, 548
 - supplementary
- – points, 359
- – variables, 359
- principal components, 875
- printing
 - three-colour, 22
- probability
 - cylindrical, 919
 - gate, 919
 - plot, 908
 - product, 919
- probability theory, 21
- process
 - aggregate claim, 900
 - claim arrival, 900
 - counting, 900
 - density evolution, 915
 - Poisson
 - – homogeneous (HPP), 912
 - renewal, 912
 - risk, 900
 - – trajectory, 912
 - stationary, 916
- Procrustes analysis, 316, 330
- product kernel, 404, 406–408
- product probability, 919
- profile plots, 888
- profiles, 164
- projection, 297, 301
 - axonometric, 30
- projection pursuit, 172, 543
- projection pursuit index, 304
- projection pursuit, central mass index, 307
- projection pursuit, holes index, 305
- projection pursuit, LDA index, 307
- projection pursuit, optimization, 305
- projection pursuit, PCA index, 307
- propagate, 878
- propagation condition, 477, 478, 482, 485, 487
- Propagation-Separation (PS), 485
- Property Claim Services, 901
- proximate planes, 648
- proximity data, 316
- proximity graph, 136
- proximity matrices, 683, 684
- Psychometrika, 316

-
- Ptolemy, Claudius, 18, 21
 Python, 98
- Quetelet, Adolphe, 27, 29, 33
 quantile, 916
 quantile line, 916
 query
 - atomic, 649
 - – explained, 649–653
 - atomic – EDA, 649–655
 - compound boolean – EDA, 664
 - compound-boolean, 649
 - compound-boolean – EDA, 655
 - exploration, 649
- R, 41, 157, 417, 418, 427, 564, 591, 777, 784, 786
 R^2 , 804, 806
 R2WinBUGS, 718, 722
 radial basis function, 546, 550, 877
 rail travel data, 320
 random forest, 257
 random graph, 123
 random subset, 543
 Rao, 318
 rating, 888
 reciprocal averaging, 340
 recover, 877
 rectangle selection, 223
 recursive partitioning, 244
 reduced kernel, 543, 548, 552, 553
 reference band, 499, 500, 502, 506
 reference distribution, 710, 711
 reference region, 505
 region competition, 815
 regression, 32, 47, 448, 472
 - diagnostic, 429, 431, 435
 - model, 804
 - Poisson, 453, 465
 - sliced inverse, 456
 - stepwise, 453
 - tree, 244, 449, 451
- relationship, 875, 886
 relative risk, 442–444
 relative safety loading, 912
 relativity of a statistical graph, 683, 687, 688
- renewal process, 912
 reorderable matrix, 682
 reordering, 682
 replacement, 210
 replicated data, 715, 721
 replication, 210
 replication distribution, 711
 reproducing kernel, 541
 reproducing kernel Hilbert space, 540, 541
 residual, 448
 residual-based shading, 602
 resolution of a statistical graph, 686
 response model, 155
 response surface, 801–805
 RGB
 - cube, 685
- Rheticus, Georg, 19
 Rice Virtual Lab in Statistics, 393
 ridge regression
 - smooth support vector regression, 543
- risk process, 900
 - α -stable Lévy motion approximation, 919
 - diffusion approximation, 916
 - trajectory, 912
- Riverplot, 893
 Robinson matrix, 688
 ROC curve, 546
 - area under, 546
- Rogers, Tanimoto, 318
 rooted tree, 125
 rootogram, 582, 584
 roughness functional, 392
 roughness penalty, 877
 Royal Statistical Society, 28, 43
 rubber-band selection, 223
 rug fringe, 437
 rug plot, 875, 888
 ruin
 - probability, 914
 - theory, 912
- Ruin Probabilities toolbox, 902
 ruin probability, 914
 ruin theory, 912
 Russell, 318

-
- S-Plus, 417, 418, 427
 - Saccharomyces cerevisiae yeast, 696
 - Sammon map, 321
 - SAS, 41
 - scagnostics, 143
 - scalability, 194, 195
 - scalable vector graphics, 86, 90, 93, 96, 728, 762, 765
 - scale, 64
 - common, 162
 - scaling, 169
 - multidimensional, 40, 41
 - scattergram, 890
 - scatterplot, 401–402, 433, 687, 697, 728, 736, 738, 875, 886
 - brushing, 200, 224
 - matrix, 40, 70, 166, 299, 311, 418–422, 426, 430, 437, 697, 698, 728, 729, 736, 752
 - Scheiner, Christopher, 20
 - Schwabe, Hermann, 33
 - scree plot, 326
 - SDE-Solver, 902
 - SearchBox, 893
 - searching, 893
 - sectional display, 692
 - sectioned scatterplot, 249, 259, 262
 - sediment display, 691
 - selecting, 894
 - selection bias, 449
 - selection modes, 226
 - self-organizing map, 577, 690
 - semitransparency, 80, 86
 - Senefelder, Aloys, 22
 - separation, 892
 - sequences, 145
 - seriation, 682
 - shadow value, 567, 572
 - Shakespeare keywords, 325
 - shape, 874
 - Shepard plot, 323
 - shingles, 158
 - shortest path, 689
 - shortest spanning path, 683
 - sieve diagram, 41
 - sieve plot, 590, 595
 - significance probability, 454
 - silhouette plot, 573
 - silhouette value, 573
 - similar set, 895
 - similarity, 684
 - similarity coefficient, 318
 - similarity metric, 540
 - simple effects, 434, 437, 438
 - simple matching coefficient, 318, 701
 - Simpson, 318
 - simulated annealing, 344
 - SiZer plot, 400
 - sliced inverse regression (SIR), 815
 - small multiple, 20, 35, 433
 - Smalltalk, 733
 - Smith, William, 25
 - smoother, 877
 - smoothing, 37, 47, 877
 - smoothing parameter, 496, 497, 501–503, 507, 508, 514, 877, 881
 - smoothing parameters, 878
 - smoothness, 877
 - snake energy model, 807
 - snake–balloon model, 815
 - Sneath, Peter H.A., 318
 - Snow, John, 27, 30
 - Soergel metric, 317
 - software, 76
 - Ruin Probabilities toolbox, 902
 - SDE-Solver, 902
 - XploRe
 - – Insurance Library, 902
 - Sokal, Robert, 318
 - Solka, Jeff L., 366
 - SOM, *see* self-organizing map
 - sorting, 66, 169
 - Sorting auctions, 893
 - spanning tree, 132
 - spatial data, 764
 - Spearman’s rank correlation, 684
 - speckle noise, 815
 - spectrum, 123
 - spineplot, 253, 593
 - spineplot of leaves, 252, 263
 - spline order, 877

-
- split, 244
 - SPLOM, *see* scatterplot matrix
 - SPOL, *see* spineplot of leaves
 - spreadplot, 40
 - spring model, 341
 - star plot, 34, 40
 - static, 875
 - static graphs, 890
 - statistical atlas, 33
 - statistics
 - demographic, 20
 - moral, 26
 - stem-leaf plot, 39
 - stereogram, 30
 - Steven and Weber laws, 373
 - stratigraphic geology, 25
 - streaming, 875
 - stress, 323, 705
 - strip labels, 158
 - structural adaptation, 475
 - Sturges' rule, 392
 - subjects space, 333
 - sufficient matrix visualization, 690
 - summary, 882
 - summary statistics, 875, 894
 - supervised classification, 310
 - supervised classification, LDA, 311
 - supervised classification, QDA, 312
 - supervised classification, trees, 311
 - support vector, 556, 557
 - bounded, 557
 - support vector machine, 854
 - smooth, 543
 - support vector regression
 - smooth, 543
 - surface estimation, 503–509, 513, 526
 - SVG, *see* scalable vector graphics
 - SVM, *see* support vector machine
 - symbol
 - sunflower, 47
 - table
 - graphical, 24
 - life, 21
 - semi-graphic, 47
 - tableau-graphique, 28
 - tabular, 890
 - tabular views, 891
 - tandem analysis, 365
 - Tcl/Tk, 41
 - temporal information, 875
 - terminal node, 244, 251, 252, 689
 - test statistic, 720
 - thematic cartography, 22
 - theorem
 - Ekart and Young, 355
 - three-dimensional scatterplot, 886
 - threshold, 800, 804
 - time boxes, 890
 - time series, 24, 74, 799, 875, 887
 - multiple, 30
 - operation, 891
 - time-lagged, 799–801
 - TimeBox, 893, 894
 - timeline, 22, 42
 - TimeSearcher, 890
 - Titanic, 66
 - tour
 - axes, 298
 - basis, 302
 - finding structure, 300, 307
 - frame, 302
 - geodesic, 301
 - grand, 40, 296, 303
 - guided, 296, 303
 - interpolation, 302
 - manual, 296, 307
 - path, 303, 305
 - target basis, 302
 - target plane, 302, 303
 - within-plane, 301, 302
 - Tour de France, 65, 70
 - data set, 169
 - trace plot, 260, 263
 - transformations, 684
 - traveling salesman, 683, 689
 - tree seriation, 689
 - treemap, 135, 237, 251, 263, 635
 - Trellis
 - display, 72, 152, 157, 889

- graphic, 288
- layout, 610
- paradigm, 417–419, 422, 434
- plot, 83, 94, 99, 636
- trends, 875
- triangulation, 19
- Tufte, Edward, 20, 61, 68
- Tukey, John Wilder, 39, 351, 832
- two-way table, 39
- typesetting, 96

- ultrametric tree, 124
- ultrasound image, 814
- unaggregated view, 222
- unidimensional scaling, 331
- unified modeling language (UML), 734
- UNIX, 40
- unrepresentative, 881

- van Langren, Michael F., 20
- variability band, 498, 499, 501, 504
- variable bandwidth, 399
- variable location, 398
- variable selection, 257, 455, 860
- vertex, 122, 245
- view box, 892
- viewport, 93
- virus data, 321
- vision model, 814
- ViSta, 42
- visual thinking, 22
- visualization, 644, 874, 875, 878, 882, 891
 - Archimedes, 645
 - BioMiner, 802
 - displays for exploration, 646
 - early successes, 644
 - Geometry, 645

- Graphviz, 800, 802
- interaction effects, 624
- multidimensional, 644
- Ospray, 802
- system, 108
- the case for, 646–648
- VITAMIN-S, VIsualization and daTA MIN-ing System, 372
- von Mayr, Georg, 33
- Voronoi partition, 570
- Voronoi tessellation, 137

- Walker, Francis A., 34
- wall quadrant, 19
- Wave-Hedges, 317
- weather map, 21, 33
- Wegman, Edward J., 366
- weighted Euclidean distance, 317
- weighted graph, 122
- weighted least square estimate (WLSE), 817
- weighted plots, 840, 844
- widgets, 893
- wireframe plot, 404
- WMF, 91
- World Wide Web, 99

- X3D, 771
- XGobi, 42, 312
- XML, 759
- XploRe
 - Insurance Library, 902
- xysplom, 419, 426, 427

- yeast two-hybrid screen, 794
- Yule, 318

- Zeuner, Gustav, 30
- zooming, 875, 892