Contents

Preface xiii

I Getting Started 1

1 Introduction 3
- 1.1 Data visualization and categorical data: Overview . . . 3
- 1.2 What is categorical data? . . . 4
  - 1.2.1 Case form vs. frequency form . . . 5
  - 1.2.2 Frequency data vs. count data . . . 6
  - 1.2.3 Univariate, bivariate, and multivariate data . . . 7
  - 1.2.4 Explanatory vs. response variables . . . 7
- 1.3 Strategies for categorical data analysis . . . 8
  - 1.3.1 Hypothesis testing approaches . . . 8
  - 1.3.2 Model building approaches . . . 10
- 1.4 Graphical methods for categorical data . . . 13
  - 1.4.1 Goals and design principles for visual data display . . . 13
  - 1.4.2 Categorical data require different graphical methods . . . 16
  - 1.4.3 Effect ordering and rendering for data display . . . 18
  - 1.4.4 Interactive and dynamic graphics . . . 20
  - 1.4.5 Visualization = Graphing + Fitting + Graphing . . . 22
  - 1.4.6 Data plots, model plots, and data+model plots . . . 24
  - 1.4.7 The 80–20 rule . . . 26
- 1.5 Chapter summary . . . 28
- 1.6 Lab exercises . . . 28

2 Working with Categorical Data 31
- 2.1 Working with R data: vectors, matrices, arrays, and data frames . . . 32
  - 2.1.1 Vectors . . . 32
  - 2.1.2 Matrices . . . 33
  - 2.1.3 Arrays . . . 35
  - 2.1.4 Data frames . . . 37
- 2.2 Forms of categorical data: case form, frequency form, and table form . . . 39
  - 2.2.1 Case form . . . 39

vii repaginate

2.2.2 Frequency form . . . 40
2.2.3 Table form . . . 41
2.3 Ordered factors and reordered tables . . . 43
2.4 Generating tables: table and xtabs . . . 44
2.4.1 table() . . . 44
2.4.2 xtabs() . . . 46
2.5 Printing tables: structable and ftable . . . 47
2.5.1 Text output . . . 47
2.6 Subsetting data . . . 48
2.6.1 Subsetting tables . . . 48
2.6.2 Subsetting structables . . . 49
2.6.3 Subsetting data frames . . . 50
2.7 Collapsing tables . . . 50
2.7.1 Collapsing over table factors . . . 50
2.7.2 Collapsing table levels . . . 52
2.8 Converting among frequency tables and data frames . . . 53
2.8.1 Table form to frequency form . . . 53
2.8.2 Case form to table form . . . 55
2.8.3 Table form to case form . . . 55
2.8.4 Publishing tables to LaTeX or HTML . . . 56
2.9 A complex example: TV viewing data* . . . 58
2.9.1 Creating data frames and arrays . . . 58
2.9.2 Subsetting and collapsing . . . 59
2.10 Lab exercises . . . 60

3 Fitting and Graphing Discrete Distributions 65
3.1 Introduction to discrete distributions . . . 66
3.1.1 Binomial data . . . 66
3.1.2 Poisson data . . . 69
3.1.3 Type-token distributions . . . 72
3.2 Characteristics of discrete distributions . . . 73
3.2.1 The binomial distribution . . . 74
3.2.2 The Poisson distribution . . . 76
3.2.3 The negative binomial distribution . . . 82
3.2.4 The geometric distribution . . . 84
3.2.5 The logarithmic series distribution . . . 85
3.2.6 Power series family . . . 86
3.3 Fitting discrete distributions . . . 87
3.3.1 R tools for discrete distributions . . . 88
3.3.2 Plots of observed and fitted frequencies . . . 92
3.4 Diagnosing discrete distributions: Ord plots . . . 95
3.5 Poissonness plots and generalized distribution plots . . . 99
3.5.1 Features of the Poissonness plot . . . 99
3.5.2 Plot construction . . . 99
3.5.3 The distplot function . . . 101
3.5.4 Plots for other distributions . . . 101
3.6 Fitting discrete distributions as generalized linear models* . . . 104
3.6.1 Covariates, overdispersion, and excess zeros . . . 106
3.7 Chapter summary . . . 108

3.8 Lab exercises 109

II Exploratory and Hypothesis-Testing Methods 113

4 Two-Way Contingency Tables 115
4.1 Introduction 115
4.2 Tests of association for two-way tables 119
4.2.1 Notation and terminology 119
4.2.2 2 by 2 tables: Odds and odds ratios 121
4.2.3 Larger tables: Overall analysis 124
4.2.4 Tests for ordinal variables 125
4.2.5 Sample CMH profiles 126
4.3 Stratified analysis 128
4.3.1 Computing strata-wise statistics 128
4.3.2 Assessing homogeneity of association 129
4.4 Fourfold display for 2 x 2 tables 130
4.4.1 Confidence rings for odds ratio 133
4.4.2 Stratified analysis for $2 \times 2 \times k$ tables 133
4.5 Sieve diagrams 138
4.5.1 Two-way tables 138
4.5.2 Larger tables: The strucplot framework 141
4.6 Association plots 145
4.7 Observer agreement 146
4.7.1 Measuring agreement 148
4.7.2 Observer agreement chart 150
4.7.3 Observer bias in agreement 152
4.8 Trilinear plots 153
4.9 Chapter summary 156
4.10 Lab exercises 157

5 Mosaic Displays for n-Way Tables 161
5.1 Introduction 161
5.2 Two-way tables 162
5.2.1 Shading levels 166
5.2.2 Interpretation and reordering 166
5.3 The strucplot framework 167
5.3.1 Components overview 167
5.3.2 Shading schemes 169
5.4 Three-way and larger tables 176
5.4.1 A primer on loglinear models 177
5.4.2 Fitting models 179
5.5 Model and plot collections 183
5.5.1 Sequential plots and models 184
5.5.2 Causal models 186
5.5.3 Partial association 188
5.6 Mosaic matrices for categorical data 197
5.6.1 Mosaic matrices for pairwise associations 197
5.6.2 Generalized mosaic matrices and pairs plots 201
5.7 3D mosaics 203
5.8 Visualizing the structure of loglinear models 205
5.8.1 Mutual independence 206

5.8.2 Joint independence 208
5.9 Related visualization methods 209
5.9.1 Doubledecker plots 209
5.9.2 Generalized odds ratios* 212
5.10 Chapter summary 215
5.11 Lab exercises 216

6 Correspondence Analysis **221**
6.1 Introduction 221
6.2 Simple correspondence analysis 222
6.2.1 Notation and terminology 222
6.2.2 Geometric and statistical properties 224
6.2.3 R software for correspondence analysis 224
6.2.4 Corespondence analysis and mosaic displays 231
6.3 Multi-way tables: Stacking and other tricks 232
6.3.1 Interactive coding in R 233
6.3.2 Marginal tables and supplementary variables 238
6.4 Multiple correspondence analysis 240
6.4.1 Bivariate MCA 240
6.4.2 The Burt matrix 243
6.4.3 Multivariate MCA 243
6.5 Biplots for contingency tables 248
6.5.1 CA bilinear biplots 248
6.5.2 Biadditive biplots 252
6.6 Chapter summary 254
6.7 Lab exercises 254

III Model-building Methods **259**

7 Logistic Regression Models **261**
7.1 Introduction 261
7.2 The logistic regression model 263
7.2.1 Fitting a logistic regression model 265
7.2.2 Model tests for simple logistic regression 267
7.2.3 Plotting a binary response 268
7.2.4 Grouped binomial data 270
7.3 Multiple logistic regression models 272
7.3.1 Conditional plots 275
7.3.2 Full-model plots 276
7.3.3 Effect plots 278
7.4 Case studies 281
7.4.1 Simple models: Group comparisons and effect plots 282
7.4.2 More complex models: Model selection and visualization 294
7.5 Influence and diagnostic plots 303
7.5.1 Residuals and leverage 303
7.5.2 Influence diagnostics 304
7.5.3 Other diagnostic plots* 311
7.6 Chapter summary 319
7.7 Lab exercises 320

8 Models for Polytomous Responses **323**
8.1 Ordinal response . . . 324
8.1.1 Latent variable interpretation . . . 325
8.1.2 Fitting the proportional odds model . . . 326
8.1.3 Testing the proportional odds assumption . . . 327
8.1.4 Graphical assessment of proportional odds . . . 329
8.1.5 Visualizing results for the proportional odds model . . . 331
8.2 Nested dichotomies . . . 335
8.3 Generalized logit model . . . 341
8.4 Chapter summary . . . 346
8.5 Lab exercises . . . 346

9 Loglinear and Logit Models for Contingency Tables **349**
9.1 Introduction . . . 349
9.2 Loglinear models for frequencies . . . 350
9.2.1 Loglinear models as ANOVA models for frequencies . . . 350
9.2.2 Loglinear models for three-way tables . . . 351
9.2.3 Loglinear models as GLMs for frequencies . . . 352
9.3 Fitting and testing loglinear models . . . 353
9.3.1 Model fitting functions . . . 353
9.3.2 Goodness-of-fit tests . . . 354
9.3.3 Residuals for loglinear models . . . 356
9.3.4 Using loglm() . . . 357
9.3.5 Using glm() . . . 359
9.4 Equivalent logit models . . . 363
9.5 Zero frequencies . . . 367
9.6 Chapter summary . . . 371
9.7 Lab exercises . . . 372

10 Extending Loglinear Models **373**
10.1 Models for ordinal variables . . . 374
10.1.1 Loglinear models for ordinal variables . . . 374
10.1.2 Visualizing model structure . . . 379
10.1.3 Log-multiplicative (RC) models . . . 380
10.2 Square tables . . . 387
10.2.1 Quasi-independence, symmetry, quasi-symmetry, and topological models . . . 387
10.2.2 Ordinal square tables . . . 394
10.3 Three-way and higher-dimensional tables . . . 398
10.4 Multivariate responses* . . . 401
10.4.1 Bivariate, binary response models . . . 402
10.4.2 More complex models . . . 412
10.5 Chapter summary . . . 423
10.6 Lab exercises . . . 424

11 Generalized Linear Models for Count Data **427**
11.1 Components of Generalized Linear Models . . . 428
11.1.1 Variance functions . . . 429
11.1.2 Hypothesis tests for coefficients . . . 430
11.1.3 Goodness-of-fit tests . . . 431
11.1.4 Comparing non-nested models . . . 432

11.2 GLMs for count data 433
11.3 Models for overdispersed count data 442
11.3.1 The quasi-Poisson model 443
11.3.2 The negative-binomial model 444
11.3.3 Visualizing the mean–variance relation 444
11.3.4 Testing overdispersion 447
11.3.5 Visualizing goodness-of-fit 448
11.4 Models for excess zero counts 449
11.4.1 Zero-inflated models 450
11.4.2 Hurdle models 452
11.4.3 Visualizing zero counts 452
11.5 Case studies 454
11.5.1 Cod parasites 454
11.5.2 Demand for medical care by the elderly 466
11.6 Diagnostic plots for model checking 478
11.6.1 Diagnostic measures and residuals for GLMs 478
11.6.2 Quantile–quantile and half-normal plots 483
11.7 Multivariate response GLM models* 487
11.7.1 Analyzing correlations: HE plots 489
11.7.2 Analyzing associations: Odds ratios and fourfold plots 490
11.8 Chapter summary 497
11.9 Lab exercises 499

References **503**

Author Index **523**

Example Index **527**

Subject Index **529**

Preface

> The greatest possibilities of visual display lie in vividness and inescapability of the intended message. A visual display can stop your mental flow in its tracks and make you think. A visual display can force you to notice what you never expected to see.
>
> John W. Tukey Tukey (1990)

Data analysis and graphics

This book stems from the conviction that data analysis and statistical graphics should go hand-in-hand in the process of understanding and communicating statistical data. Statistical summaries compress a data set into a few numbers, the result of an hypothesis test, or coefficients in a fitted statistical model, while graphical methods help us to explore patterns and trends, see the unexpected, identify problems in an analysis, and communicate results and conclusions in principled and effective ways.

This interplay between analysis and visualization has long been a part of statistical practice for *quantitative data*. Indeed, the origin of correlation, regression, and linear models (regression, ANOVA) can arguably be traced to Francis Galton's (1886) visual insight from a scatterplot of heights of children and their parents on which he overlaid smoothed contour curves of roughly equal bivariate frequencies and lines for the means of $Y \mid X$ and $X \mid Y$ (described in Friendly and Denis (2005), Friendly et al. (2013)).

The analysis of discrete data is a much more recent arrival, beginning in the 1960s and giving rise to a few seminal books in the 1970s (Bishop et al., 1975, Haberman, 1974, Goodman, 1978, Fienberg, 1980). Agresti (2013, Chapter 17) presents a brief historical overview of the development of these methods from their early roots around the beginning of the 20^{th} century.

Yet curiously, associated graphical methods for categorical data were much slower to develop. This began to change as it was recognized that counts, frequencies, and discrete variables required different schemes for mapping numbers into useful visual representations (Friendly, 1995, 1997), some quite novel. The special nature of discrete variables and frequency data vis-a-vis statistical graphics is now more widely accepted, and many of these new graphical methods (e.g., mosaic displays, fourfold plots, diagnostic plots for generalized linear models) have become, if not mainstream, then at least more widely used in research, teaching and communication.

Much of what had been developed through the 1990s for graphical methods for discrete data was

described in the book *Visualizing Categorical Data* (Friendly, 2000) and was implemented in SAS® software. Since that time, there has been considerable growth in both statistical methods for the analysis of categorical data (e.g., generalized linear models, zero-inflation models, mixed models for hierarchical and longitudinal data with discrete outcomes), along with some new graphical methods for visualizing and interpreting the results (3D mosaic plots, effect plots, diagnostic plots, etc.). The bulk of these developments have been implemented in R, and the time is right for an in-depth treatment of modern graphical methods for the analysis of categorical data, to which you are now invited.

Goals

This book aims to provide an applied, practically oriented treatment of modern methods for the analysis of categorical data—discrete response data and frequency data—with a main focus on graphical methods for exploring data, spotting unusual features, visualizing fitted models, and presenting or explaining results.

We describe the necessary statistical theory (sometimes in abbreviated form) and illustrate the practical application of these techniques to a large number of substantive problems: how to organize the data, conduct an analysis, produce informative graphs, and understand what they have to say about the data at hand.

Overview and organization of this book

This book is divided into three parts. Part I, Chapters 1–3, contains introductory material on graphical methods for discrete data, basic R skills needed for the book, and methods for fitting and visualizing one-way discrete distributions.

Part II, Chapters 4–6, is concerned largely with simple, traditional non-parametric tests and exploratory methods for visualizing patterns of association in two-way and larger frequency tables. Some of the discussion here introduces ideas and notation for loglinear models that are treated more generally in Part III.

Part III, Chapters 7–11, discusses model-based methods for the analysis of discrete data. These are all examples of generalized linear models. However, for our purposes, it has proved more convenient to develop this topic from the specific cases (logistic regression, loglinear models) to the general rather than the reverse.

Chapter 1: ***Introduction.*** Categorical data require different statistical and graphical methods than commonly used for quantitative data. This chapter outlines the basic orientation of the book toward visualization methods and some key distinctions regarding the analysis and visualization of categorical data.

Chapter 2: ***Working with Categorical Data.*** Categorical data can be represented in various forms: case form, frequency form, and table form. This chapter describes and illustrates the skills and techniques in R needed to input, create, and manipulate R data objects to represent categorical data, and convert these from one form to another for the purposes of statistical analysis and visualization, which are the subject of the remainder of the book.

Chapter 3: ***Fitting and Graphing Discrete Distributions.*** Understanding and visualizing discrete data distributions provides a building block for model-based methods discussed in Part III. This chapter introduces the well-known discrete distributions—the binomial, Poisson, negative-binomial, and others—in the simplest case of a one-way frequency table.

Chapter 4: *Two-Way Contingency Tables.* The analysis of two-way frequency tables concerns the association between two variables. A variety of specialized graphical displays help to visualize the pattern of association, using area of some region to represent the frequency in a cell. Some of these methods are focused on visualizing an odds ratio (for 2×2 tables), or the general pattern of association, or the agreement between row and column categories in square tables.

Chapter 5: *Mosaic Displays for n-Way Tables.* This chapter introduces mosaic displays, designed to help to visualize the pattern of associations among variables in two-way and larger tables. Extensions of this technique can reveal partial associations and marginal associations, and shed light on the structure of loglinear models themselves.

Chapter 6: *Correspondence Analysis.* Correspondence analysis provides visualizations of associations in a two-way contingency table in a small number of dimensions. Multiple correspondence analysis extends this technique to n-way tables. Other graphical methods, including mosaic matrices and biplots, provide complementary views of loglinear models for two-way and n-way contingency tables.

Chapter 7: *Logistic Regression Models.* This chapter introduces the modeling framework for categorical data in the simple situation where we have a categorical response variable, often binary, and one or more explanatory variables. A fitted model provides both statistical inference and prediction, accompanied by measures of uncertainty. Data visualization methods for discrete response data must often rely on smoothing techniques, including both direct, non-parametric smoothing and the implicit smoothing that results from a fitted parametric model. Diagnostic plots help us to detect influential observations that may distort our results.

Chapter 8: *Models for Polytomous Responses.* This chapter generalizes logistic regression models for a binary response to handle a multi-category (polytomous) response. Different models are available depending on whether the response categories are nominal or ordinal. Visualization methods for such models are mostly straightforward extensions of those used for binary responses presented in Chapter 7.

Chapter 9: *Loglinear and Logit Models for Contingency Tables.* This chapter extends the model-building approach to loglinear and logit models. These comprise another special case of generalized linear models designed for contingency tables of frequencies. They are most easily interpreted through visualizations, including mosaic displays and effect plots of associated logit models.

Chapter 10: *Extending Loglinear Models.* Loglinear models have special forms to represent additional structure in the variables in contingency tables. Models for ordinal factors allow a more parsimonious description of associations. Models for square tables allow a wide range of specific models for the relationship between variables with the same categories. Another extended class of models arise when there are two or more response variables.

Chapter 11: *Generalized Linear Models.* Generalized linear models extend the familiar linear models of regression and ANOVA to include counted data, frequencies, and other data for which the assumptions of independent, normal errors are not reasonable. We rely on the analogies between ordinary and generalized linear models (GLMs) to develop visualization methods to explore the data, display the fitted relationships, and check model assumptions. The main focus of this chapter is on models for count data.

Audience

This book has been written to appeal to two broad audiences wishing to learn to apply methods for discrete data analysis:

- Advanced undergraduate and graduate students in the social and health sciences, epidemiology, economics, business and (bio)statistics
- Substantive researchers, methodologists and consultants in various disciplines wanting to be able to use these methods with their own data and analyses.

It assumes the reader has a basic understanding of statistical concepts at least at an intermediate undergraduate level including regression and analysis of variance (for example, at the level of Neter et al. (1990) or Mendenhall and Sincich (2003)). It is less technically demanding than other modern texts covering categorical data analysis at a graduate level, such as Agresti (2013), *Categorical Data Analysis*, Powers and Xie (2008), *Statistical Methods for Categorical Data Analysis*, and Christensen (1997), *Log-Linear Models and Logistic Regression*. Nevertheless, there are some topics that are a bit more advanced or technical, and these are marked as * or ** sections.

As well, there are a number of mathematical or statistical topics that we use in passing, but do not describe in these pages (some matrix notation, basic probability theory, maximum likelihood estimation, etc.). Most of these are described in Fox (2015), which is available online and serves well as a supplement to this book.

In addition, it is not possible to include *all* details of using R effectively for data analysis. It is assumed that the reader has at least basic knowledge of the R language and environment, including interacting with the R console (RGui for Windows, R.app for Mac OS X) or other graphical user interface (e.g., RStudio), using R functions in packages, getting help for these from R, etc. One introductory chapter (Chapter 2) is devoted to covering the particular topics most important to categorical data analysis, beyond such basic skills needed in the book.

Textbook use

This book is most directly suitable for a one-semester applied advanced undergraduate or graduate course on categorical data analysis with a strong emphasis on the use of graphical methods to understand and explain data and results of analysis. A detailed outline of such a course, together with lecture notes and assignments, is available at the first author's web page, `http://euclid.psych.yorku.ca/www/psy6136/`, using this book as the main text. This course also uses Agresti (2007), *An Introduction to Categorical Data Analysis* for additional readings.

For instructors teaching a more traditional course using one of the books mentioned above as the main text, this book would be a welcome supplement, because almost all other texts treat graphical methods only perfunctorily, if at all. A few of these contain a brief appendix mentioning software, or have a related web site with some data sets and software examples. Moreover, none actually describe how to do these analyses and graphics with R.

Features

- Provides an accessible introduction to the major methods of categorical data analysis for data exploration, statistical testing, and statistical models.
- The emphasis throughout is on computing, visualizing, understanding, and communicating the results of these analyses.
- As opposed to more theoretical books, the goal here is to help the reader to translate theory into practical application, by providing skills and software tools for carrying out these methods.
- Includes many examples using real data, often treated from several perspectives.
- The book is supported directly by R packages vcd (Meyer et al., 2015) and vcdExtra (Friendly, 2015), along with numerous other R packages.
- All materials (data sets, R code) will be available online on the web site for the book.

- Each chapter contains a collection of lab exercises, which work through applications of some of the methods presented in that chapter. This makes the book more suitable for both self-study and classroom use.

Acknowledgments

We are grateful to many colleagues, friends, students and Internet acquaintences who have contributed to to this book, directly or indirectly.

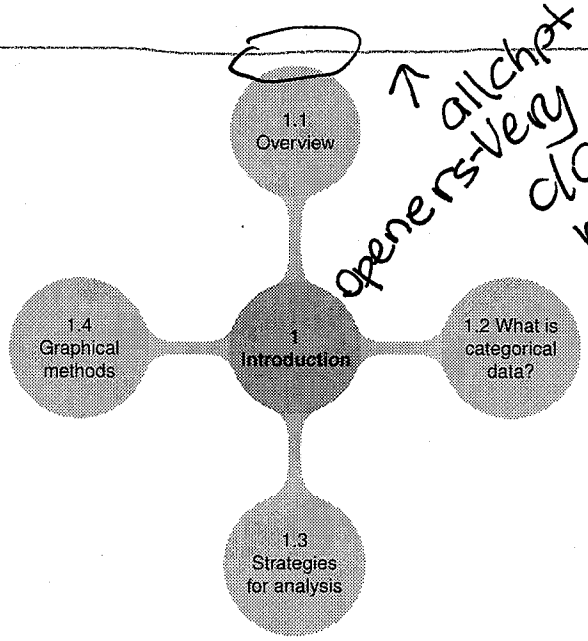
We thank those who read and commented on various drafts of the book or chapters. In particular, John Fox, Michael Greenacre, and several anonymous reviewers gave insightful comments on the organization of the book and made many helpful suggestions. Matthew Sigal used his wizardly skills to turn sketches of conceptual diagrams into final figures. Phil Chalmers contributed greatly with technical and stylistic editing of a number of chapters.

At a technical level, we were aided by the cooperation of a number of R package authors, who helped to enhance the graphic displays: Achim Zeilleis who served as a guiding hand in the development of the vcd and vcdExtra packages; John Fox and Sandy Weisberg for enhancements to the car (Fox and Weisberg, 2015a) and effects (Fox et al., 2015) packages; Milan Bouchet-Valat for incorporating suggestions dealing with plotting `rc()` solutions into the logmult (Bouchet-Valat, 2015) package; Michael Greenacre and Oleg Nenadic for help to enhance plotting in the ca (Greenacre and Nenadic, 2014) package; Heather Turner for advice and help with plotting models fit using the gnm (Turner and Firth, 2014) package; Jay Emerson for improvements to the gpairs (Emerson and Green, 2014) package.

There were also many contributors from the R-Help email list (`r-help@r-project.org`), too many to name them all. Special thanks for generous assistance go to: David Carlson, William Dunlap, Bert Gunter, Jim Lemon, Duncan Murdoch, Denis Murphy, Jeff Newmiller, Richard Heiberger, Thierry Onkelinx, Marc Schwartz, David Winsemius, and Ista Zahn.

The book was written using the knitr (Xie, 2015) package, allowing a relatively seamless integration of LaTeX text, R code, and R output and graphs, so that any changes in the code were automatically incorporated in the book. Thanks are due to Yihui Xie and all the contributors to the `knitr` project for making this possible. We are also grateful to Phil Chalmers and Derek Harnanansingh for assistance in using GitHub to manage our collaboration. Pere Millán-Martínez and Marcus Fontaine helped considerably with LaTeX formatting issues.

The first author's work on this project was supported by grants from the National Science and Engineering Research Council of Canada (Grant 8150) and a sabbatical leave from York University in 2013–14, during which most of this book was written.



1

Introduction

Categorical data consist of variables whose values comprise a set of discrete categories. Such data require different statistical and graphical methods than commonly used for quantitative data. The focus of this book is on visualization techniques and graphical methods designed to reveal patterns of relationships among categorical variables. This chapter outlines the basic orientation of the book and some key distinctions regarding the analysis and visualization of categorical data.

1.1 Data visualization and categorical data: Overview

> Graphs carry the message home. A universal language, graphs convey information directly to the mind. Without complexity there is imaged to the eye a magnitude to be remembered. Words have wings, but graphs interpret. Graphs are pure quantity, stripped of verbal sham, reduced to dimension, vivid, unescapable.
>
> Henry D. Hubbard, in Foreword to Brinton (1939), *Graphic Presentation*

"Data visualization" can mean many things, from popular press infographics, to maps of voter turnout or party choice. Here we use this term in the narrower context of statistical analysis. As such, we refer to an approach to data analysis that focuses on *insightful* graphical display in the service of both *understanding* our data and *communicating* our results to others.

We may display the raw data, some summary statistics, or some indicators of the quality or adequacy of a fitted model. The word "insightful" suggests that the goal is (hopefully) to reveal some aspects of the data that might not be perceived, appreciated, or absorbed by other means. As

contrast, Figure 1.4 shows two other model-based smoothers that relax the assumption of the linear logistic regression model. The left panel shows the result of fitting a semi-parametric model with a natural cubic spline with one more degree of freedom than the linear logistic model. The right panel shows the fitted curve for a non-parametric, locally weighted scatterplot smoothing (loess) model. Both of these hint that the relationship of survival to age is more complex than what is captured in the linear logistic regression model. We return to these data in Chapter 7.

△

1.4 Graphical methods for categorical data

> You can see a lot, just by looking
>
> Yogi Berra

The graphical methods for categorical data described in this book are in some cases straightforward adaptations of more familiar visualization techniques developed for quantitative data. Graphical principles and strategies, and the relations between the visualization approach and traditional statistical methods, are described in a number of sources, including Chambers et al. (1983), Cleveland (1993b), and several influential books by Tufte (Tufte, 1983, 1990, 1997, 2006).

The fundamental idea of statistical graphics as a comprehensive system of visual signs and symbols with a grammar and semantics was first proposed in Jacques Bertin's *Semiology of Graphics* (1983). These ideas were later extended to a computational theory in Wilkinson's *Grammar of Graphics* (2005), and implemented in R in Hadley Wickham's ggplot2 (Wickham and Chang, 2015) package (Wickham, 2009, Wickham and Chang, 2015).

Another perspective on visual data display is presented in Section 1.4.1 focusing on the communication goals of statistical graphics. However, the discrete nature of categorical data implies that some familiar graphic methods need to be adapted, while in other cases we require a new graphic metaphor for data display. These issues are illustrated in Section 1.4.2. Section 1.4.3 discusses the principle of effect ordering for categorical variables in graphs and tables.

1.4.1 Goals and design principles for visual data display

Designing good graphics is surely an art, but as surely, it is one that ought to be informed by science. In constructing a graph, quantitative and qualitative information is encoded by visual features, such as position, size, texture, symbols, and color. This translation is reversed when a person studies a graph. The representation of numerical magnitude and categorical grouping, and the apperception of patterns and their *meaning* must be extracted from the visual display.

There are many views of graphs, of graphical perception, and of the roles of data visualization in discovering and communicating information. On the one hand, one may regard a graphical display as a *stimulus*—a package of information to be conveyed to an idealized observer. From this perspective certain questions are of interest: which form or graphic aspect promotes greater accuracy or speed of judgment (for a particular task or question)? What aspects lead to greatest memorability or impact? Cleveland (Cleveland and McGill, 1984, 1985, Cleveland, 1993a), Spence and Lewandowsky (Lewandowsky and Spence, 1989, Spence, 1990, Spence and Lewandowsky, 1990) have made important contributions to our understanding of these aspects of graphical display.

An alternative view regards a graphical display as an act of *communication*—like a narrative, or even a poetic text or work of art. This perspective places the greatest emphasis on the desired communication goal, and judges the effectiveness of a graphical display in how well that goal is achieved (Friendly and Kwan, 2011). Kosslyn (1985, 1989) and Tufte (1983, 1990, 1997) have articulated this perspective most clearly.

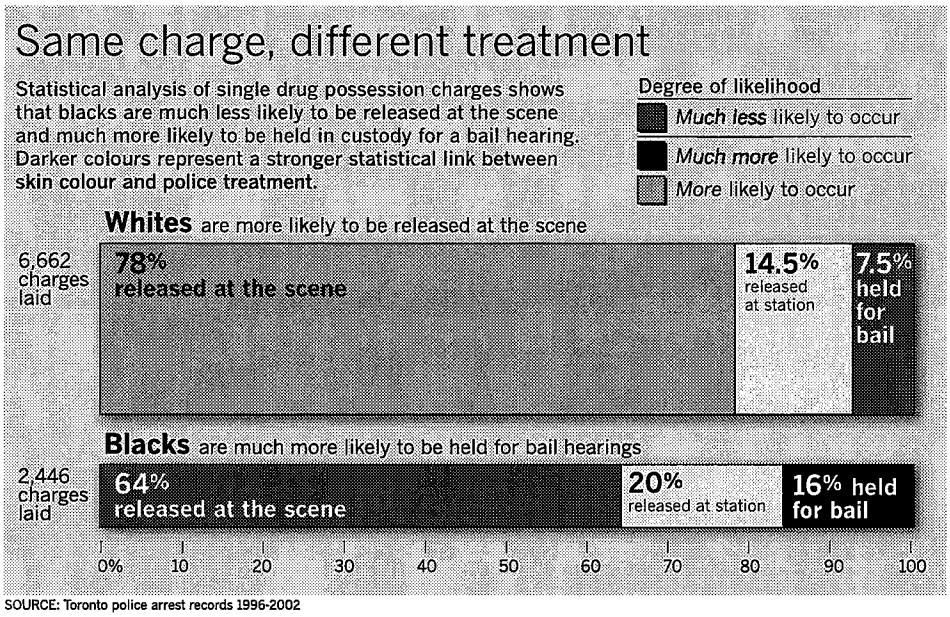


Figure 1.8: Redesign of Figure 1.7 as a presentation graphic. *Source*: Graphics department, *The Toronto Star*, December 11, 2002. Used by permission.

this is hardly a graph that would be clear to a general audience, and would require a good deal of explanation.

In contrast, Figure 1.8 shows a redesign of this as a *presentation graphic* prepared by the *Star* and published on December 11, 2002 in conjunction with a meeting between the newspaper and the Toronto Police Services Board to consider the issue of racial profiling. The police vehemently denied that racial profiling was taking place. The revision makes the point immediately obvious and compelling in the following ways:

- It announces the conclusion in the figure title: "Same charge, different treatment."
- The text box at the top provides the context for this conclusion.
- Skin colors "Brown" and "Other," which appeared less frequently, were removed, and the release categories "Form 10" and "Form 11.1" were combined as "released at station."
- The graphic is still a mosaic display, however, it now shows explicitly the number of charges laid against whites and blacks and the percentage of each treatment.
- The labels for whites and blacks were enhanced by indicating what a reader should see for each.
- The legend for color is titled non-technically as "degree of likelihood."

Clear communication is not achieved without effort. The revised graph required several iterations and emails between the graphic designer and the statistical consultant (the first author of this book) in the few hours available before the newspaper went to press. The main question was, "what are we trying to show here?" Starting with the original Figure 1.7 mosaic, we asked, "what can we remove?" and "what can we add?" to make the message clearer.

△

1.4.2 Categorical data require different graphical methods

We mentioned earlier, and will see in greater detail in Chapter 7 and Chapter 9, that statistical models for discrete response data and for frequency data are close analogs of the linear regression

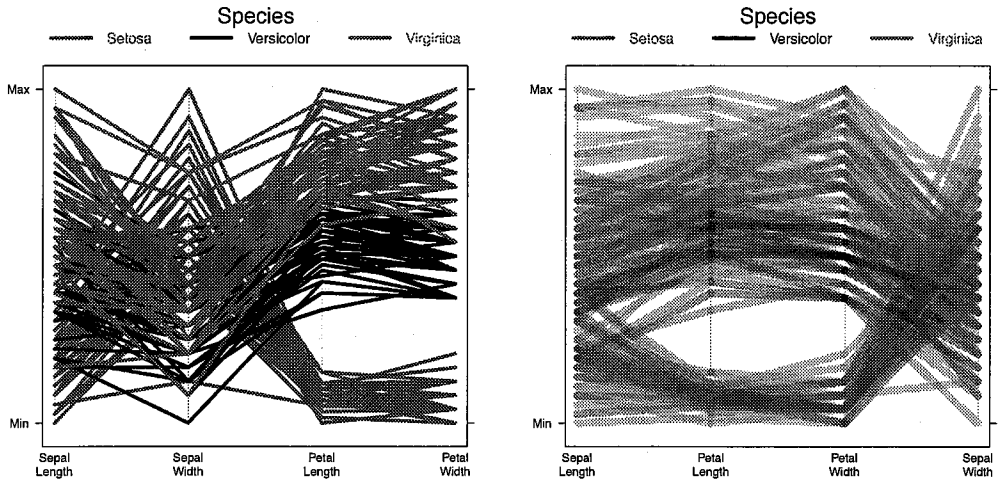


Figure 1.11: Parallel coordinates plots of the Iris data. Left: Default variable order; right: Variables ordered to make the pattern of correlations more coherent.

- graphical controls (sliders, selection boxes, and other widgets) to control details of an analysis (e.g., a smoothing parameter) or graph (colors and other graphic details), to
- higher-level interaction including zooming in or out, drilling down to a data subset, linking multiple displays, selecting terms in a model, and so forth.

The important effect is that the analysis and/or display is immediately re-computed and updated visually.

In addition, ***dynamic graphics*** use animation to show a series of views, as frames in a movie. Adding time as an additional dimension allows far more possibilities, for example showing a rotating view of a 3D graph or showing smooth transitions or interpolations from one view to another.

There are now many packages in R providing interactive and dynamic plots (e.g., rggobi (Temple Lang et al., 2014), iplots (Urbanek and Wichtrey, 2013)) as well as capabilities to incorporate these into interactive documents, presentations, and web pages (e.g., rCharts (Vaidyanathan, 2013), googleVis (Gesmann and de Castillo, 2015), ggvis (Chang and Wickham, 2015)). The animation (Xie, 2014) package facilitates creating animated graphics and movies in a variety of formats. The RStudio editor and development environment[5] provides its own manipulate (RStudio, Inc., 2011) package, as well as the shiny (RStudio, Inc., 2015) framework for developing interactive R web applications.

EXAMPLE 1.8: 512 paths to the White House

Shortly before the 2012 U.S. presidential election (November 2, 2012) *The New York Times* published an interactive graphic,[6] designed by Mike Bostock and Shan Carter,[7] showing the effect that a win for Barack Obama or Mitt Romney in the nine most highly contested states would have on the chances that either candidate would win the presidency.

With these nine states in play there are $2^9 = 512$ possible outcomes, each with a different number of votes in the Electorial College. In Figure 1.12, a win for Obama in Florida and Virginia

[5] `http://www.rstudio.com`.

[6] `http://www.nytimes.com/interactive/2012/11/02/us/politics/paths-to-the-white-house.html`.

[7] see: `https://source.opennews.org/en-US/articles/nyts-512-paths-white-house/`. for a description of their design process.

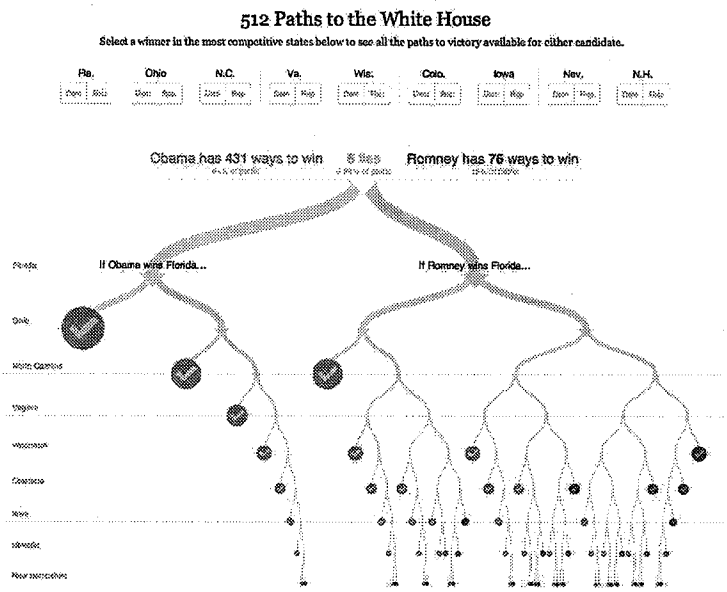


Figure 1.12: 512 paths to the White House. This interactive graphic allows the viewer to select a winner in any one or more of the nine most highly contested U.S. states and highlights the number of paths leading to a win by Obama or Romney, sorted and weighted by the number of Electoral College votes. *Source*: Mike Bostock & Shan Carter, *New York Times* interactive, November 2, 2012. Used by permission.

was selected, with wins for Romney in Ohio and North Carolina. Most other selections also lead to a win by Obama, but those with the most votes are made most visible at the top. An R version of this chart was created using the rCharts package.[8] The design of this graphic as a ***binary tree*** was chosen here, but another possibility would be a ***treemap*** graphic (Shneiderman, 1992) or a mosaic plot.

△

1.4.5 Visualization = Graphing + Fitting + Graphing ...

> Look here, upon this picture, and on this.
>
> Shakespeare, Hamlet

Statistical summaries, hypothesis tests, and the numerical parameters derived in fitted models are designed to capture a particular feature of the data. A quick analysis of the data from Table 1.1, for example, shows that 1198/2691 = 44.5% of male applicants were admitted, compared to 557/1835 = 30.4% of female applicants.

Statistical tests give a Pearson χ^2 of 92.2 with 1 degree of freedom for association between admission and gender ($p < 0.001$), and various measures for the strength of association. Expressed in terms of the ***odds ratio***, males were apparently 1.84 times as likely to be admitted as females, with 99% confidence bounds $(1.56, 2.17)$. Each of these numbers expresses some part of the relationship between gender and admission in the Berkeley data. Numerical summaries such as these are each designed to compress the information in the data, focusing on some particular feature.

[8] `http://timelyportfolio.github.io/rCharts_512paths/`

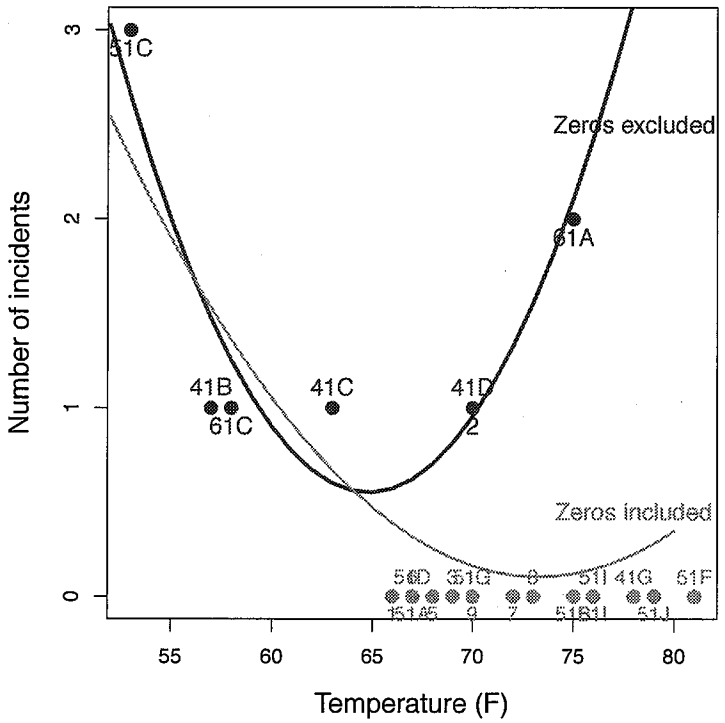


Figure 1.14: Re-drawn version of the NASA pre-launch graph, showing the locations of the excluded observations and with fitted quadratics for both sets of observations.

- **Data plots**: these are well-known. They help answer questions like: (a) What do the data look like? (b) Are there unusual features? (c) What kinds of summaries would be useful?

  An immediate (but bad) example is the plot of failures of O-rings against temperature in Figure 1.13. Many other examples appear throughout Chapter 3, using barplots for discrete distributions, and Chapter 4, using various graphic forms to display frequencies in two-way tables.

- **Model plots**: these are less well-known as such, but also help answer important questions: (a) What does the model "look" like? (plot predicted values); (b) How does the model change when its *parameters* change? (plot competing models); (c) How does the model change when the *data* is changed? (e.g., influence plots).

  Models are simplifed descriptions of data. In Section 5.8 we use mosaic plots to show what loglinear models "look like" (e.g., Figure 5.31). Plots for correspondence analysis methods (Chapter 6) show the relationships among table variables as fitted points in a two-dimensional space. Effect plots (Section 7.3.3) show fitted values for logistic regression models. Models for ordinal variables in terms of log odds ratios (Section 10.1.2) can also be illustrated in terms of simple models plots (Figure 10.4).

- **Data + Model plots** combine these features, and lead to other questions: (a) How well does a model fit the data? (b) Does a model fit uniformly good or bad, or just good/bad in some regions? (c) How can a model be improved? (d) Model *uncertainty*: show confidence/prediction intervals or regions. (e) Data *support*: where is data too "thin" to make a difference in competing models?

  Figure 1.2 and Figure 1.3, Figure 1.4 show several data+model plots for the space shuttle and

Donner data respectively, both showing confidence bands for predicted values. Figure 1.14 is an another example, comparing two models for the space shuttle data. The model-building methods described in Chapter 7–Chapter 11 make frequent use of data+model plots.

1.4.7 The 80–20 rule

The Italian economist Vilfredo Pareto observed in 1906 that 80% of the land in Italy was owned by 20% of the population and this ratio also applied in other countries. It also applied to the yield of peas from peapods in his garden (Pareto, 1971). This idea became known as the ***Pareto principle*** or the ***80–20 rule***. The particular 80/20 ratio is not as important as the more general idea of the uneven distribution of results and causes in a variety of areas.

Common applications are the rules of thumb that: (a) in business 80% of sales come from 20% of clients; (b) in criminology 80% of crimes are said to be committed by 20% of the population. (c) In software development, it is said that 80% of errors and (d) crashes can be eliminated by fixing the top 20% most-reported bugs or that 80% of errors reside in 20% of the code.

The ***Pareto chart*** was designed to display the frequency distribution of a variable with a histogram or bar chart together with a cumulative line graph to highlight the most frequent category, and the ***Pareto distribution*** gives a mathematical form to such distributions with a parameter α (the *Pareto index*) reflecting the degree of inequality.

Applied to statistical graphics, the precept is that

20% of your effort can generate 80% of your desired result in producing a given plot.

This is good news for exploratory graphs you produce for yourself. Very often, the default settings will give a reasonable result, or you will see immediately something simple to add or change to make the plot easier to understand.

The bad news is the corollary of this rule:

80% of your effort may be required to produce the remaining 20% of a finished graph.

This is particularly important for presentation graphs, where several iterations may be necessary to get it right (or right enough) for your communication purposes. Some important details are:

graph title A presentation graphic can be more effective when it announces the main point or conclusion in the graphic title, as in Figure 1.8.

axis and value labels Axes should be labelled with meaningful variable descriptions (and perhaps the data units) rather than just plot defaults (e.g., "Temperature (degrees F)" in Figure 1.2, not `temp`). Axis values are often more of a challenge for categorical variables, where their text labels often overlap, requiring abbreviation, a smaller font, or text rotation.

grouping attributes Meaningfully different subsets of the data should be rendered with distinct visual attributes such as color, shape, and line style, and sometimes with more than one.

legends and direct labels Different data groups in a graphic display shown by color, shape, etc., usually need at least a graphic legend defining the symbols and group labels. Sometimes you can do better by applying the labels directly to the graphical elements,[10] as was done in Figure 1.14.

legibility A common failure in presentation graphs in journals and lectures is the use of text fonts too small to be read easily. One rule of thumb is to hold the graph at arms length for a journal and put it on the floor for a lecture slide. If you can't read the labels, the font is too small.

[10] For example, the `identify()` function allows points in a plot to be labeled interactively with a mouse. The directlabels (Hocking, 2013) package provides a general method for a variety of plots.

plot annotations Beyond the basic graphic data display, additional annotations can add considerable information to interpret the context or uncertainty, as in the use of plot envelopes to show confidence bands or regions (see Figure 1.3 and Figure 1.4).

aspect ratio Line graphs (such as Figure 3.1) are often easiest to understand when the ratio of height to width is such that line segments have an average slope near 1.0 (Cleveland et al., 1988). In R, you can easily manipulate a graph window manually with a mouse to observe this effect and find an aspect ratio that looks right.

Moreover, in graphs for biplots and correspondence analysis (Chapter 6), interpretation involves distances between points and angles between line segments. This requires an aspect ratio that equates the units on the axes. Careful software will do this for you,[11] and you should resist the temptation to re-shape the plot.

colors Whereas a good choice of colors can greatly enhance a graphical display, badly chosen colors, ignoring principles of human perception, can actually spoil it. First, considering that graphs are often reproduced in black and white and a significant percentage of the human population is affected by color deficiencies, important information should not be coded by color alone without careful thought.

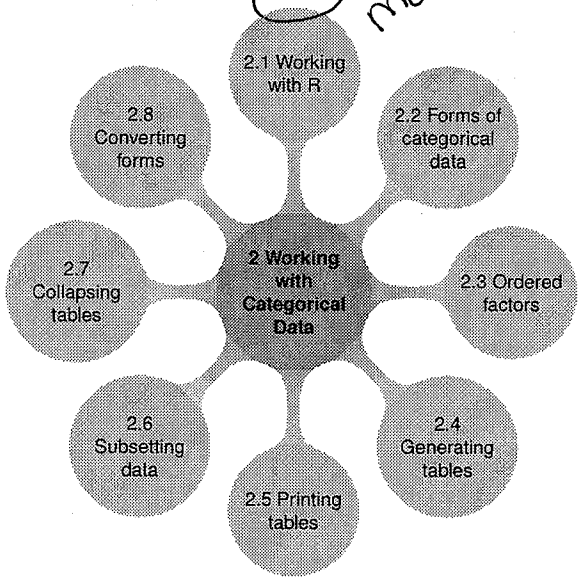
Second, color palettes should be chosen carefully to put the desired emphasis on the information visualized. For example, consider Figure 1.15 showing qualitative color palettes (appropriate for unordered categories) taken from two different color spaces: Hue-Saturation-Value (HSV) and Hue-Chroma-Luminance (HCL), where only the hue is varied. Whereas one would expect such a palette to be balanced with respect to colorfulness and brightness, the red colors in the left (HSV) color wheel are generally perceived to be more more intense and flashy than the corresponding blue colors, and the highly saturated dark blue dominates the wheel. Consequently, areas shaded with these colors may appear more important than others in an uncontrolled way, distracting from the information to be conveyed. In contrast, the colors from the right (HCL) wheel are all balanced to the same gray level and in "harmony." These clearly should be preferred whenever categories of the same importance shall be compared.

Another related perception rule prescribes that lighter and darker colors should not be mixed in a display where areas should be compared since lighter colors look larger than darker ones. More background information on the choice of "good" colors for statistical graphics can be found in Zeileis et al. (2009).

visual impact Somewhat related, important features of a display should be visually distinguished from the less important. This may be achieved by different color or gray shading levels, or simply by contrasting filled with non-filled geometric shapes, or a different density of shading lines. One useful test for visual impact is to put a printed copy of a graph on the floor, rise up, and see what stands out.

Nearly all of the graphs in this book were produced using R code in scripts saved as files and embedded in the text (via the knitr package). This has the advantages of reproducibility and enhancement: just re-run the code, or tweak it to improve a graph. If this is too hard, you can always use an external graphics editor (Gimp, Inkscape, Adobe Illustrator, etc.) to make improvements manually.

[11] For example using the graphics parameter `asp=1`, `eqsplot()` in MASS (Ripley, 2015a), or the equivalents in lattice (`aspect="iso"`) and ggplot2 (`coord_equal`).



2

Working with Categorical Data

Creating and manipulating categorical data sets requires some skills and techniques in R beyond those ordinarily used for quantitative data. This chapter illustrates these for the main formats for categorical data: case form, frequency form and table form.

> I'm a tidy sort of bloke. I don't like chaos. I kept records in the record rack, tea in the tea caddy, and pot in the pot box
>
> George Harrison, from
> `http://www.brainyquote.com/quotes/keywords/tidy.html`

Categorical data can be represented as data sets in various formats: case form, frequency form, and table form. This chapter describes and illustrates the skills and techniques in R needed to input, create, and manipulate R data objects to represent categorical data. More importantly, you also need to be able to convert these from one form to another for the purposes of statistical analysis and visualization, which are the subject of the remainder of the book.

As mentioned earlier, this book assumes that you have at least a basic knowledge of the R language and environment, including interacting with the R console (Rgui for Windows, R.app for Mac OS X) or some other editor/environment (e.g., R Studio), loading and using R functions in packages (e.g., `library(vcd)`) getting help for these from R (e.g., `help(matrix)`), etc. This chapter is therefore devoted to covering those topics needed in the book beyond such basic skills.[1]

[1] Some excellent introductory treatments of R are: Fox and Weisberg (2011a, Chapter 2), Maindonald and Braun (2007), and Dalgaard (2008). Tom Short's *R Reference Card*, `http://cran.us.r-project.org/doc/contrib/Short-refcard.pdf`, is a handy 4-page summary of the main functions. The web sites Quick-R `http://www.statmethods.net/` and Cookbook for R `http://www.cookbook-r.com/` provide very helpful examples, organized by topics and tasks.

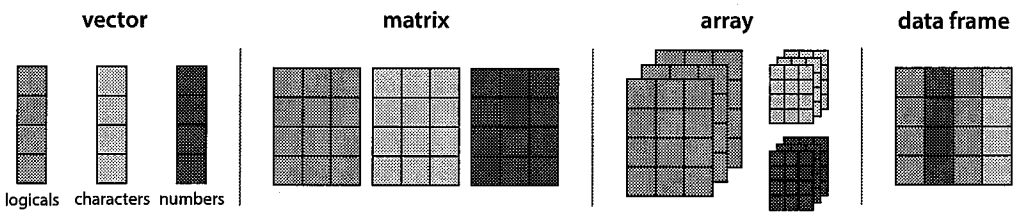


Figure 2.1: Principal data structures and data types in R. Colors represent different data types: numeric, character, logical.

2.1 Working with R data: vectors, matrices, arrays, and data frames

R has a wide variety of data structures for storing, manipulating, and calculating with data. Among these, vectors, matrices, arrays, and data frames are most important for the material in this book.

In R, a ***vector*** is a collection of values, like numbers, character strings, or logicals (`TRUE`, `FALSE`), and often correspond to a variable in some analysis. Matrices are rectangular arrays like a traditional table, composed of vectors in their columns or rows. Arrays add additional dimensions, so that, for example, a 3-way table can be represented as composed of rows, columns, and layers. An important consideration is that the values in vectors, matrices and arrays must all be of the same *mode*, e.g., numbers or character strings. A ***data frame*** is a rectangular table, like a traditional data set in other statistical environments, and composed of rows and columns like a matrix, but allowing variables (columns) of different types. These data structures and the types of data they can contain are illustrated in Figure 2.1. A more general data structure is a *list*, a generic vector that can contain any other types of objects (including lists, allowing for *recursive* data structures). A data frame is basically a list of equally sized vectors, each representing a column of the data frame.

2.1.1 Vectors

The simplest data structure in R is a ***vector***, a one-dimensional collection of elements of the same type. An easy way to create a vector is with the `c()` function, which combines its arguments. The following examples create and print vectors of length 4, containing numbers, character strings, and logical values, respectively:

```
> c(17, 20, 15, 40)
[1] 17 20 15 40
> c("female", "male", "female", "male")
[1] "female" "male"   "female" "male"
> c(TRUE, TRUE, FALSE, FALSE)
[1]  TRUE  TRUE FALSE FALSE
```

To store these values in variables, R uses the assignment operator (`<-`) or equals sign (`=`). This creates a variable named on the left-hand side. An assignment doesn't print the result, but a bare expression does, so you can assign and print by surrounding the assignment with `()`.

objects, which are specialized forms of matrices and arrays containing integer frequencies in a contingency table. These are discussed in more detail below (Section 2.4).

2.1.4 Data frames

Data frames are the most commonly used form of data in R and more general than matrices in that they can contain columns of different types. For statistical modeling, data frames play a special role, in that many modeling functions are designed to take a data frame as a `data=` argument, and then find the variables mentioned within that data frame. Another distinguishing feature is that discrete variables (columns) like character strings (`"M"`, `"F"`) or integers (`1, 2, 3`) in data frames can be represented as *factor*s, which simplifies many statistical and graphical methods.

A data frame can be created using keyboard input with the `data.frame()` function, applied to a list of objects, `data.frame(a, b, c, ...)`, each of which can be a vector, matrix, or another data frame, but typically all containing the same number of rows. This works roughly like `cbind()`, collecting the arguments as columns in the result.

The following example generates `n = 100` random observations on three discrete factor variables, `A, B, sex`, and a numeric variable, `age`. As constructed, all of these are statistically independent, since none depends on any of the others. The function `sample()` is used here to generate `n` random samples from the first argument allowing replacement (`replace = TRUE`). The `rnorm()` function produces a vector of `n` normally distributed values with mean 30 and standard deviation 5. The call to `set.seed()` guarantees the reproducibility of the resulting data. Finally, all four variables are combined into the data frame `mydata`.

```
> set.seed(12345)    # reproducibility
> n <- 100
> A <- factor(sample(c("a1", "a2"), n, replace = TRUE))
> B <- factor(sample(c("b1", "b2"), n, replace = TRUE))
> sex <- factor(sample(c("M", "F"), n, replace = TRUE))
> age <- round(rnorm(n, mean = 30, sd = 5))
> mydata <- data.frame(A, B, sex, age)
> head(mydata, 5)

   A  B sex age
1 a2 b1   F  22
2 a2 b2   F  33
3 a2 b2   M  31
4 a2 b2   F  26
5 a1 b2   F  29

> str(mydata)

'data.frame': 100 obs. of  4 variables:
 $ A  : Factor w/ 2 levels "a1","a2": 2 2 2 2 1 1 1 2 2 2 ...
 $ B  : Factor w/ 2 levels "b1","b2": 1 2 2 2 2 2 2 2 1 1 ...
 $ sex: Factor w/ 2 levels "F","M": 1 1 2 1 1 1 2 2 1 1 ...
 $ age: num  22 33 31 26 29 29 38 28 30 27 ...
```

Rows, columns, and individual values in a data frame can be manipulated in the same way as a matrix, using subscripting (`[,]`). Additionally, variables can be extracted using the `$` operator:

```
> mydata[1,2]

[1] b1
Levels: b1 b2

> mydata$sex
```

```
  [1] F F M F F F M M F F M F M M F M M F F M M M M F F F F M M F
 [31] M F M F F F F F M M F F F F F F F F F M F M F M M F F M M M M
 [61] F F F F F M M F F F M M M F F M F M M F M F M M M M M F F F
 [91] F F M M F M F M F M
Levels: F M

> ##same as: mydata[,"sex"] or mydata[,3]
```

Values in data frames can also be edited conveniently using, e.g., `fix(mydata)`, opening a simple, spreadsheet-like editor.

For real data sets, it is usually most convenient to read these into R from external files, and this is easiest using plain text (ASCII) files with one line per observation and fields separated by commas (or tabs), and with a first header line giving the variable names—called *comma-separated* or CSV format. If your data is in the form of Excel, SAS, SPSS, or other file format, you can almost always export that data to CSV format first.[3]

The function `read.table()` has many options to control the details of how the data are read and converted to variables in the data frame. Among these some important options are:

`header` indicates whether the first line contains variable names. The default is `FALSE` unless the first line contains one fewer field than the number of columns;

`sep` (default: `""`, meaning white space, i.e., one or more spaces, tabs or newlines) specifies the separator character between fields;

`stringsAsFactors` (default: `TRUE`) determines whether character string variables should be converted to factors;

`na.strings` (default: `"NA"`) refers to one or more strings that are interpreted as missing data values (`NA`);

For delimited files, `read.csv()` and `read.delim()` are convenient wrappers to `read.table()`, with default values `sep=","` and `sep="\t"` respectively, and `header=TRUE`.

EXAMPLE 2.1: Arthritis treatment

The file `Arthritis.csv` contains data in CSV format from Koch and Edwards (1988), representing a double-blind clinical trial investigating a new treatment for rheumatoid arthritis with 84 patients.[4] The first ("header") line gives the variable names. Some of the lines in the file are shown below, with `...` representing omitted lines:

```
ID,Treatment,Sex,Age,Improved
57,Treated,Male,27,Some
46,Treated,Male,29,None
77,Treated,Male,30,None
17,Treated,Male,32,Marked
 ...
42,Placebo,Female,66,None
15,Placebo,Female,66,Some
71,Placebo,Female,68,Some
1,Placebo,Female,74,Marked
```

We read this into R using `read.table()` as shown below:

[3] The foreign (R Core Team, 2015) package contains specialized functions to *directly* read data stored by Minitab, SAS, SPSS, Stata, Systat, and other software. There are also a number of packages for reading (and writing) Excel spreadsheets directly (gdata (Warnes et al., 2014), XLConnect (Mirai Solutions GmbH, 2015), xlsx (Dragulescu, 2014)). The R manual, *R Data Import/Export* covers many other variations, including data in relational data bases.

[4] This data set can be created using: `library(vcd); write.table(Arthritis, file = "Arthritis.csv", quote = FALSE, sep = ",")`.

```
> path <- "ch02/Arthritis.csv" ## set path
> ## for convenience, use path <- file.choose() to retrieve a path
> ## then, use file.show(path) to inspect the data format
> Arthritis <- read.table(path, header = TRUE, sep = ",")
> str(Arthritis)

'data.frame': 84 obs. of  5 variables:
 $ ID        : int  57 46 77 17 36 23 75 39 33 55 ...
 $ Treatment: Factor w/ 2 levels "Placebo","Treated": 2 2 2 2 2 2 2 2 2 2 ...
 $ Sex       : Factor w/ 2 levels "Female","Male": 2 2 2 2 2 2 2 2 2 2 ...
 $ Age       : int  27 29 30 32 46 58 59 59 63 63 ...
 $ Improved : Factor w/ 3 levels "Marked","None",..: 3 2 2 1 1 1 2 1 2 2 ...
```

Note that the character variables `Treatment`, `Sex`, and `Improved` were converted to factors, and the levels of those variables were ordered *alphabetically*. This often doesn't matter much for binary variables, but here, the response variable `Improved` has levels that should be considered *ordered*, as `"None"`, `"Some"`, `"Marked"`. We can correct this here by re-assigning `Arthritis$Improved` using `ordered()`. The topic of re-ordering variables and levels in categorical data is considered in more detail in Section 2.3.

```
> levels(Arthritis$Improved)

[1] "Marked" "None"   "Some"

> Arthritis$Improved <- ordered(Arthritis$Improved,
+                               levels = c("None", "Some", "Marked"))
```

△

2.2 Forms of categorical data: case form, frequency form, and table form

As we saw in Chapter 1, categorical data can be represented as ordinary data sets in case form, but the discrete nature of factors or stratifying variables allows the same information to be represented more compactly in summarized form with a frequency variable for each cell of factor combinations, or in tables. Consequently, we sometimes find data created or presented in one form (e.g., a spreadsheet data set, a two-way table of frequencies) and want to input that into R. Once we have the data in R, it is often necessary to manipulate the data into some other form for the purposes of statistical analysis, visualizing results, and our own presentation. It is useful to understand the three main forms of categorical data in R and how to work with them for our purposes.

2.2.1 Case form

Categorical data in case form are simply data frames, with one or more discrete classifying variables or response variables, most conveniently represented as factors or ordered factors. In case form, the data set can also contain numeric variables (covariates or other response variables) that cannot be accommodated in other forms.

As with any data frame, `X`, you can access or compute with its attributes using `nrow(X)` for the number of observations, `ncol(X)` for the number of variables, `names(X)` or `colnames(X)` for the variable names, and so forth.

EXAMPLE 2.2: Arthritis treatment

The *Arthritis* data is available in case form in the vcd package. There are two explanatory factors: `Treatment` and `Sex`. `Age` is a numeric covariate, and `Improved` is the response—an ordered factor, with levels `"None" < "Some" < "Marked"`. Excluding `Age`, we would have a $2 \times 2 \times 3$ contingency table for `Treatment`, `Sex`, and `Improved`.

```
> data("Arthritis", package = "vcd")  # load the data
> names(Arthritis)      # show the variables

[1] "ID"        "Treatment" "Sex"       "Age"       "Improved"

> str(Arthritis)        # show the structure

'data.frame': 84 obs. of  5 variables:
 $ ID       : int  57 46 77 17 36 23 75 39 33 55 ...
 $ Treatment: Factor w/ 2 levels "Placebo","Treated": 2 2 2 2 2 2 2 2 2 2 ...
 $ Sex      : Factor w/ 2 levels "Female","Male": 2 2 2 2 2 2 2 2 2 2 ...
 $ Age      : int  27 29 30 32 46 58 59 59 63 63 ...
 $ Improved : Ord.factor w/ 3 levels "None"<"Some"<..: 2 1 1 3 3 3 1 3 1 1 ...

> head(Arthritis, 5)    # first 5 observations, same as Arthritis[1:5,]

  ID Treatment  Sex Age Improved
1 57   Treated Male  27     Some
2 46   Treated Male  29     None
3 77   Treated Male  30     None
4 17   Treated Male  32   Marked
5 36   Treated Male  46   Marked
```

△

2.2.2 Frequency form

Data in frequency form is also a data frame, containing one or more discrete factor variables and a frequency variable (often called `Freq` or `count`) representing the number of basic observations in that cell.

This is an alternative representation of a table form data set considered below. In frequency form, the number of cells in the equivalent table is `nrow(X)`, and the total number of observations is the sum of the frequency variable, `sum(X$Freq)`, `sum(X[,"Freq"])` or a similar expression.

EXAMPLE 2.3: General social survey

For small frequency tables, it is often convenient to enter them in frequency form using `expand.grid()` for the factors and `c()` to list the counts in a vector. The example below, from Agresti (2002), gives results for the 1991 General Social Survey, with respondents classified by sex and party identification. As a table, the data look like this:

	party		
sex	dem	indep	rep
female	279	73	225
male	165	47	191

We use `expand.grid()` to create a 6 × 2 matrix containing the combinations of `sex` and `party` with the levels for `sex` given first, so that this varies most rapidly. Then, input the frequencies in the table by columns from left to right, and combine these two results with `data.frame()`.

```
> # Agresti (2002), table 3.11, p. 106
> tmp <- expand.grid(sex = c("female", "male"),
+                    party = c("dem", "indep", "rep"))
> tmp

     sex party
1 female   dem
2   male   dem
3 female indep
4   male indep
5 female   rep
6   male   rep
```

```
  < 15k      1       3       10      6
  15-25k     2       3       10      7
  25-40k     1       6       14     12
  > 40k      0       1        9     11
```

△

2.3 Ordered factors and reordered tables

As we saw above (Example 2.1), the levels of factor variables in data frames (case form or frequency form) can be re-ordered (and the variables declared as ordered factors) using `ordered()`. As well, the order of the factor values themselves can be rearranged by sorting the data frame using `sort()`.

However, in table form, the values of the table factors are ordered by their position in the table. Thus in the *JobSat* data, both `income` and `satisfaction` represent ordered factors, and the *positions* of the values in the rows and columns reflect their ordered nature, but only implicitly.

Yet, for analysis or graphing, there are occasions when you need *numeric* values for the levels of ordered factors in a table, e.g., to treat a factor as a quantitative variable. In such cases, you can simply re-assign the `dimnames` attribute of the table variables. For example, here, we assign numeric values to `income` as the middle of their ranges, and treat `satisfaction` as equally spaced with integer scores.

```
> dimnames(JobSat)$income <- c(7.5, 20, 32.5, 60)
> dimnames(JobSat)$satisfaction <- 1:4
```

A related case is when you want to preserve the character labels of table dimensions, but also allow them to be sorted in some particular order. A simple way to do this is to prefix each label with an integer index using `paste()`.

```
> dimnames(JobSat)$income <-
+     paste(1:4, dimnames(JobSat)$income, sep = ":")
> dimnames(JobSat)$satisfaction <-
+     paste(1:4, dimnames(JobSat)$satisfaction, sep = ":")
```

A different situation arises with tables where you want to *permute* the levels of one or more variables to arrange them in a more convenient order without changing their labels. For example, in the *HairEyeColor* table, hair color and eye color are ordered arbitrarily. For visualizing the data using mosaic plots and other methods described later, it turns out to be more useful to assure that both hair color and eye color are ordered from dark to light. Hair colors are actually ordered this way already: `"Black"`, `"Brown"`, `"Red"`, `"Blond"`. But eye colors are ordered as `"Brown"`, `"Blue"`, `"Hazel"`, `"Green"`. It is easiest to re-order the eye colors by indexing the columns (dimension 2) in this array to a new order, `"Brown"`, `"Hazel"`, `"Green"`, `"Blue"`, giving the indices of the old levels in the new order (here: 1,3,4,2). Again `str()` is your friend, showing the structure of the result to check that the result is what you want.

```
> data("HairEyeColor", package = "datasets")
> HEC <- HairEyeColor[, c(1, 3, 4, 2), ]
> str(HEC)

 num [1:4, 1:4, 1:2] 32 53 10 3 10 25 7 5 3 15 ...
 - attr(*, "dimnames")=List of 3
  ..$ Hair: chr [1:4] "Black" "Brown" "Red" "Blond"
  ..$ Eye : chr [1:4] "Brown" "Hazel" "Green" "Blue"
  ..$ Sex : chr [1:2] "Male" "Female"
```

2.6.3 Subsetting data frames

Data available in data frames (frequency or case form) can also be subsetted, either by using indexes on the rows and/or columns, or, more conveniently, by applying the `subset()` function. The following statement will extract the `Treatment` and `Improved` variables for all female patients older than 68:

```
> rows <- Arthritis$Sex == "Female" & Arthritis$Age > 68
> cols <- c("Treatment", "Improved")
> Arthritis[rows, cols]

   Treatment Improved
39   Treated     None
40   Treated     Some
41   Treated     Some
84   Placebo   Marked
```

Note the use of the single `&` for the logical expression selecting the rows. The same result can be achieved more conveniently using the `subset()` function, first taking the data set, followed by an expression for selecting the rows (evaluated in the context of the data frame), and then an expression for selecting the columns:

```
> subset(Arthritis, Sex == "Female" & Age > 68,
+        select = c(Treatment, Improved))

   Treatment Improved
39   Treated     None
40   Treated     Some
41   Treated     Some
84   Placebo   Marked
```

Note the non-standard evaluation of `c(Treatment, Improved)`: the meaning of `c()` is not "combine the two columns into a single vector", but "select both from the data frame." Likewise, columns can be removed using `-` on column names, which is not possible using standard indexing in matrices or data frames:

```
> subset(Arthritis, Sex == "Female" & Age > 68,
+        select = -c(Age, ID))

   Treatment    Sex Improved
39   Treated Female     None
40   Treated Female     Some
41   Treated Female     Some
84   Placebo Female   Marked
```

2.7 Collapsing tables

2.7.1 Collapsing over table factors: `aggregate()`, `margin.table()` and `apply()`

It sometimes happens that we have a data set with more variables or factors than we want to analyse, or else, having done some initial analyses, we decide that certain factors are not important, and so should be excluded from graphic displays by collapsing (summing) over them. For example, mosaic plots and fourfold displays are often simpler to construct from versions of the data collapsed over the factors which are not shown in the plots.

```
> # create some sample data in frequency form
> set.seed(12345)    # reproducibility
> sex <- c("Male", "Female")
> age <- c("10-19", "20-29", "30-39", "40-49", "50-59", "60-69")
> education <- c("low", "med", "high")
> dat <- expand.grid(sex = sex, age = age, education = education)
> counts <- rpois(36, 100)    # random Poisson cell frequencies
> dat <- cbind(dat, counts)
> # make it into a 3-way table
> tab1 <- xtabs(counts ~ sex + age + education, data = dat)
> structable(tab1)

                age 10-19 20-29 30-39 40-49 50-59 60-69
sex    education
Male   low            105    98   123    97    95   105
       med             74   113   114    82    95    85
       high           121   116   104   103    89   100
Female low            107    95   105   116   103    92
       med             96    88    93   118    99   108
       high           120   102    96   103   127    84
```

Now collapse `age` to 20-year intervals, and `education` to 2 levels. In the arguments to `collapse.table()`, levels of `age` and `education` given the same label are summed in the resulting smaller table.

```
> # collapse age to 3 levels, education to 2 levels
> tab2 <- collapse.table(tab1,
+           age = c("10-29", "10-29", "30-49", "30-49", "50-69", "50-69"),
+           education = c("<high", "<high", "high"))
> structable(tab2)

                age 10-29 30-49 50-69
sex    education
Male   <high          390   416   380
       high           237   207   189
Female <high          386   432   402
       high           222   199   211
```

△

2.8 Converting among frequency tables and data frames

As we've seen, a given contingency table can be represented equivalently in case form, frequency form, and table form. However, some R functions were designed for one particular representation. Table 2.1 gives an overview of some handy tools (with sketched usage) for converting from one form to another, discussed below.

2.8.1 Table form to frequency form

A contingency table in table form (an object of class "table") can be converted to a data frame in frequency form with `as.data.frame()`.[6] The resulting data frame contains columns representing the classifying factors and the table entries (as a column named by the `responseName` argument, defaulting to `Freq`). The function `as.data.frame()` is the inverse of `xtabs()`, which converts a data frame to a table.

[6] Because R is object-oriented, this is actually a shorthand for the function `as.data.frame.table()`, which is automatically selected for objects of class "table".

Table 2.1: Tools for converting among different forms for categorical data

From this	To this: Case form	Frequency form	Table form
Case form	—	`Z <- xtabs(~ A + B)` `as.data.frame(Z)`	`table(A, B)`
Frequency form	`expand.dft(X)`	—	`xtabs(count ~ A + B)`
Table form	`expand.dft(X)`	`as.data.frame(X)`	—

EXAMPLE 2.9: General social survey

Convert the `GSStab` object in table form to a data.frame in frequency form. By default, the frequency variable is named `Freq`, and the variables `sex` and `party` are made factors.

```
> as.data.frame(GSStab)

     sex party Freq
1 female   dem  279
2   male   dem  165
3 female indep   73
4   male indep   47
5 female   rep  225
6   male   rep  191
```

△

In addition, there are situations where numeric table variables are represented as factors, but you need to convert them to numerics for calculation purposes.

EXAMPLE 2.10: Death by horse kick

For example, we might want to calculate the weighted mean of `nDeaths` in the *HorseKicks* data. Using `as.data.frame()` won't work here, because the variable `nDeaths` becomes a factor.

```
> str(as.data.frame(HorseKicks))

'data.frame': 5 obs. of  2 variables:
 $ nDeaths: Factor w/ 5 levels "0","1","2","3",..: 1 2 3 4 5
 $ Freq   : int  109 65 22 3 1
```

One solution is to use `data.frame()` directly and `as.numeric()` to coerce the table names to numbers.

```
> horse.df <- data.frame(nDeaths = as.numeric(names(HorseKicks)),
+                        Freq = as.vector(HorseKicks))
> str(horse.df)

'data.frame': 5 obs. of  2 variables:
 $ nDeaths: num  0 1 2 3 4
 $ Freq   : int  109 65 22 3 1

> horse.df

  nDeaths Freq
1       0  109
2       1   65
3       2   22
4       3    3
5       4    1
```

	nDeaths
0	109
1	65
2	22
3	3
4	1

```
> tab <- as.data.frame(HorseKicks)
> colnames(tab) <- c("nDeaths", "Freq")
> print(xtable(tab), include.rownames = FALSE,
+       include.colnames = TRUE)
```

nDeaths	Freq
0	109
1	65
2	22
3	3
4	1

There are many more options to control the LaTeX details and polish the appearance of the table; see `help(xtable)` and `vignette("xtableGallery", package = "xtable")`.

Finally, in Chapter 3, we display a number of similar one-way frequency tables in a transposed form to save display space. Table 3.3 is the finished version we show there. The code below uses the following techniques: (a) `addmargins()` is used to show the sum of all the frequency values; (b) `t()` transposes the data frame to have 2 rows; (c) `rownames()` assigns the labels we want for the rows; (d) using the `caption` argument provides a table caption, and a numbered table in LaTeX; (d) column alignment (`"r"` or `"l"`) for the table columns is computed as a character string used for the `align` argument.

```
> horsetab <- t(as.data.frame(addmargins(HorseKicks)))
> rownames(horsetab) <- c( "Number of deaths", "Frequency" )
> horsetab <- xtable(horsetab, digits = 0,
+       caption = "von Bortkiewicz's data on deaths by horse kicks",
+       align = paste0("l|", paste(rep("r", ncol(horsetab)),
+                                  collapse = "")) 
+       )
> print(horsetab, include.colnames = FALSE, caption.placement="top")
```

Table 2.2: von Bortkiewicz's data on deaths by horse kicks

Number of deaths	0	1	2	3	4	Sum
Frequency	109	65	22	3	1	200

For use in a web page, blog, or Word document, you can use `type="HTML"` in the call to `print()` for "xtable" objects.

```
> tv_data <- read.table(file.choose())
```

We could use this data in frequency form for analysis by renaming the variables, and converting the integer-coded factors `V1` – `V4` to R factors. The lines below use the function `within()` to avoid having to use `TV.dat$Day <- factor(TV.dat$Day)` etc., and only supply labels for the first variable.

```
> TV_df <- tv_data
> colnames(TV_df) <- c("Day", "Time", "Network", "State", "Freq")
> TV_df <- within(TV_df, {
+             Day <- factor(Day,
+                           labels = c("Mon", "Tue", "Wed", "Thu", "Fri"))
+             Time <- factor(Time)
+             Network <- factor(Network)
+             State <- factor(State)
+         })
```

Alternatively, we could just reshape the frequency column (`V5` or `tv_data[,5]`) into a 4-way array. In the lines below, we rely on the facts that (a) the table is complete—there are no missing cells, so `nrow(tv_data)` = 825; (b) the observations are ordered so that `V1` varies most rapidly and `V4` most slowly. From this, we can just extract the frequency column and reshape it into an array using the `dim` argument. The level names are assigned to `dimnames(TV)` and the variable names to `names(dimnames(TV))`.

```
> TV <- array(tv_data[,5], dim = c(5, 11, 5, 3))
> dimnames(TV) <-
+     list(c("Mon", "Tue", "Wed", "Thu", "Fri"),
+          c("8:00", "8:15", "8:30", "8:45", "9:00", "9:15",
+            "9:30", "9:45", "10:00", "10:15", "10:30"),
+          c("ABC", "CBS", "NBC", "Fox", "Other"),
+          c("Off", "Switch", "Persist"))
> names(dimnames(TV)) <- c("Day", "Time", "Network", "State")
```

More generally (even if there are missing cells), we can use `xtabs()` to do the cross-tabulation, using `V5` as the frequency variable. Here's how to do this same operation with `xtabs()`:

```
> TV <- xtabs(V5 ~ ., data = tv_data)
> dimnames(TV) <-
+     list(Day = c("Mon", "Tue", "Wed", "Thu", "Fri"),
+          Time = c("8:00", "8:15", "8:30", "8:45", "9:00", "9:15",
+                   "9:30", "9:45", "10:00", "10:15", "10:30"),
+          Network = c("ABC", "CBS", "NBC", "Fox", "Other"),
+          State = c("Off", "Switch", "Persist"))
```

Note that in the lines above, the variable names are assigned directly as the names of the elements in the `dimnames` list.

2.9.2 Subsetting and collapsing

For many purposes, the 4-way table `TV` is too large and awkward to work with. Among the networks, Fox and Other occur infrequently, so we will remove them. We can also cut it down to a 3-way table by considering only viewers who persist with the current station.[8]

[8]This relies on the fact that indexing an array drops dimensions of length 1 by default, using the argument `drop = TRUE`; the result is coerced to the lowest possible dimension.

```
                  Dept
Admit:Gender        A   B   C   D   E   F
  Admitted:Female  89  17 202 131  94  24
  Admitted:Male   512 353 120 138  53  22
  Rejected:Female  19   8 391 244 299 317
  Rejected:Male   313 207 205 279 138 351
```

(a) Try this the long way: convert *UCBAdmissions* to a data frame (`as.data.frame()`), manipulate the factors (e.g., `interaction()`), then convert back to a table (`as.data.frame()`).
(b) Try this the short way: both `ftable()` and `structable()` have `as.matrix()` methods that convert their result to a matrix.

in margin

Exercise 2.8 The data set *VisualAcuity* in vcd gives a $4 \times 4 \times 2$ table as a frequency data frame.

```
> data("VisualAcuity", package = "vcd")
> str(VisualAcuity)

'data.frame': 32 obs. of  4 variables:
 $ Freq  : num  1520 234 117 36 266 ...
 $ right : Factor w/ 4 levels "1","2","3","4": 1 2 3 4 1 2 3 4 1 2 ...
 $ left  : Factor w/ 4 levels "1","2","3","4": 1 1 1 1 2 2 2 2 3 3 ...
 $ gender: Factor w/ 2 levels "male","female": 2 2 2 2 2 2 2 2 2 2 ...
```

(a) From this, use `xtabs()` to create two 4×4 frequency tables, one for each gender.
(b) Use `structable()` to create a nicely organized tabular display.
(c) Use `xtable()` to create a LaTeX or HTML table.

The chapter then describes methods for fitting data to a distribution of a given form, and presents simple but effective graphical methods that can be used to visualize goodness of fit, to diagnose an appropriate model (e.g., does a given data set follow the Poisson or negative binomial?) and to determine the impact of individual observations on estimated parameters.

3.1 Introduction to discrete distributions

Discrete data analysis is concerned with the study of the tabulation of one or more types of events, often categorized into mutually exclusive and exhaustive categories. ***Binary events*** having two outcome categories include the toss of a coin (head/tails), sex of a child (male/female), survival of a patient following surgery (lived/died), and so forth. ***Polytomous events*** have more outcome categories, which may be *ordered* (rating of impairment: low/medium/high, by a physician) and possibly numerically valued (number of dots (pips), 1–6 on the toss of a die) or *unordered* (political party supported: Liberal, Conservative, Greens, Socialist).

In this chapter, we focus largely on one-way frequency tables for a single numerically valued variable. Probability models for such data provide the opportunity to describe or explain the *structure* in such data, in that they entail some data-generating mechanism and provide the basis for testing scientific hypotheses and predicting future results. If a given probability model does not fit the data, this can often be a further opportunity to extend understanding of the data, or the underlying substantive theory, or both.

The remainder of this section gives a few substantive examples of situations where the well-known discrete frequency distributions (binomial, Poisson, negative binomial, geometric, and logarithmic series) might reasonably apply, at least approximately. The mathematical characteristics and properties of these theoretical distributions are postponed to Section 3.2.

In many cases, the data at hand pertain to two types of variables in a one-way frequency table. There is a basic outcome variable, k, taking integer values, $k = 0, 1, \ldots$, and called a ***count***. For each value of k, we also have a ***frequency***, n_k that the count k was observed in some sample. For example, in the study of children in families, the count variable k could be the total number of children or the number of male children; the frequency variable, n_k, would then give the number of families with that basic count k.

3.1.1 Binomial data

Binomial-type data arise as the discrete distribution of the number of "success" events in n independent binary trials, each of which yields a success (yes/no, head/tail, lives/dies, male/female) with a constant probability p.

Sometimes, as in Example 3.1 below, the available data record only the number of successes in n trials, with separate such observations recorded over time or space. More commonly, as in Example 3.2 and Example 3.3, we have available data on the frequency n_k of $k = 0, 1, 2, \ldots n$ successes in the n trials.

EXAMPLE 3.1: Arbuthnot data

Sex ratios, such as births of male to female children, have long been of interest in population studies and demography. Indeed, in 1710, John Arbuthnot (Arbuthnot, 1710) used data on the ratios of male to female christenings in London from 1629–1710 to carry out the first known significance test. The data for these 82 years showed that in *every* year there were more boys than girls. He calculated that, under the assumption that male and female births were equally likely, the probability of 82 years of more males than females was vanishingly small, ($\Pr \approx 4.14 \times 10^{-25}$). He used this to argue that a nearly constant birth ratio > 1 (or $\Pr(\text{Male}) > 0.5$) could be interpreted to show the guiding hand of a divine being.

Arbuthnot's data, along with some other related variables, are available in `Arbuthnot` in the

Table 3.1: Number of male children in 6115 Saxony families of size 12

Males (k)	0	1	2	3	4	5	6	7	8	9	10	11	12	Sum
Families (n_k)	3	24	104	286	670	1,033	1,343	1,112	829	478	181	45	7	6,115

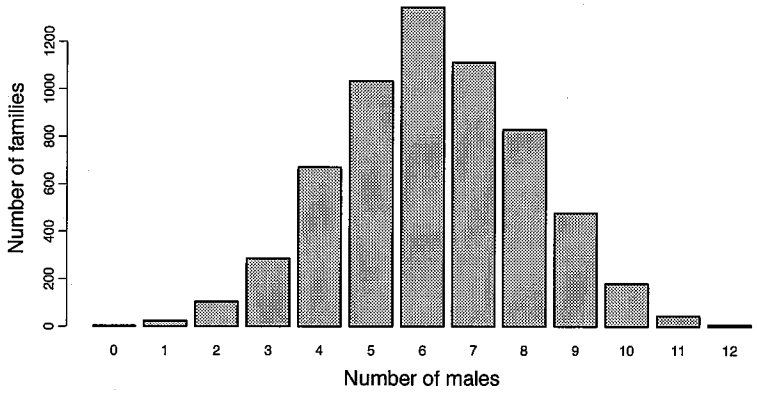


Figure 3.2: Number of males in Saxony families of size 12.

standard form of a complete discrete distribution. The basic outcome variable, $k = 0, 1, \ldots, 12$, is the number of male children in a family, and the frequency variable, n_k is the number of families with that number of boys.

Figure 3.2 shows a bar plot of the frequencies in Table 3.1. It can be seen that the distribution is quite symmetric. The questions of interest here are: (a) how close does the data follow a binomial distribution, with a constant $\Pr(\text{Male}) = p$? (b) is there evidence to reject the hypothesis that $p = 0.5$?

```
> data("Saxony", package = "vcd")
> barplot(Saxony, xlab = "Number of males", ylab = "Number of families",
+         col = "lightblue", cex.lab = 1.5)
```

△

EXAMPLE 3.3: Weldon's dice

Common examples of binomial distributions involve tossing coins or dice, where some event outcome is considered a "success" and the number of successes (k) are tabulated in a long series of trials to give the frequency (n_k) of each basic count, k.

Perhaps the most industrious dice-tosser of all time, W. F. Raphael Weldon, an English evolutionary biologist and joint founding editor of *Biometrika* (with Francis Galton and Karl Pearson), tallied the results of throwing 12 dice 26,306 times. For his purposes, he considered the outcome of 5 or 6 pips showing on each die to be a success, and all other outcomes as failures.

Weldon reported his results in a letter to Francis Galton dated February 2, 1894, in order "to judge whether the differences between a series of group frequencies and a theoretical law ... were more than might be attributed to the chance fluctuations of random sampling" (Kemp and Kemp, 1991). In his seminal paper, Pearson (1900) used Weldon's data to illustrate the χ^2 goodness-of-fit test, as did Kendall and Stuart (1963, Table 5.1, p. 121).

These data are shown here as Table 3.2, in terms of the number of occurrences of a 5 or 6 in the throw of 12 dice. If the dice were all identical and perfectly fair (balanced), one would expect that $p = \Pr\{5 \text{ or } 6\} = \frac{1}{3}$ and the distribution of the number of 5 or 6 would be binomial.

Table 3.2: Frequencies of 5s or 6s in throws of 12 dice

# 5s or 6s (k)	0	1	2	3	4	5	6	7	8	9	10+	Sum
Frequency (n_k)	185	1,149	3,265	5,475	6,114	5,194	3,067	1,331	403	105	18	26,306

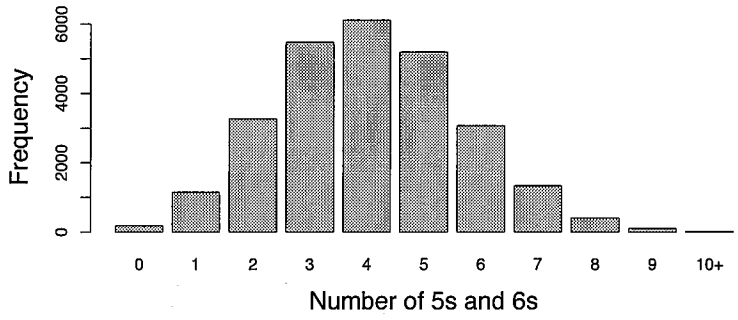


Figure 3.3: Weldon's dice data, frequency distribution of 5s and 6s in throws of 12 dice.

A peculiar feature of these data as presented by Kendall and Stuart (not uncommon in discrete distributions) is that the frequencies of 10–12 successes are lumped together.[2] This grouping must be taken into account in fitting the distribution. This dataset is available as `WeldonDice` in the vcd package. The distribution is plotted in Figure 3.3.

```
> data("WeldonDice", package = "vcd")
> dimnames(WeldonDice)$n56[11] <- "10+"
> barplot(WeldonDice, xlab = "Number of 5s and 6s", ylab = "Frequency",
+         col = "lightblue", cex.lab = 1.5)
```

△

3.1.2 Poisson data

Data of Poisson type arise when we observe the counts of events k within a fixed interval of time or space (length, area, volume) and tabulate their frequencies, n_k. For example, we may observe the number of radioactive particles emitted by a source per second or number of births per hour, or the number of tiger or whale sightings within some geographical regions.

In contrast to binomial data, where the counts are bounded below and above, in Poisson data the counts k are bounded below at 0, but can take integer values with no fixed upper limit. One defining characteristic for the Poisson distribution is for rare events, which occur independently with a small and constant probability, p, in small intervals, and we count the number of such occurrences.

Several examples of data of this general type are given below.

EXAMPLE 3.4: Death by horse kick

One of the oldest and best known examples of a Poisson distribution is the data from von Bortkiewicz (1898) on deaths of soldiers in the Prussian army from kicks by horses and mules,

[2] The unlumped entries are, for (number of 5s or 6s: frequency) — (10: 14); (11: 4), (12:0), given by Labby (2009). In this remarkable paper, Labby describes a mechanical device he constructed to repeat Weldon's experiment physically and automate the counting of outcomes. He created electronics to roll 12 dice in a physical box, and hooked that up to a webcam to capture an image of each toss and used image processing software to record the counts.

Table 3.5: Number of butterfly species n_k for which k individuals were collected

Individuals (k)	1	2	3	4	5	6	7	8	9	10	11	12	
Species (n_k)	118	74	44	24	29	22	20	19	20	15	12	14	
Individuals (k)	13	14	15	16	17	18	19	20	21	22	23	24	Sum
Species (n_k)	6	12	6	9	9	6	10	10	11	5	3	3	501

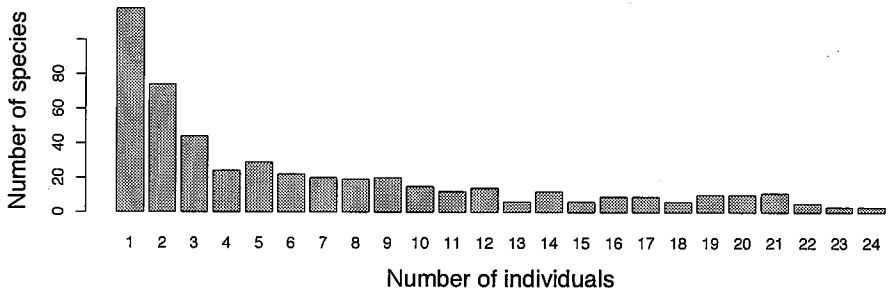


Figure 3.7: Butterfly species in Malaya.

```
> data("Butterfly", package = "vcd")
> barplot(Butterfly, xlab = "Number of individuals", ylab = "Number of species",
+         cex.lab = 1.5)
```

△

3.2 Characteristics of discrete distributions

This section briefly reviews the characteristics of some of the important discrete distributions encountered in practice and illustrates their use with R. An overview of these distributions is shown in Table 3.6. For more detailed information on these and other discrete distributions, Johnson et al. (1992) and Wimmer and Altmann (1999) present the most comprehensive treatments; Zelterman (1999, Chapter 2) gives a compact summary.

For each distribution, we describe properties and generating mechanisms, and show how its parameters can be estimated and how to plot the frequency distribution. R has a wealth of functions

Table 3.6: Discrete probability distributions

Discrete distribution	Probability function, $p(k)$	Parameters
Binomial	$\binom{n}{k} p^k (1-p)^{n-k}$	p = Pr (success); n = # trials
Poisson	$e^{-\lambda}\lambda^k / k!$	λ = mean
Negative binomial	$\binom{n+k-1}{k} p^n (1-p)^k$	p; n = # *successful* trials
Geometric	$p(1-p)^k$	p
Logarithmic series	$\theta^k / [-k \log(1-\theta)]$	θ

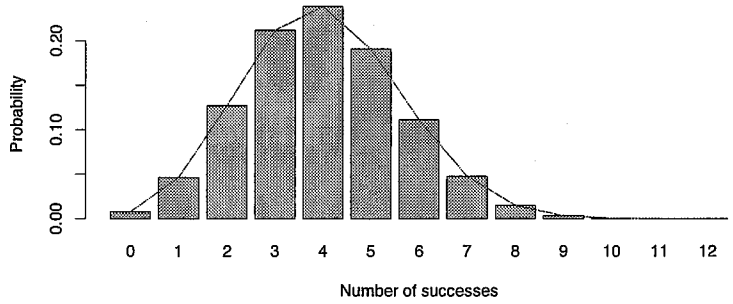


Figure 3.8: Binomial distribution for $k = 0, \ldots, 12$ successes in 12 trials and p=1/3

If we are given data in the form of a discrete (binomial) distribution (and n is known), then the maximum likelihood estimator[6] of p can be obtained as the weighted mean of the values k with weights n_k,

$$\hat{p} = \frac{\bar{x}}{n} = \frac{(\sum_k k \times n_k) / \sum_k n_k}{n} ,$$

and has sampling variance $\mathcal{V}(\hat{p}) = pq/n$.

3.2.1.1 Calculation and visualization

As indicated in Table 3.7 (but without listing the parameters of these functions), binomial probabilities can be calculated with `dbinom(x, n, p)`, where `x` is a vector of the number of successes in `n` trials and `p` is the probability of success on any one trial. Cumulative probabilities, summed to a vector of quantiles, `Q`, can be calculated with `pbinom(Q, n, p)`, and the quantiles (the smallest value x such that $F(x) \geq P$) with `qbinom(P, n, p)`. To generate `N` random observations from a binomial distribution with `n` trials and success probability `p` use `rbinom(N, n, p)`[7].

For example, to find and plot the binomial probabilities corresponding to Weldon's tosses of 12 dice, with $k = 0, \ldots, 12$ and $p = \frac{1}{3}$, we could do the following (giving Figure 3.8):

```
> k <- 0 : 12
> Pk <- dbinom(k, 12, 1/3)
> b <- barplot(Pk, names.arg = k,
+              xlab = "Number of successes", ylab = "Probability")
> lines(x = b, y = Pk, col = "red")
```

We illustrate other styles for plotting in Section 3.2.2, Example 3.11 below.

EXAMPLE 3.8: Weldon's dice

Going a bit further, we can compare Weldon's data with the theoretical binomial distribution as shown below. Because the `WeldonDice` data collapsed the frequencies for 10–12 successes as 10+, we do the same with the binomial probabilities. The expected frequencies (`Exp`), if Weldon's dice tosses obeyed the binomial distribution, are calculated as $N \times p(k)$ for $N = 26,306$ tosses. In addition, we compute the differences of the observed (`Freq`) and expected (`Exp`) frequencies as column `Diff`, to be used for the χ^2 test for goodness of fit described later in Section 3.3, but a glance reveals that these are all negative for $k = 0, \ldots 4$ and positive thereafter.

[6] For the purpose of this book, we assume at least a basic familiarity with the idea of maximum likelihood estimation. A useful brief introduction to this topic for binomial data is Fox (2015, Section D.6), available online.

[7] Note that the actual R function arguments differ from the ones used here.

- If $X \sim \text{Pois}(\lambda)$, then $\sqrt{X}$ converges much faster to a normal distribution $N(\lambda, \frac{1}{4})$, with mean $\sqrt{\lambda}$ and constant variance $\frac{1}{4}$. Hence, the square root transformation is often recommended as a *variance stabilizing* transformation for count data when classical methods (ANOVA, regression) assuming normality are employed.

EXAMPLE 3.9: UK soccer scores

Table 3.8 gives the distributions of goals scored by the 20 teams in the 1995/96 season of the Premier League of the UK Football Association as presented originally by Lee (1997), and now available as the two-way table `UKSoccer` in the vcd package. Over a season each team plays each other team exactly once, so there are a total of $20 \times 19 = 380$ games. Because there may be an advantage for the home team, the goals scored have been classified as "home team" goals and "away team" goals in the table. Of interest for this example is whether the number of goals scored by home teams and away teams follow Poisson distributions, and how this relates to the distribution of the total number of goals scored.

If we assume that in any small interval of time there is a small, constant probability that the home team or the away team may score a goal, the distributions of the goals scored by home teams (the row totals in Table 3.8) may be modeled as $\text{Pois}(\lambda_H)$ and the distribution of the goals scored by away teams (the column totals) may be modeled as $\text{Pois}(\lambda_A)$.

If the number of goals scored by the home and away teams are independent[9], we would expect that the total number of goals scored in any game would be distributed as $\text{Pois}(\lambda_H + \lambda_A)$. These totals are shown in Table 3.9.

As a preliminary check of the distributions for the home and away goals, we can determine if the means and variances are reasonably close to each other. If so, then the total goals variable should also have a mean and variance equal to the sum of those statistics for the home and away goals.

In the R code below, we first convert the two-way frequency table `UKSoccer` to a data frame in

[9]This question is examined visually in Chapter 5 (Example 5.5) and Chapter 6 (Example 6.11), where we find that the answer is "basically, yes."

Table 3.8: Goals scored by home and away teams in 380 games in the Premier Football League, 1995/96 season

Home Team Goals	Away Team Goals 0	1	2	3	4+	Total
0	27	29	10	8	2	76
1	59	53	14	12	4	142
2	28	32	14	12	4	90
3	19	14	7	4	1	45
4+	7	8	10	2	0	27
Total	140	136	55	38	11	380

Table 3.9: Total goals scored in 380 games in the Premier Football League, 1995/95 season

Total goals	0	1	2	3	4	5	6	7
Number of games	27	88	91	73	49	31	18	3

where $\alpha = -1/\log(1-p)$ and $0 < p < 1$. For this distribution, the first two central moments are:

$$\begin{aligned} \text{Mean}(X) &= \alpha\left(\frac{p}{1-p}\right) \\ \text{Var}(X) &= -p\frac{p+\log(1-p)}{(1-p)^2\log^2(1-p)} \end{aligned}$$

Fisher derived the logarithmic series distribution by assuming that for a given species the number of individuals trapped has a Poisson distribution with parameter $\lambda = \gamma t$, where γ is a parameter of the species (susceptibility to entrapment) and t is a parameter of the trap. If different species vary so that the parameter γ has a gamma distribution, then the number of representatives of each species trapped will have a negative binomial distribution. However, the observed distribution is necessarily truncated on the left, because one cannot observe the number of species never caught (where $k = 0$). The logarithmic series distribution thus arises as a limiting form of the zero-truncated negative binomial.

Maximum likelihood estimation of the parameter p in the log-series distribution is described by Böhning (1983), extending a simpler Newton's method approximation by Birch (1963a). The vcdExtra package contains the set of R functions, `dlogseries(x, prob)`, `plogseries(q, prob)`, `qlogseries(p, prob)`, and `rlogseries(n, prob)`, where `prob` represents p here.

3.2.6 Power series family

We mentioned earlier that the Poisson distribution was unique among all discrete (one-parameter) distributions, in that it is the only one whose mean and variance are equal (Kosambi, 1949). The relation between mean and variance of discrete distributions also provides the basis for integrating them into a general family. All of the discrete distributions described in this section are in fact special cases of a family of discrete distributions called the power series distributions by Noack (1950) and defined by

$$p(k) = a(k)\theta^k / f(\theta) \qquad k = 0, 1, \ldots ,$$

with parameter $\theta > 0$, where $a(k)$ is a coefficient function depending only on k and $f(\theta) = \sum_k a(k)\theta^k$ is called the series function. The definitions of these functions are shown in Table 3.10.

These relations among the discrete distribution provide the basis for graphical techniques for diagnosing the form of discrete data described later in this chapter (Section 3.5.4).

Table 3.10: The Power Series family of discrete distributions

Discrete Distributiion	Probability function, $p(k)$	Series parameter, θ	Series function, $f(\theta)$	Series coefficient, $a(k)$
Poisson	$e^{-\lambda}\lambda^k/k!$	$\theta = \lambda$	e^{θ}	$1/k!$
Binomial	$\binom{n}{k}p^k(1-p)^{n-k}$	$\theta = p/(1-p)$	$(1+\theta)^n$	$\binom{n}{k}$
Negative binomial	$\binom{n+k-1}{k}p^n(1-p)^k$	$\theta = (1-p)$	$(1-\theta)^{-k}$	$\binom{n+k-1}{k}$
Geometric	$p(1-p)^k$	$\theta = (1-p)$	$(1-\theta)^{-k}$	1
Logarithmic series	$\theta^k/[-k\log(1-\theta)]$	$\theta = \theta$	$-\log(1-\theta)$	$1/k$

```
> summary(Fed_fit0)

 Goodness-of-fit test for poisson distribution

                     X^2 df   P(> X^2)
Likelihood Ratio 25.243  5 0.00012505
```

Mosteller and Wallace (1963) determined that the negative binomial distribution provided a better fit to these data than the Poisson. We can verify this as follows:

```
> Fed_fit1 <- goodfit(Federalist, type = "nbinomial")
> unlist(Fed_fit1$par)

   size    prob
1.18633 0.64376

> summary(Fed_fit1)

 Goodness-of-fit test for nbinomial distribution

                    X^2 df P(> X^2)
Likelihood Ratio 1.964  4  0.74238
```

Recall that the Poisson distribution assumes that the probability of a word like *may* appearing in a block of text is small and constant, and that for the Poisson, $\mathcal{E}(x) = \mathcal{V}(x) = \lambda$. One interpretation of the better fit of the negative binomial is that the use of a given word occurs with Poisson frequencies, but Madison varied its rate λ_i from one block of text to another according to a gamma distribution, allowing the variance to be greater than the mean.

△

3.3.2 Plots of observed and fitted frequencies

In the examples of the last section, we saw cases where the GOF tests showed close agreement between the observed and model-fitted frequencies, and cases where they diverged significantly, to cause rejection of a hypothesis that the data followed the specified distribution.

What is missing from such numerical summaries is any appreciation of the *details* of this statistical comparison. Plots of the observed and fitted frequencies can help to show both the shape of the theoretical distribution we have fitted and the pattern of any deviations between our data and theory.

In this section we illustrate some simple plotting tools for these purposes, using the `plot.goodfit()` method for "goodfit" objects.[14] The left panel of Figure 3.14 shows the fit of the Poisson distribution to the Federalist Papers data, using one common form of plot that is sometimes used for this purpose. In this plot, observed frequencies are shown by bars and fitted frequencies are shown by points, connected by a smooth (spline) curve.

Such a plot, however, is dominated by the largest frequencies, making it hard to assess the deviations among the smaller frequencies. To make the smaller frequencies more visible, Tukey (1977) suggests plotting the frequencies on a square-root scale, which he calls a ***rootogram***. This plot is shown in the right panel of Figure 3.14.

[14] Quantile–quantile (QQ) plots are a common alternative for the goal of comparing observed and expected values under some distribution. These plots are useful for unstructured samples, but less so when we also want to see the shape of a distribution, as is the case here.

3.5.3 The `distplot()` function

Poissonness plots (and versions for other distributions) are produced by the function `distplot()` in vcd. As with `Ord_plot()`, the first argument is either a vector of counts, a one-way table of frequencies of counts, or a data frame or matrix with frequencies in the first column and the corresponding counts in the second column. Nearly all of the examples in this chapter use one-way tables of counts.

The `type` argument specifies the type of distribution. For `type = "poisson"`, specifying a value for `lambda` $= \lambda_0$ gives the leveled version of the plot.

EXAMPLE 3.20: Death by horse kick

The calculations for the Poissonness plot, including confidence intervals, are shown below for the *HorseKicks* data. The call to `distplot()` produces the plot in the left panel of Figure 3.21.

```
> data("HorseKicks", package = "vcd")
> dp <- distplot(HorseKicks, type = "poisson",
+   xlab = "Number of deaths", main = "Poissonness plot: HorseKicks data")
> print(dp, digits = 4)

  Counts Freq Metameter CI.center CI.width CI.lower CI.upper
1      0  109    -0.607   -0.6131   0.1305  -0.7436  -0.4827
2      1   65    -1.124   -1.1343   0.2069  -1.3412  -0.9274
3      2   22    -1.514   -1.5451   0.4169  -1.9620  -1.1281
4      3    3    -2.408   -2.6607   1.3176  -3.9783  -1.3431
5      4    1    -2.120   -3.1203   2.6887  -5.8089  -0.4316
```

In this plot, the open circles show the calculated observed values of the count `Metameter` $= \phi(n_k)$. The smaller filled points show the centers of the confidence intervals, `CI.center` $= \phi(n_k^*)$ (Eqn. (3.15)), and the dashed lines show the extent of the confidence intervals.

The fitted least squares line has a slope of -0.431, which would indicate $\lambda = e^{-0.431} = 0.65$. This compares well with the MLE, $\lambda = \bar{x} = 0.61$.

Using `lambda = 0.61` as below gives the leveled version shown in the right panel of Figure 3.21.

```
> # leveled version, specifying lambda
> distplot(HorseKicks, type = "poisson", lambda = 0.61,
+   xlab = "Number of deaths", main = "Leveled Poissonness plot")
```

In both plots the fitted line is within the confidence intervals, indicating the adequacy of the Poisson model for these data. The widths of the intervals for $k > 2$ are graphic reminders that these observations have decreasingly low precision where the counts n_k are small.

△

3.5.4 Plots for other distributions

As described in Section 3.2.6, the binomial, Poisson, negative binomial, geometric, and logseries distributions are all members of the general power series family of discrete distributions. For this family, Hoaglin and Tukey (1985) developed similar plots of a count metameter against k, which appear as a straight line when a data distribution follows a given family member.

The distributions which can be analyzed in this way are shown in Table 3.12, with the interpretation given to the slope and intercept in each case. For example, for the Binomial distribution, a "binomialness" plot is constructed by plotting $\log n_k^*/N\binom{n}{k}$ against k. If the points in this plot approximate a straight line, the slope is interpreted as $\log(p/(1-p))$, so the binomial parameter p may be estimated as $p = e^b/(1+e^b)$.

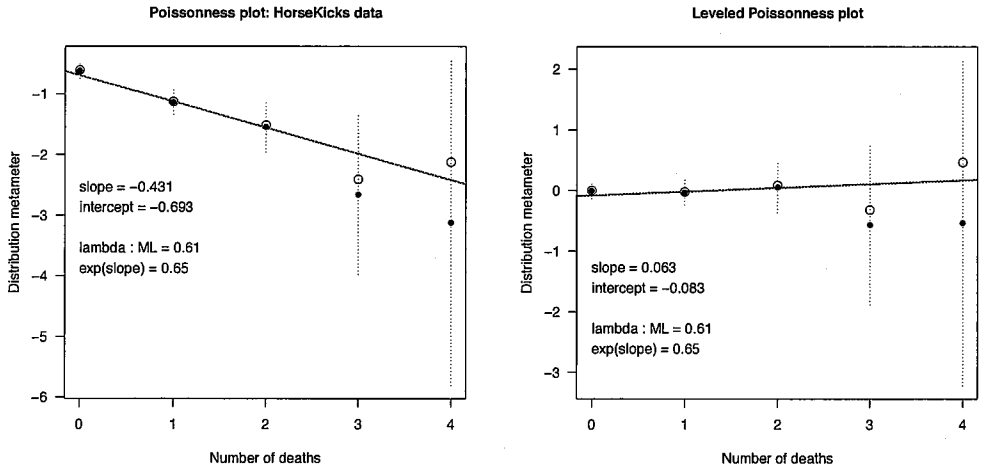


Figure 3.21: Poissonness plots for the HorseKick data. Left: standard plot; right: leveled plot.

Unlike the Ord plot, a different plot is required for each distribution, because the count metameter, $\phi(n_k)$, differs from distribution to distribution. Moreover, systematic deviation from a linear relationship does not indicate which distribution provides a better fit. However, the attention to robustness, and the availability of confidence intervals and influence diagnostics, make this a highly useful tool for visualizing discrete distributions.

EXAMPLE 3.21: Families in Saxony

Our analysis in Example 3.2 and Example 3.13 of the `Saxony` data showed that the distribution of male children had slightly heavier tails than the binomial, meaning the observed distribution is overdispersed. We can see this in the `goodfit()` plot shown in Figure 3.22 (left), and even more clearly in the distribution diagnostic plot produced by `distplot()` in the right panel of Figure 3.22. For a binomial distribution, we call this distribution plot a "binomialness plot."

Table 3.12: Plot parameters for five discrete distributions. In each case the count metameter, $\phi(n_k^*)$ is plotted against k, yielding a straight line when the data follow the given distribution.

Distribution	Probability function, $p(k)$	Count) metameter, $\phi(n_k^*)$	Theoretical slope (b)	Theoretical intercept (a)
Poisson	$e^{-\lambda}\lambda^k/k!$	$\log(k!n_k^*/N)$	$\log(\lambda)$	$-\lambda$
Binomial	$\binom{n}{k}p^k(1-p)^{n-k}$	$\log\left(n_k^*/N\binom{n}{k}\right)$	$\log\left(\frac{p}{1-p}\right)$	$n\log(1-p)$
Negative binomial	$\binom{n+k-1}{k}p^n(1-p)^k$	$\log\left(n_k^*/N\binom{n+k-1}{k}\right)$	$\log(1-p)$	$n\log(p)$
Geometric	$p(1-p)^k$	$\log\left(n_k^*/N\right)$	$\log(1-p)$	$\log(p)$
Log series	$\theta^k/[-k\log(1-\theta)]$	$\log\left(kn_k^*/N\right)$	$\log(\theta)$	$-\log\left(-\log(1-\theta)\right)$

Source: adapted from Hoaglin and Tukey (1985), Table 9-15.

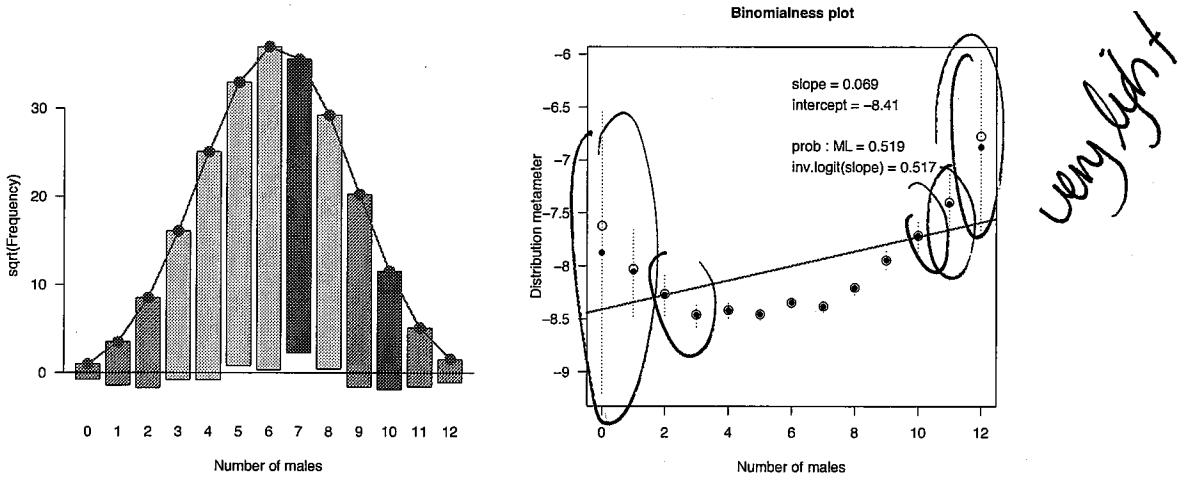


Figure 3.22: Diagnostic plots for males in Saxony families. Left: `goodfit()` plot; right: `distplot()` plot. Both plots show heavier tails than in a binomial distribution.

```
> plot(goodfit(Saxony, type = "binomial", par = list(size=12)),
+      shade=TRUE, legend=FALSE,
+      xlab = "Number of males")
> distplot(Saxony, type = "binomial", size = 12,
+   xlab = "Number of males")
```

The weight of evidence is thus that, as simple as the binomial might be, it is inadequate to fully explain the distribution of sex ratios in this large sample of families of 12 children. To understand this data better, it is necessary to question the assumptions of the binomial (births of males are independent Bernoulli trials with constant probability p) as a model for this birth distribution and/or find a more adequate model.[17] △

EXAMPLE 3.22: Federalist Papers

In Example 3.16 we carried out GOF tests for the Poisson and negative binomial models with the Federalist Papers data; Figure 3.16 showed the corresponding rootogram plots. Figure 3.23 compares these two using the diagnostic plots of this section. Again the Poisson shows systematic departure from the linear relation required in the Poissonness plot, while the negative binomial model provides an acceptable fit to these data.

```
> distplot(Federalist, type = "poisson", xlab = "Occurrences of 'may'")
> distplot(Federalist, type = "nbinomial", xlab = "Occurrences of 'may'")
```

△

[17] On these questions, Edwards (1958) reviews numerous other studies of these Geissler's data, and fits a so-called ***β-binomial*** model proposed by Skellam (1948), where p varies among families according to a β distribution. He concludes that there is evidence that p varies between families of the same size. One suggested explanation is that family decisions to have a further child is influenced by the balance of boys and girls among their earlier children.

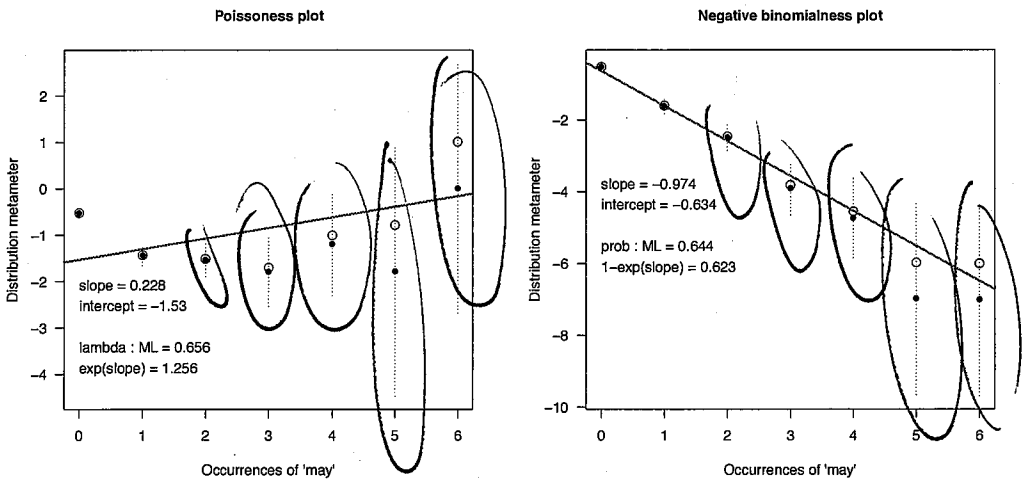


Figure 3.23: Diagnostic plots for the Federalist Papers data. Left: Poissonness plot; right: negative binomialness plot.

3.6 Fitting discrete distributions as generalized linear models*

In Section 3.2.6, we described how the common discrete distributions are all members of the general power series family. This provides the basis for the generalized distribution plots described in Section 3.5.4. Another general family of distributions—the ***exponential family***—includes most of the common continuous distributions: the normal, gamma, exponential, and others, and is the basis of the class of generalized linear models (GLMs) fit by `glm()`.

A clever approach by Lindsey and Mersch (1992), Lindsey (1995, Section 6.1) shows how various discrete (and continuous) distributions can be fit to frequency data using generalized linear models for log frequency (which are equivalent to Poisson loglinear models). The uniform, geometric, binomial, and the Poisson distributions may all be fit easily in this way, but the idea extends to some other distributions, such as the ***double binomial*** distribution, that allows a separate parameter for overdispersion relative to the binomial. A clear advantage is that this method gives estimated standard errors for the distribution parameters as well as estimated confidence intervals for fitted probabilities.

The essential idea is that, for frequency data, any distribution in the exponential family may be represented by a linear model for the logarithm of the cell frequency, with a Poisson distribution for errors, otherwise known as a "Poisson loglinear regression model." These have the form

$$\log(N\pi_k) = \text{ offset } + \beta_0 + \boldsymbol{\beta}^{\mathsf{T}}\boldsymbol{S}(k)\,,$$

where N is the total frequency, π_k is the modeled probability of count k, $\boldsymbol{S}(k)$ is a vector of zero or more sufficient statistics for the canonical parameters of the exponential family distribution, and the offset term is a value that does not depend on the parameters.

Table 3.13 shows the sufficient statistics and offsets for several discrete distributions. See Lindsey and Mersch (1992) for further details, and definitions for the double-binomial distribution,[18]

[18] In R, the double binomial distribution is implemented in the rmutil package, providing the standard complement of density function (`ddoublebinom()`), CDF (`pdoublebinom()`), quantiles (`qdoublebinom()`), and random genera-

Table 3.13: Poisson loglinear representations for some discrete distributions

Distribution	Sufficient statistics	Offset
Geometric	k	
Poisson	k	$-\log(k!)$
Binomial	k	$\log \binom{n}{k}$
Double binomial	$k, k\log(k) + (n-k)\log(n-k)$	$\log \binom{n}{k}$

and Lindsey (1995, pp. 130–133) for his analysis of the `Saxony` data using this distribution. Lindsey and Altham (1998) provide an analysis of the complete Geissler data (provided in the data set `Geissler` in vcdExtra) using several different models to handle overdispersion.

EXAMPLE 3.23: Families in Saxony

The binomial distribution and the double binomial can both be fit to frequency data as a Poisson regression via `glm()` using $\log \binom{n}{k}$ as an offset. First, we convert `Saxony` into a numeric data frame for use with `glm()`.

```
> data("Saxony", package = "vcd")
> Males <- as.numeric(names(Saxony))
> Families <- as.vector(Saxony)
> Sax.df <- data.frame(Males, Families)
```

To calculate the offset for `glm()` in R, note that `choose(12,0:12)` returns the binomial coefficients, and `lchoose(12,0:12)` returns their logs.

```
> # fit binomial (12, p) as a glm
> Sax.bin <- glm(Families ~ Males, offset = lchoose(12, 0:12),
+                family = poisson, data = Sax.df)
>
> # brief model summaries
> LRstats(Sax.bin)

Likelihood summary table:
        AIC BIC LR Chisq Df Pr(>Chisq)
Sax.bin 191 192       97 11      7e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

> coef(Sax.bin)

(Intercept)       Males
  -0.069522    0.076898
```

As we have seen, this model fits badly. The parameter estimate for `Males`, $\beta_1 = 0.0769$ is actually estimating the logit of p, $\log p/(1-p)$, so the inverse transformation gives $\hat{p} = \frac{\exp(\beta_1)}{1+\exp(\beta_1)} = 0.5192$, as we had before.

The double binomial model can be fitted as follows. The term `YlogitY` calculates $k\log(k) + (n-k)\log(n-k)$, the second sufficient statistic for the double binomial (see Table 3.13) fitted via `glm()`.

tion (`rdoublebinom()`). This package is not on CRAN, but is available at `http://www.commanster.eu/rcode.html`.

```
          AwayGoals
HomeGoals  0  1  2  3  4  5  6
        0 26 16 13  5  0  1  0
        1 19 58 20  5  4  0  1
        2 27 23 20  5  1  1  1
        3 14 11 10  4  2  0  0
        4  3  5  3  0  0  0  0
        5  4  1  0  1  0  0  0
        6  1  0  0  1  0  0  0
```

(a) As in Example 3.9, find the one-way distributions of `HomeGoals`, `AwayGoals`, and `TotalGoals = HomeGoals + AwayGoals`.
(b) Use `goodfit()` to fit and plot the Poisson distribution to each of these. Does the Poisson seem to provide a reasonable fit?
(c) Use `distplot()` to assess fit of the Poisson distribution.
(d) What circumstances of scoring goals in soccer might cause these distributions to deviate from Poisson distributions?

Exercise 3.9 * Repeat the exercise above, this time using the data for all years in which there was the standard number (306) of games, that is for `Year>1965`, tabulated as shown below.

```
> BL <- xtabs(~ HomeGoals + AwayGoals, data = Bundesliga,
+             subset = (Year > 1965))
```

Exercise 3.10 Using the data *CyclingDeaths* introduced in Example 3.6 and the one-way frequency table `CyclingDeaths.tab = table(CyclingDeaths$deaths)`,

(a) Make a sensible plot of the number of deaths over time. For extra credit, add a smoothed curve (e.g., using `lines(lowess(...))`).
(b) Test the goodness of fit of the table `CyclingDeaths.tab` to a Poisson distribution statistically using `goodfit()`.
(c) Continue this analysis using a `rootogram()` and `distplot()`.
(d) Write a one-paragraph summary of the results of these analyses and your conclusions.

Exercise 3.11 * The one-way table, *Depends*, in vcdExtra and shown below gives the frequency distribution of the number of dependencies declared in 4,983 R packages maintained on the CRAN distribution network on January 17, 2014. That is, there were 986 packages that had no dependencies, 1,347 packages that depended on one other package, ... up to 2 packages that depended on 14 other packages.

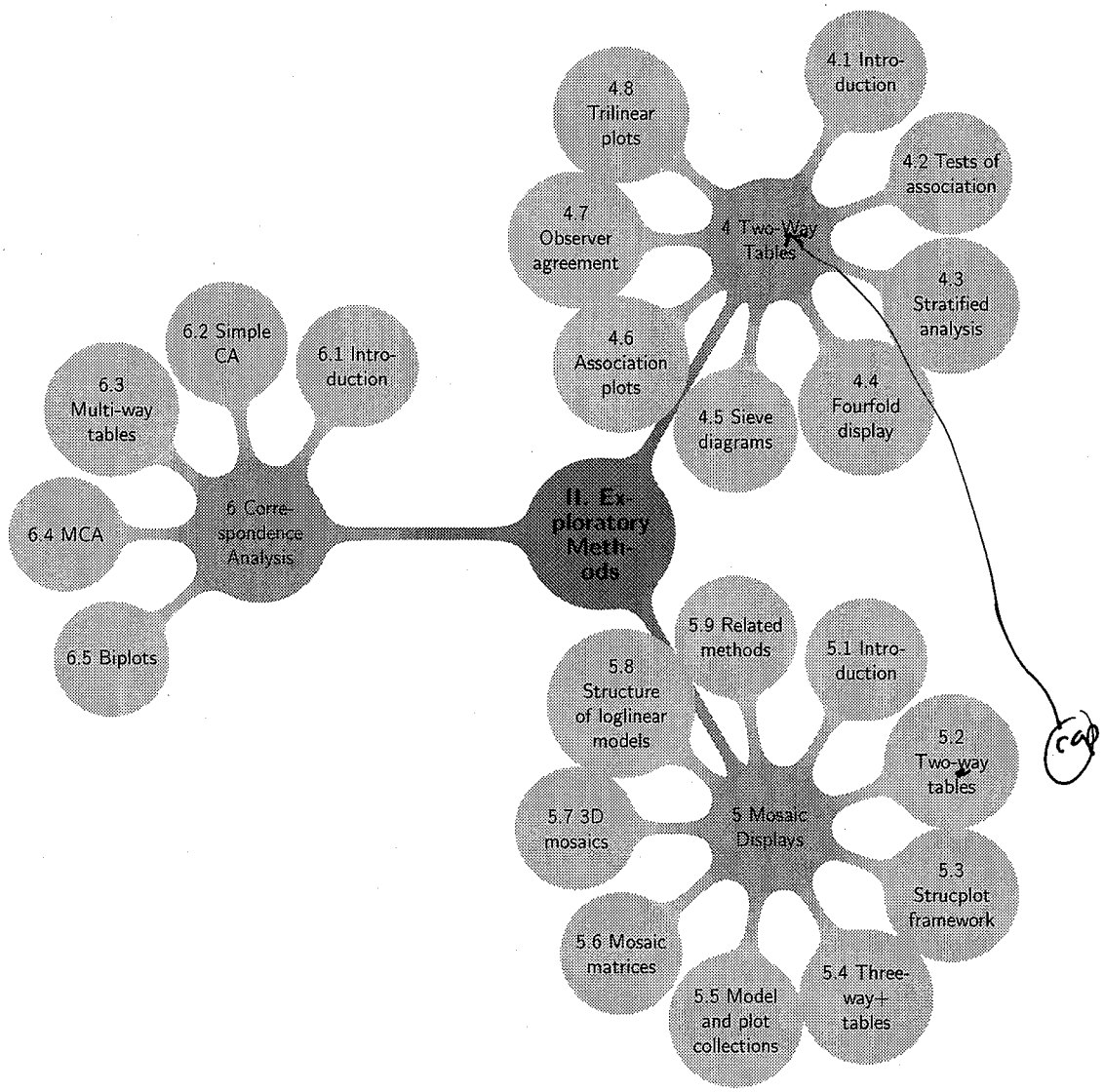
Depends	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
# Pkgs	986	1,347	993	685	375	298	155	65	32	19	9	4	9	4	2

(a) Make a bar plot of this distribution.
(b) Use `Ord_plot()` to see if this method can diagnose the form of the distribution.
(c) Try to fit a reasonable distribution to describe dependencies among R packages.

Exercise 3.12 * How many years does it take to get into the baseball Hall of Fame? The Lahman (Friendly, 2014b) package provides a complete record of historical baseball statistics from 1871 to the present. One table, *HallOfFame*, records the history of players nominated to the Baseball Hall of Fame, and those eventually inducted. The table below, calculated in `help(HallOfFame, package="Lahman")`, records the distribution of the number of years taken (from first nomination) for the 109 players in the Hall of Fame to be inducted (1936–present). Note that `years==0`

Part II

Exploratory and Hypothesis-testing Methods



"what is the eye color *given* the hair color?," "how does eye color influence hair color?"), even though such an interpretation may be inappropriate (as in this example);

- labeling may become increasingly complex.

A somewhat better approach is a ***tile plot*** (using `tile()` in vcd), as shown next to the bar plot in Figure 4.1:

```
> tile(hec)
```

The table frequencies are represented by the area of rectangles arranged in the same tabular form as the raw data, facilitating comparisons between tiles accross both variables (by rows or by columns), by maintaining a one-to-one relationship to the underlying table[2].

Everyday observation suggests that there probably is an association between hair color and eye color, and we will describe tests and measures of associations for larger tables in Section 4.2.3. If, as is suspected, hair color and eye color are associated, we would like to understand *how* they are associated. The graphical methods described later in this chapter and in Chapter 5 help reveal the pattern of associations present. △

EXAMPLE 4.3: Mental impairment and parents' SES

Srole et al. (1978, p. 289) gave the data in Table 4.3 on the mental health status of a sample of 1660 young New York residents in midtown Manhattan classified by their parents' socioeconomic status (SES); see `Mental` in the vcdExtra package. These data have also been analyzed by many authors, including Agresti (2013, Section 10.5.3), Goodman (1979), and Haberman (1979, p. 375).

There are six categories of SES (from 1 = "High" to 6 = "Low"), and mental health is classified in the four categories "well," "mild symptom formation," "moderate symptom formation," and "impaired." It may be useful here to consider SES as explanatory and ask whether and how it predicts mental health status as a response, that is, whether there is an association, and if so, investigate its nature.

```
> data("Mental", package = "vcdExtra")
> mental <- xtabs(Freq ~ ses + mental, data = Mental)
> spineplot(mental)
```

Figure 4.2 shows a ***spineplot*** of this data—basically a stacked barchart of the row percentages of mental impairment for each SES category, the width of each bar being proportional to the overall SES percentages.[3] From this graph, it is apparant that the "well" mental state decreases with social-economic status, while the "impaired" state increases. This pattern is more specific than overall

[2]This kind of display is more generally known as a ***fluctuation diagram*** (Hofmann, 2000), flexibly implemented by function `fluctile()` in the package extracat (Pilhoefer, 2014).

[3]Thus, in the more technical terms introduced in 4.2.1, this spineplot shows the conditional distribution of impairment, given the categories of SES.

Table 4.3: Mental impairment and parents' SES

	Mental impairment			
SES	Well	Mild	Moderate	Impaired
1	64	94	58	46
2	57	94	54	40
3	57	105	65	60
4	72	141	77	94
5	36	97	54	78
6	21	71	54	71

```
> confint(OR)

                                2.5 % 97.5 %
Admitted:Rejected/Male:Female 1.6244 2.0867

> confint(LOR)

                                2.5 %  97.5 %
Admitted:Rejected/Male:Female 0.48512 0.73558
```

Finally, we note that an exact test (based on the hypergeometric distribution) is provided by `fisher.test()` (see the help page for the details):

```
> fisher.test(UCB)

Fisher's Exact Test for Count Data

data:  UCB
p-value <2e-16
alternative hypothesis: true odds ratio is not equal to 1
95 percent confidence interval:
 1.6214 2.0912
sample estimates:
odds ratio
    1.8409
```

In general, exact tests are to be prefered over asymptotic tests like the one described above. Note, however, that the results are very similar in this example. △

4.2.3 Larger tables: Overall analysis

For two-way tables, overall tests of association can be carried out using `assocstats()`. If the data set has more than two factors (as in the *Arthritis* data), the other factors will be ignored (and collapsed) if not included when the table is constructed. This simplified analysis may be misleading if the excluded factors interact with the factors used in the analysis.

EXAMPLE 4.6: Arthritis treatment

Since the main interest is in the relation between `Treatment` and `Improved`, an overall analysis (that ignores `Sex`) can be carried out by creating a two-way table with `xtabs()` as shown below.

```
> data("Arthritis", package = "vcd")
> Art <- xtabs(~ Treatment + Improved, data = Arthritis)
> Art

         Improved
Treatment None Some Marked
  Placebo   29    7      7
  Treated   13    7     21

> round(100 * prop.table(Art, margin = 1), 2)

         Improved
Treatment  None  Some Marked
  Placebo 67.44 16.28  16.28
  Treated 31.71 17.07  51.22
```

△

EXAMPLE 4.10: Arthritis treatment

For the arthritis data, homogeneity means the association between treatment and outcome (`improve`) is the same for both men and women. Again, we are using `woolf_test()` to test if this assumption holds.

```
> woolf_test(Art2)

Woolf-test on Homogeneity of Odds Ratios (no 3-Way
assoc.)

data:  Art2
X-squared = 0.318, df = 1, p-value = 0.57
```

Even though we found in the CMH analysis above that the association between `Treatment` and `Improved` was stronger for females than males, the analysis using `woolf_test()` is clearly non-significant, so we cannot reject homogeneity of association. △

Remark

As will be discussed later (Section 5.4) in the case of a 3-way table, the hypothesis of homogeneity of association among three variables A, B and C can be stated as the *loglinear model* of no three-way association, [AB][AC][BC]. This notation (described in Section 5.4.1 and Section 9.2) lists only the high-order association terms in a linear model for log frequency.

This hypothesis can be stated as the loglinear model,

```
[SexTreatment] [SexImproved] [TreatmentImproved].
```
(4.4)

Such tests can be carried out most conveniently using `loglm()` in the MASS package. The model formula uses the standard R notation `()^2` to specify all terms of order 2.

```
> library(MASS)
> loglm(~ (Treatment + Improved + Sex)^2, data = Art2)

Call:
loglm(formula = ~(Treatment + Improved + Sex)^2, data = Art2)

Statistics:
                     X^2 df P(> X^2)
Likelihood Ratio 1.7037  2  0.42663
Pearson          1.1336  2  0.56735
```

Consistent with the Woolf test, the interaction terms are not significant.

4.4 Fourfold display for 2 x 2 tables

The ***fourfold display*** is a special case of a ***radial diagram*** (or "polar area chart") designed for the display of 2×2 (or $2\times2\times k$) tables (Fienberg, 1975, Friendly, 1994a,c). In this display the frequency n_{ij} in each cell of a fourfold table is shown by a quarter circle, whose radius is proportional to $\sqrt{n_{ij}}$, so the area is proportional to the cell count. The fourfold display is similar to a pie chart in using segments of a circle to show frequencies. It differs from a pie chart in that it keeps the angles of the segments constant and varies the radius, whereas the pie chart varies the angles and keeps the radius constant.

and tests of agreement provide a method of assessing the reliability of a subjective classification or assessment procedure.

For example, two (or more) clinical psychologists might classify patients on a scale with categories (a) normal, (b) mildly impaired, (c) severely impaired. Or, ethologists might classify the behavior of animals in categories of cooperation, dominance and so forth, or paleologists might classify pottery fragments according to categories of antiquity or cultural groups. As these examples suggest, the rating categories are often ordered, but not always.

For two raters, a contingency table can be formed by classifying all the subjects/objects rated according to the rating categories used by the two observers. In most cases, the same categories are used by both raters, so the contingency table is square, and the entries in the diagonal cells are the cases where the raters agree.

In this section we describe some measures of the strength of agreement and then a method for visualizing the pattern of agreement. But first, the following examples show some typical agreement data.

EXAMPLE 4.16: Sex is fun

The *SexualFun* table in vcd (Agresti (1990, Table 2.10), from Hout et al. (1987)) summarizes the responses of 91 married couples to a questionnaire item: "Sex is fun for me and my partner: (a) Never or occasionally, (b) Fairly often, (c) Very often, (d) Almost always."

```
> data("SexualFun", package = "vcd")
> SexualFun

             Wife
Husband       Never Fun Fairly Often Very Often Always fun
  Never Fun           7            7          2           3
  Fairly Often        2            8          3           7
  Very Often          1            5          4           9
  Always fun          2            8          9          14
```

In each row the diagonal entry is not always the largest, though it appears that the partners tend to agree more often when either responds "Almost always." △

EXAMPLE 4.17: Diagnosis of MS patients

Landis and Koch (1977) gave data on the diagnostic classification of multiple sclerosis (MS) patients by two neurologists, one from Winnipeg and one from New Orleans. There were two samples of patients, 149 from Winnipeg and 69 from New Orleans, and each neurologist classified all patients into one of four diagnostic categories: (a) Certain MS, (b) Probable MS, (c) Possible MS, (d) Doubtful, unlikely, or definitely not MS.

These data are available in *MSPatients*, a 4 × 4 × 2 table, as shown below. It is convenient to show the data in separate slices for the Winnipeg and New Orleans patients:

```
> MSPatients[, , "Winnipeg"]

                     Winnipeg Neurologist
New Orleans Neurologist Certain Probable Possible Doubtful
            Certain         38        5        0        1
            Probable        33       11        3        0
            Possible        10       14        5        6
            Doubtful         3        7        3       10

> MSPatients[, , "New Orleans"]

                     Winnipeg Neurologist
New Orleans Neurologist Certain Probable Possible Doubtful
            Certain         5        3        0        0
```

Hence, it is common to conduct a test of $H_0 : \kappa = 0$ by referring $z = \kappa / \hat{\sigma}(\kappa)$ to a unit normal distribution. The hypothesis of agreement no better than chance is rarely of much interest, however. It is preferable to estimate and report a confidence interval for κ.

4.7.1.3 Weighted Kappa

The original (unweighted) κ only counts strict agreement (the same category is assigned by both observers). A weighted version of κ (Cohen, 1968) may be used when one wishes to allow for *partial* agreement. For example, exact agreements might be given full weight, while a one-category difference might be given a weight of 1/2. This typically makes sense only when the categories are *ordered*, as in severity of diagnosis.

Weighted κ uses weights, $0 \leq w_{ij} \leq 1$ for each cell in the table, with $w_{ii} = 1$ for the diagonal cells. In this case P_o and P_c are defined as weighted sums

$$
\begin{aligned}
P_o &= \sum_i \sum_j w_{ij} p_{ij} \\
P_c &= \sum_i \sum_j w_{ij} p_{i+} p_{+j}
\end{aligned}
$$

and these weighted sums are used in Eqn. (4.6).

For an $R \times R$ table, two commonly used patterns of weights are those based on equal spacing of weights (Cicchetti and Allison, 1971) for a near-match, and *Fleiss-Cohen weights* (Fleiss and Cohen, 1972), based on an inverse-square spacing,

$$
\begin{aligned}
w_{ij} &= 1 - \frac{|i-j|}{R-1} && \text{equal spacing} \\
w_{ij} &= 1 - \frac{|i-j|^2}{(R-1)^2} && \text{Fleiss-Cohen}
\end{aligned}
$$

The Fleiss-Cohen weights attach greater importance to near disagreements, as you can see below for a 4×4 table. These weights also provide a measure equivalent to the intraclass correlation.

```
      Integer Spacing                    Inverse Square Spacing
  Cicchetti Allison weights               Fleiss-Cohen weights
-----------------------------      ----------------------------
  1     2/3     1/3       0          1     8/9     5/9       0
2/3       1     2/3     1/3        8/9       1     8/9     5/9
1/3     2/3       1     2/3        5/9     8/9       1     8/9
  0     1/3     2/3       1          0     5/9     8/9       1
```

4.7.1.4 Computing Kappa

The function `Kappa()` in vcd calculates unweighted and weighted Kappa. The `weights` argument can be used to specify the weighting scheme as either `"Equal-Spacing"` or `"Fleiss-Cohen"`. The function returns a "Kappa" object, for which there is a `confint.Kappa()` method, providing confidence intervals. The `summary.Kappa()` method also prints the weights.

The lines below illustrate `Kappa` for the *SexualFun* data.

```
> Kappa(SexualFun)

             value    ASE    z Pr(>|z|)
Unweighted 0.129 0.0686 1.89  0.05939
Weighted   0.237 0.0783 3.03  0.00244
```

```
> confint(Kappa(SexualFun))

Kappa                lwr     upr
  Unweighted -0.0051204 0.26378
  Weighted    0.0838834 0.39088
```

4.7.2 Observer Agreement Chart

The observer agreement chart proposed by Bangdiwala (1985, 1987) provides a simple graphic representation of the strength of agreement in a contingency table, and alternative measures of strength of agreement with an intuitive interpretation. More importantly, it shows the *pattern* of disagreement when agreement is less than perfect.

4.7.2.1 Construction of the basic plot

Given a $k \times k$ contingency table, the agreement chart is constructed as an $n \times n$ square, where $n = n_{++}$ is the total sample size. Black squares, each of size $n_{ii} \times n_{ii}$, show observed agreement. These are positioned within k larger rectangles, each of size $n_{i+} \times n_{+i}$ as shown in the left panel of Figure 4.18. Each rectangle is subdivided by the row/column frequencies n_{ij} of row i/column j, where cell (i, i) is filled black. The large rectangle shows the maximum possible agreement, given the marginal totals. Thus, a visual impression of the strength of agreement is given by

$$B = \frac{\text{area of dark squares}}{\text{area of rectangles}} = \frac{\sum_i^k n_{ii}^2}{\sum_i^k n_{i+}\, n_{+i}} \,. \tag{4.7}$$

When there is perfect agreement, the k rectangles determined by the marginal totals are all squares, completely filled by the shaded squares reflecting the diagonal n_{ii} entries, and $B = 1$.

```
> agreementplot(SexualFun, main = "Unweighted", weights = 1)
> agreementplot(SexualFun, main = "Weighted")
```

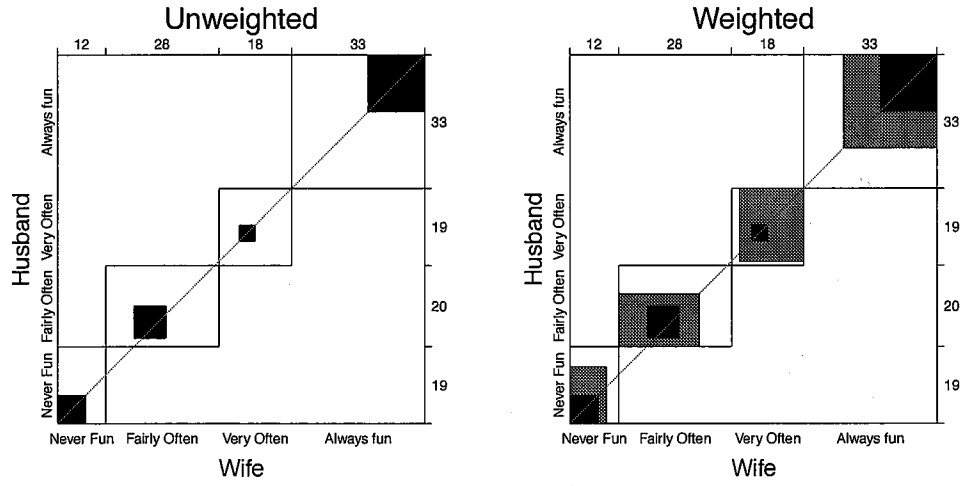


Figure 4.18: Agreement charts for husbands' and wives' sexual fun. Left: unweighted chart, showing only exact agreement; right: weighted chart, using weight $w_1 = 8/9$ for a one-step disagreement.

(a) Try one or more of the following other functions for visualizing two-way contingency tables with this data: `plot()`, `tile()`, `mosaic()`, and `spineplot()`. [For all except `spineplot()`, it is useful to include the argument `shade=TRUE`].
(b) Comment on the differences among these displays for understanding the relation between visits and length of stay.

Exercise 4.6 The two-way table *Mammograms* in vcdExtra gives ratings on the severity of diagnosis of 110 mammograms by two raters.

(a) Assess the strength of agreement between the raters using Cohen's κ, both unweighted and weighted.
(b) Use `agreementplot()` for a graphical display of agreement here.
(c) Compare the Kappa measures with the results from `assocstats()`. What is a reasonable interpretation of each of these measures?

Exercise 4.7 Agresti and Winner (1997) gave the data in Table 4.8 on the ratings of 160 movies by the reviewers Gene Siskel and Roger Ebert for the period from April 1995 through September 1996. The rating categories were Con ("thumbs down"), Mixed, and Pro ("thumbs up").

Table 4.8: Movie ratings by Siskel & Ebert, April 1995–September 1996. *Source*: Agresti and Winner (1997)

		Ebert			
		Con	Mixed	Pro	Total
Siskel	Con	24	8	13	45
	Mixed	8	13	11	32
	Pro	10	9	64	83
	Total	42	30	88	160

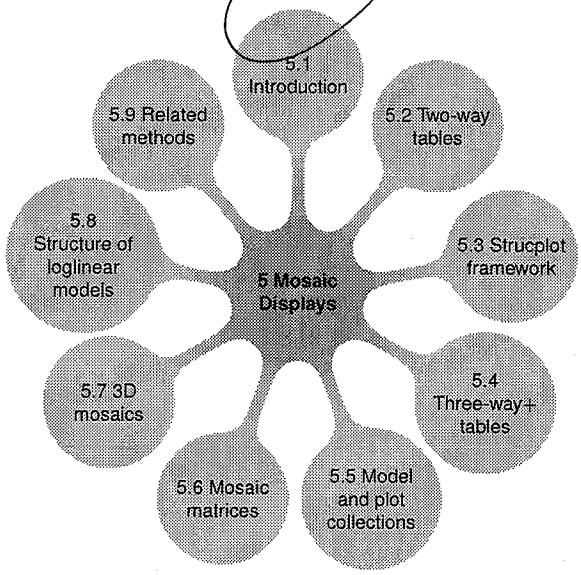
(a) Assess the strength of agreement between the raters using Cohen's κ, both unweighted and weighted.
(b) Use `agreementplot()` for a graphical display of agreement here.
(c) Assess the hypothesis that the ratings are *symmetric* around the main diagonal, using an appropriate χ^2 test. *Hint*: Symmetry for a square table $\boldsymbol{T}$ means that $t_{ij} = t_{ji}$ for $i \neq j$. The expected frequencies under the hypothesis of symmetry are the average of the off-diagonal cells, $\boldsymbol{E} = (\boldsymbol{T} + \boldsymbol{T}^\mathsf{T})/2$.
(d) Compare the results with the output of `mcnemar.test()`.

Exercise 4.8 For the *VisualAcuity* data set:

(a) Use the code shown in the text to create the table form, `VA.tab`.
(b) Perform the CMH tests for this table.
(c) Use the `woolf_test()` described in Section 4.3.2 to test whether the association between left and right eye acuity can be considered the same for men and women.

Exercise 4.9 The graph in Figure 4.23 may be misleading, in that it doesn't take into account of the differing capacities of the 18 life boats on the *Titanic*, given in the variable `cap` in the *Lifeboats* data.

(a) Calculate a new variable, `pctloaded`, as the percentage loaded relative to the boat capacity.
(b) Produce a plot similar to Figure 4.23, showing the changes over time in this measure.



5

Mosaic Displays for n-Way Tables

Mosaic displays help to visualize the pattern of associations among variables in two-way and larger tables. Extensions of this technique can reveal partial associations, marginal associations, and shed light on the structure of loglinear models themselves.

5.1 Introduction

Little boxes on the hillside, Little boxes made of ticky tacky,
Little boxes on the hillside, Little boxes all the same.
There's a green one and a pink one, And a blue one and a yellow one,
And they're all made out of ticky tacky, And they all look just the same.

Words and music by Malvina Reynolds, ©Schroder Music Company 1962, 1990;
recorded by Pete Seeger

In Chapter 4, we described a variety of graphical techniques for visualizing the pattern of association in simple contingency tables. These methods are somewhat specialized for particular sizes and shapes of tables: 2×2 tables (fourfold display), $R \times C$ tables (tile plot, sieve diagram), square tables (agreement charts), $R \times 3$ tables (trilinear plots), and so forth.

This chapter describes the ***mosaic display*** and related graphical methods for n-way frequency tables, designed to show various aspects of high-dimensional contingency tables in a hierarchical way. These methods portray the frequencies in an n-way contingency table by a collection of rectangular "tiles" whose size (area) is proportional to the cell frequency. In this respect, the mosaic

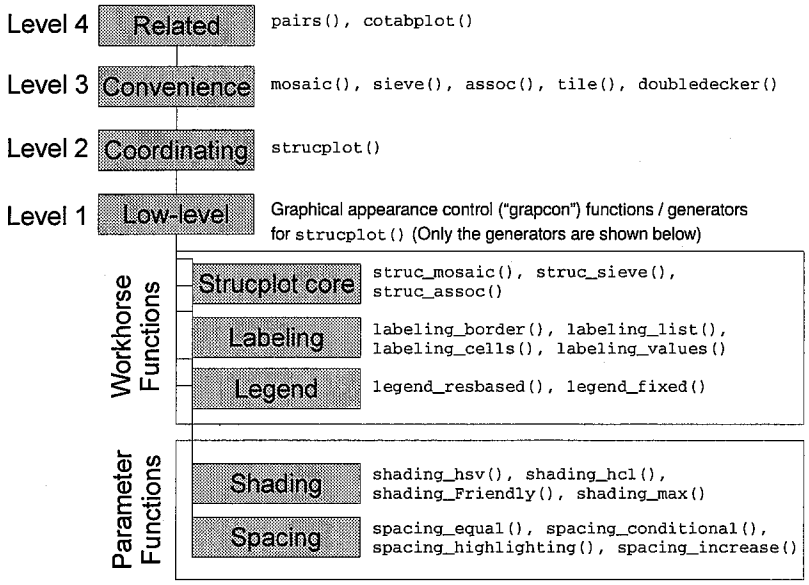


Figure 5.5: Components of the strucplot framework. High-level functions use those at lower levels to provide a general system for tile-based plots of frequency tables.

functions at the top of the diagram, but it is more convenient to describe the framework from the bottom up.

1. On the lowest level, there are several groups of workhorse and parameter functions that directly or indirectly influence the final appearance of the plot (see Table 5.1 for an overview). These are examples of *graphical appearance control* functions (called ***grapcon functions***). They are created by generating functions (*grapcon generators*), allowing flexible parameterization and extensibility (Figure 5.5 only shows the generators). The generator names follow the naming convention `group_foo()`, where `group` reflects the group the generators belong to (strucplot core, labeling, legend, shading, or spacing).

   - The workhorse functions (created by `struc_foo()`) are `labeling_foo()`, and `legend_foo()`. These functions directly produce graphical output (i.e., "add ink to the canvas"), for labels and legends respectively.
   - The parameter functions (created by `spacing_foo()` and `shading_foo()`) compute graphical parameters used by the others. The grapcon functions returned by `struc_foo()` implement the core functionality, creating the tiles and their content.

2. On the second level of the framework, a suitable combination of the low-level grapcon functions (or, alternatively, corresponding generating functions) is passed as "hyperparameters" to `strucplot()`. This central function sets up the graphical layout using grid viewports, and coordinates the specified core, labeling, shading, and spacing functions to produce the plot.

3. On the third level, vcd provides several convenience functions such as `mosaic()`, `sieve()`, `assoc()`, `tile()`, and `doubledecker()` which interface to `strucplot()` through sensible parameter defaults and support for model formulae.

4. Finally, on the fourth level, there are "related" vcd functions (such as `cotabplot()` and the

```
> loglm(~ Hair + Eye, data = haireye)

Call:
loglm(formula = ~Hair + Eye, data = haireye)

Statistics:
                     X^2 df P(> X^2)
Likelihood Ratio 146.44  9        0
Pearson          138.29  9        0
```

The output includes both the χ^2 and the deviance test statistics, both significant, indicating strong lack of fit. We now extend the analysis by including `Sex`, i.e., use the full *`HairEyeColor`* data set. In the section's introductory example, this was visualized using a mosaic plot, leading to the hypothesis whether `Hair` and `Eye` were jointly independent of `Sex`. To test this formally, we fit the corresponding model [HairEye][Sex] to the data:

```
> HE_S <- loglm(~ Hair * Eye + Sex, data = HairEyeColor)
> HE_S

Call:
loglm(formula = ~Hair * Eye + Sex, data = HairEyeColor)

Statistics:
                     X^2 df P(> X^2)
Likelihood Ratio 19.857 15  0.17750
Pearson          19.567 15  0.18917
```

giving a non-significant Pearson $\chi^2(15) = 19.567, p = 0.189$. The residuals from this model could be retrieved using

```
> residuals(HE_S, type = "pearson")
```

for further inspection. Mosaic plots can conveniently be used for this purpose, either by specifying the residuals with the `residuals=` argument, or by providing the `loglm` model formula as the `expected=` argument, letting `mosaic()` calculate them by calling `loglm()`. In the call to `mosaic()` below, the model of joint independence is specified as `expected = ~ Hair * Eye + Sex`. The strucplot labeling function `labeling_residuals` is used to display the residuals in the highlighted cells.

```
> HEC <- HairEyeColor[, c("Brown", "Hazel", "Green", "Blue"),]
> mosaic(HEC, expected = ~ Hair * Eye + Sex,
+        labeling = labeling_residuals,
+        digits = 2, rot_labels = c(right = -45))
```

Although non-significant, the two largest residuals highlighted in the plot account for nearly half ($-2.15^2 + 2.03^2 = 8.74$) of the lack of fit, and so are worthy of attention here. An easy (probably facile) interpretation is that among the blue-eyed blonds, some of the females benefited from hair products. △

5.4.2 Fitting models

When three or more variables are represented in a table, we can fit several different models of types of "independence" and display the residuals from each model. We treat these models as null or ***baseline models***, which may not fit the data particularly well. The deviations of observed frequencies from expected ones, displayed by shading, will often suggest terms to be added to an explanatory model that achieves a better fit.

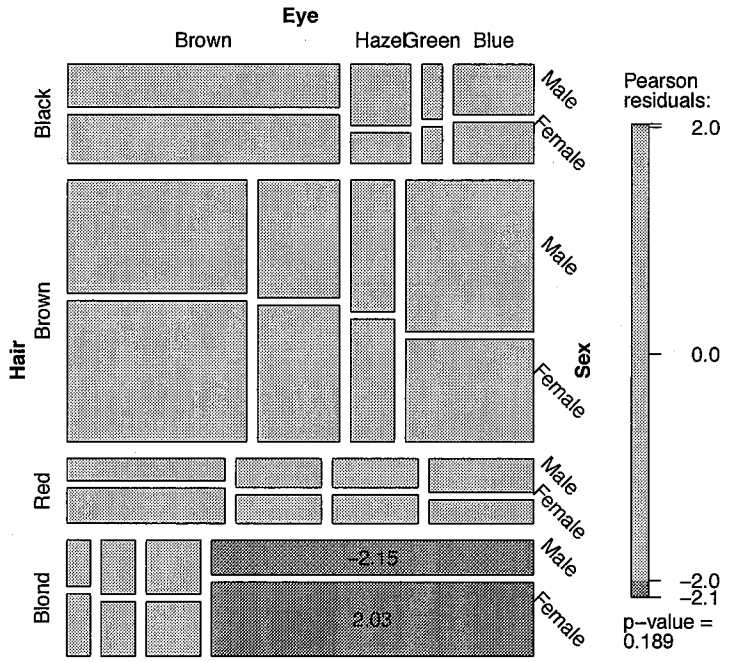


Figure 5.12: Three-way mosaic for hair color, eye color and sex. Residuals from the model of joint independence, [HE][S] are shown by shading.

For a three-way table, with variables A, B and C, some of the hypothesized models that can be fit are described below and summarized in Table 5.2. Here we use $[\bullet]$ notation to list the ***high-order terms*** in a hierarchical loglinear model; these correspond to the margins of the table that are fitted exactly, and which translate directly into R formulas used in `loglm()` and `mosaic(..., expected=)`.

The notation [AB][AC], for example, is shorthand for the model `loglm(~ A*B + A*C)` that implies

$$\log(m_{ijk}) = \mu + \lambda_i^A + \lambda_j^B + \lambda_k^C + \lambda_{ij}^{AB} + \lambda_{ik}^{AC} \,, \tag{5.3}$$

and reproduces the $\{AB\}$ and $\{AC\}$ marginal subtables.[7] That is, the calculated expected frequencies in these margins are always equal to the corresponding observed frequencies, $m_{ij+} = n_{ij+}$ and $m_{i+k} = n_{i+k}$.

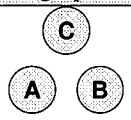
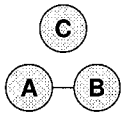
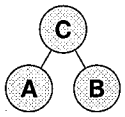
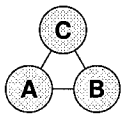
In this table, $A \perp B$ is read, "A is independent of B." The independence interpretation of the model Eqn. (5.3) is $B \perp C \mid A$, which can be read as "B is independent of C, given (conditional on) A." Table 5.2 also depicts the relations among variables as an ***association graph***, where associated variables are connected by an edge and variables that are asserted to be independent are unconnected. In mosaic-like displays, other associations present in the data will appear in the pattern of residuals.

For a three-way table, there are four general classes of independence models illustrated in Table 5.2, as described below.[8] Not included here is the ***saturated model***, [ABC], which fits the observed data exactly.

[7] The notation here uses curly braces, $\{\bullet\}$ to indicate a marginal subtable summed over all other variables.

[8] For H_2 and H_3, permutation of the variables A, B, and C gives other members of each class.

Table 5.2: Fitted margins, model symbols, and interpretations for some hypotheses for a three-way table.

Hypothesis	Fitted margins	Model symbol	Independence interpretation	Association graph
H_1	$n_{i++}, n_{+j+}, n_{++k}$	[A][B][C]	$A \perp B \perp C$	
H_2	n_{ij+}, n_{++k}	[AB][C]	$(A, B) \perp C$	
H_3	n_{i+k}, n_{+jk}	[AC][BC]	$A \perp B \mid C$	
H_4	$n_{ij+}, n_{i+k}, n_{+jk}$	[AB][AC][BC]	NA	

H_1: **Complete independence.** The model of complete (mutual) independence, symbolized $A \perp B \perp C$, with model formula `~ A + B + C`, asserts that all joint probabilities are products of the one-way marginal probabilities:

$$\pi_{ijk} = \pi_{i++}\, \pi_{+j+}\, \pi_{++k}\,,$$

for all i, j, k in a three-way table. This corresponds to the log-linear model [A][B][C]. Fitting this model puts all higher terms, and hence all association among the variables, into the residuals.

H_2: **Joint independence.** Another possibility is to fit the model in which variable C is jointly independent of variables A and B, ($\{A, B\} \perp C$), with model formula `~ A*B + C`, where

$$\pi_{ijk} = \pi_{ij+}\, \pi_{++k}\,.$$

This corresponds to the loglinear model [AB][C]. Residuals from this model show the extent to which variable C is related to the combinations of variables A and B but they do not show any association between A and B, since that association is fitted exactly. For this model, variable C is also independent of A and B in the marginal $\{AC\}$ table (collapsing over B) and in the marginal $\{BC\}$.

H_3: **Conditional independence.** Two variables, say A and B, are conditionally independent given the third (C) if A and B are independent when we control for C, symbolized as $A \perp B \mid C$, and model formula `~ A*C + B*C` (or `~ (A + B) * C`). This means that conditional probabilities, $\pi_{ij|k}$ obey

$$\pi_{ij|k} = \pi_{i+|k}\, \pi_{+j|k}\,,$$

where $\pi_{ij|k} = \pi_{ijk}/\pi_{ij+}$, $\pi_{i+|k} = \pi_{i+k}/\pi_{i++}$, and $\pi_{+j|k} = \pi_{+jk}/\pi_{+j+}$. The corresponding loglinear models is denoted [AC][BC]. When this model is fit, the mosaic display shows the conditional associations between variables A and B, controlling for C, but does not show the associations between A and C, or B and C.

H_4: **No three-way interaction.** For this model, no pair is marginally or conditionally independent, so there is *no* independence interpretation. Nor is there a closed-form expression for the cell probabilities. However, the association between any two variables is the same at each level of the third variable. The corresponding loglinear model formula is [AB][AC][BC] , indicating that all two-way margins are fit exactly and so only the three-way association is shown in the mosaic residuals.

EXAMPLE 5.8: Hair color, eye color, and sex

We continue with the analysis of the `HairEyeColor` data from Example 5.6 and Example 5.7. Figure 5.12 showed the fit of the joint-independence model [HairEye][Sex], testing whether the joint distribution of hair color and eye color is associated with sex.

Any other model fit to this table will have the same size tiles in the mosaic since the areas depend on the observed frequencies; the residuals, and hence the shading of the tiles will differ. Figure 5.13 shows mosaics for two other models. Shading in the left panel shows residuals from the model of mutual independence, [Hair][Eye][Sex], and so includes all sources of association among these three variables. The right panel shows the conditional independence model, [HairSex][EyeSex], testing whether hair color and eye color are independent, given sex. Note that the pattern of residuals here is similar to that in the two-way display, Figure 5.4, that collapsed over sex.

```
> abbrev <- list(abbreviate = c(FALSE, FALSE, 1))
> mosaic(HEC, expected = ~ Hair + Eye + Sex, labeling_args = abbrev,
+   main = "Model: ~ Hair + Eye + Sex")
> mosaic(HEC, expected = ~ Hair * Sex + Eye * Sex, labeling_args = abbrev,
+        main="Model: ~ Hair*Sex + Eye*Sex")
```

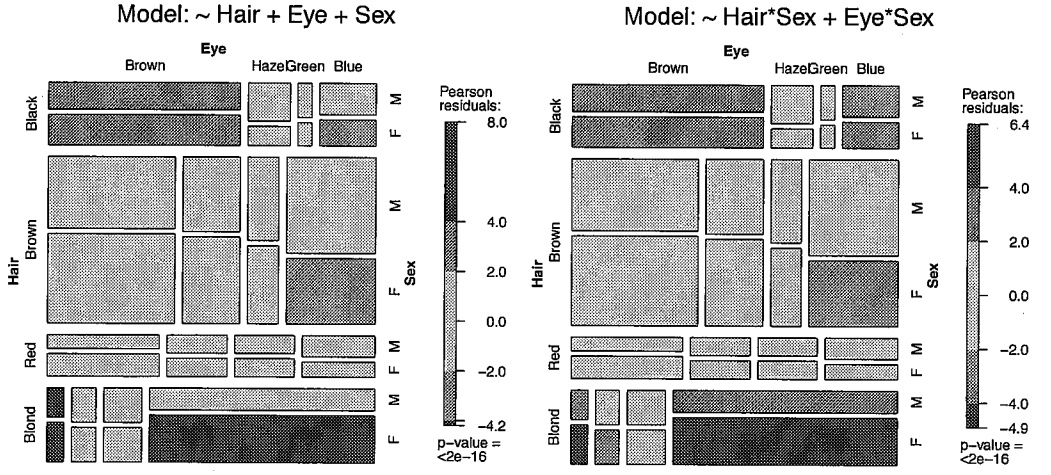


Figure 5.13: Mosaic displays for other models fit to the data on hair color, eye color, and sex. Left: Mutual independence model; right: Conditional independence of hair color and eye color given sex.

Compared with Figure 5.12 for the joint independence model, [HairEye][Sex], it is easy to see that both of these models fit very poorly.

That is, the overall likelihood ratio G^2 for the conditional independence model with $(I-1)(J-1)K$ degrees of freedom is the sum of the values for the ordinary association between A and B over the levels of C (each with $(I-1)(J-1)$ degrees of freedom). The same additive relationship holds for the Pearson χ^2 statistics: $\chi^2_{A\perp B\,|\,C} = \sum_k^K \chi^2_{A\perp B\,|\,C(k)}$.

Thus, (a) the overall G^2 (χ^2) may be decomposed into portions attributable to the AB association in the layers of C, and (b) the collection of mosaic displays for the dependence of A and B for each of the levels of C provides a natural visualization of this decomposition. These provide an analog, for categorical data, of the conditioning plot, or ***coplot***, that Cleveland (1993b) has shown to be an effective display for quantitative data. See Friendly (1999a) for further details.

Mosaic and other displays in the strucplot framework for partial association can be produced in several different ways. One way is to use a model formula in the call to `mosaic()` which lists the conditioning variables after the " | " (given) symbol, as in

```
~ Memory + Attitude | Age + Education
```

Another way is to use `cotabplot()`. This takes the same kind of conditioning model formula, but presents each panel for the conditioning variables in a separate frame within a trellis-like grid.[12]

EXAMPLE 5.10: Employment status data

Data from a 1974 Danish study of 1,314 employees who had been laid off are given in the data table *Employment* in vcd (from Andersen (1991, Table 5.12)). The workers are classified by: (a) their employment status, on January 1, 1975 ("NewJob" or still "Unemployed"), (b) the length of their employment at the time of layoff, (c) the cause of their layoff ("Closure", etc., or "Replaced").

```
> data("Employment", package = "vcd")
> structable(Employment)

                          EmploymentLength <1Mo 1-3Mo 3-12Mo 1-2Yr 2-5Yr >5Yr
EmploymentStatus LayoffCause
NewJob           Closure                    8    35     70    62    56   38
                 Replaced                  40    85    181    85   118   56
Unemployed       Closure                   10    42     86    80    67   35
                 Replaced                  24    42     41    16    27   10
```

In this example, it is natural to regard `EmploymentStatus` (variable A) as the response variable, and `EmploymentLength` (B) and `LayoffCause` (C) as predictors. In this case, the minimal baseline model is the joint independence model, $[A]\,[BC]$, which asserts that employment status is independent of both length and cause. This model fits quite poorly, as shown in the output from `loglm()` below.

```
> loglm(~ EmploymentStatus + EmploymentLength * LayoffCause,
+       data = Employment)

Call:
loglm(formula = ~EmploymentStatus + EmploymentLength * LayoffCause,
    data = Employment)

Statistics:
                     X^2 df P(> X^2)
Likelihood Ratio 172.28 11        0
Pearson          165.70 11        0
```

The residuals, shown in Figure 5.17, indicate an opposite pattern for the two categories of `LayoffCause`: those who were laid off as a result of a closure are more likely to be unemployed,

[12] Depending on your perspective, this has the advantage of adjusting for the total frequency in each conditional panel, or the disadvantage of ignoring these differences.

```
Likelihood Ratio 23.151  5 0.00031578
Pearson          24.589  5 0.00016727
```

Extracting the model fit statistics for these partial models and adding the fit statistics for the overall model of conditional independence, [AC][BC], gives the table below, illustrating the additive property of G^2 and χ^2 (Eqn. (5.6)).

Model	df	G^2	χ^2
$A \perp B \mid C_1$	5	1.49	1.48
$A \perp B \mid C_2$	5	23.15	24.59
$A \perp B \mid C$	10	24.63	26.07

One simple way to visualize these results is to call `mosaic()` separately for each of the layers corresponding to `LayoffCause`. The result is shown in Figure 5.19.

```
> mosaic(Employment[,,"Closure"], shade = TRUE,
+        gp_args = list(interpolate = 1 : 4),
+        margin = c(right = 1), main = "Layoff: Closure")
> mosaic(Employment[,,"Replaced"], shade = TRUE,
+        gp_args = list(interpolate = 1 : 4),
+        margin = c(right = 1), main = "Layoff: Replaced")
```

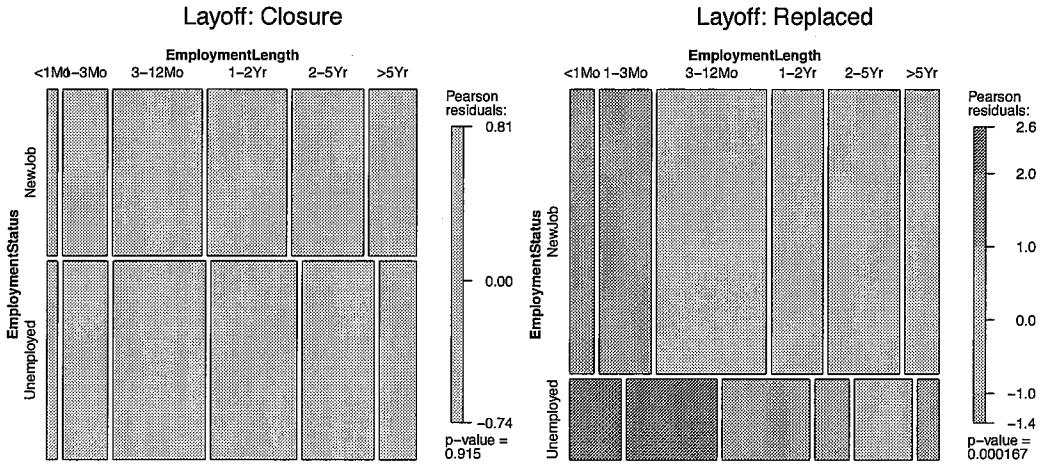


Figure 5.19: Mosaic displays for the employment status data, with separate panels for cause of layoff.

The simple summary from this example is that for workers laid off due to closure of their company, length of previous employment is unrelated to whether or not they are re-employed. However, for workers who were replaced, there is a systematic pattern: those who had been employed for three months or less are likely to remain unemployed, while those with longer job tenure are somewhat more likely to have found a new job. △

The statistical methods and R techniques described above for three-way tables extend naturally to higher-way tables, as can be seen in the next example.

EXAMPLE 5.11: Corporal punishment data

Here we use the *Punishment* data from vcd, which contains the results of a study by the Gallup Institute in Denmark in 1979 about the attitude of a random sample of 1,456 persons towards corporal punishment of children (Andersen, 1991, pp. 207–208). As shown below, this data set is a frequency data frame representing a $2 \times 2 \times 3 \times 3$ table, with table variables (a) `attitude` toward use of corporal punishment (approve of "moderate" use or "no" approval); (b) `memory` of whether the respondent had experienced corporal punishment as a child (yes/no); (c) `education` level of respondent (elementary, secondary, high); and (d) `age` category of respondent.

```
> data("Punishment", package = "vcd")
> str(Punishment, vec.len = 2)

'data.frame': 36 obs. of  5 variables:
 $ Freq     : num  1 3 20 2 8 ...
 $ attitude : Factor w/ 2 levels "no","moderate": 1 1 1 1 1 ...
 $ memory   : Factor w/ 2 levels "yes","no": 1 1 1 1 1 ...
 $ education: Factor w/ 3 levels "elementary","secondary",..: 1 1 1 2 2 ...
 $ age      : Factor w/ 3 levels "15-24","25-39",..: 1 2 3 1 2 ...
```

Of main interest here is the association between attitude toward corporal punishment as an adult (A) and memory of corporal punishment as a child (B), controlling for age (C) and education (D); that is, the model $A \perp B \mid (C, D)$, or [ACD][BCD] in shorthand notation.

As noted above, this conditional independence hypothesis can be decomposed into the 3×3 partial tests of $A \perp B \mid (C_k, D_\ell)$.

These tests and the associated graphics are somewhat easier to carry out with the data in table form (`pun`) constructed below. While we're at it, we recode the variable names and factor levels for nicer graphical displays.

```
> pun <- xtabs(Freq ~ memory + attitude + age + education,
+              data = Punishment)
> dimnames(pun) <- list(
+   Memory = c("yes", "no"),
+   Attitude = c("no", "moderate"),
+   Age = c("15-24", "25-39", "40+"),
+   Education = c("Elementary", "Secondary", "High"))
```

Then, the overall test of conditional independence can be carried using `loglm()` out as

```
> (mod.cond <- loglm(~ Memory * Age * Education +
+                      Attitude * Age * Education, data = pun))

Call:
loglm(formula = ~Memory * Age * Education + Attitude * Age *
    Education, data = pun)

Statistics:
                     X^2 df   P(> X^2)
Likelihood Ratio 39.679  9 8.6851e-06
Pearson          34.604  9 6.9964e-05
```

Alternatively, `coindep_test()` in vcd provides tests of conditional independence of two variables in a contingency table by simulation from the marginal permutation distribution of the input table. The version reporting a Pearson χ^2 statistic is given by

```
> set.seed(1071)
> coindep_test(pun, margin = c("Age", "Education"),
+              indepfun = function(x) sum(x ^ 2), aggfun = sum)


Permutation test for conditional independence
```

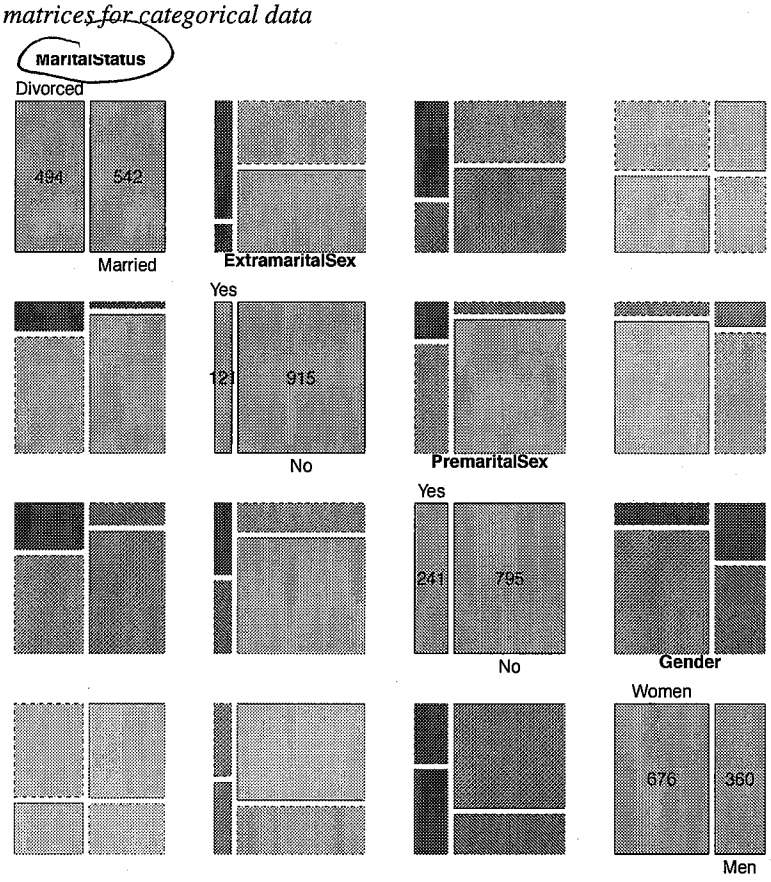


Figure 5.23: Mosaic pairs plot for the PreSex data. Each panel shows the bivariate marginal relation between the row and column variables.

marital status depends on each predictor marginally. The remaining panels show the relations within the set of explanatory variables.

Thus, we see in row 1, column 4, that marital status is independent of gender (all residuals equal zero, here), by design of the data collection. In the (1, 3) panel, we see that reported premarital sex is more often followed by divorce, while non-report is more prevalent among those still married. The (1, 2) panel shows a similar, but stronger relation between extramarital sex and marriage stability. These effects pertain to the associations of P and E with marital status (M)—the terms [PM] and [EM] in the loglinear model. We saw earlier that an interaction of P and E (the term [PEM]) is required to fully account for these data. This effect is not displayed in Figure 5.23.

Among the background variables (the loglinear term [GPE]), the (2, 3) panel shows a strong relation between premarital sex and subsequent extramarital sex, while the (2, 4) and (3, 4) panels show that men are far more likely to report premarital sex than women in this sample, and also more likely to report extramarital sex.

Even though the mosaic matrix shows only pairwise, bivariate associations, it provides an integrated view of all of these together in a single display.

△

EXAMPLE 5.14: Berkeley admissions

In Chapter 4 we examined the relations among the variables Admit, Gender, and Department in the Berkeley admissions data (Example 4.1, Example 4.11, Example 4.15) using fourfold displays

(Figure 4.5 and Figure 4.6) and sieve diagrams (Figure 4.13). These displays showed either a marginal relation (e.g., Admit, Gender) or the full three-way table.

In contrast, Figure 5.24 shows all pairwise marginal relations among these variables, produced using `pairs()`. Some additional arguments are used to control the details of labels for the diagonal and off-diagonal panels.

```
> largs <- list(labeling = labeling_border(varnames = FALSE,
+                 labels = c(T, T, F, T), alternate_labels = FALSE))
> dargs <- list(gp_varnames = gpar(fontsize = 20), offset_varnames = -1,
+               labeling = labeling_border(alternate_labels = FALSE))
> pairs(UCBAdmissions, shade = TRUE, space = 0.25,
+       diag_panel_args = dargs,
+       upper_panel_args = largs, lower_panel_args = largs)
```

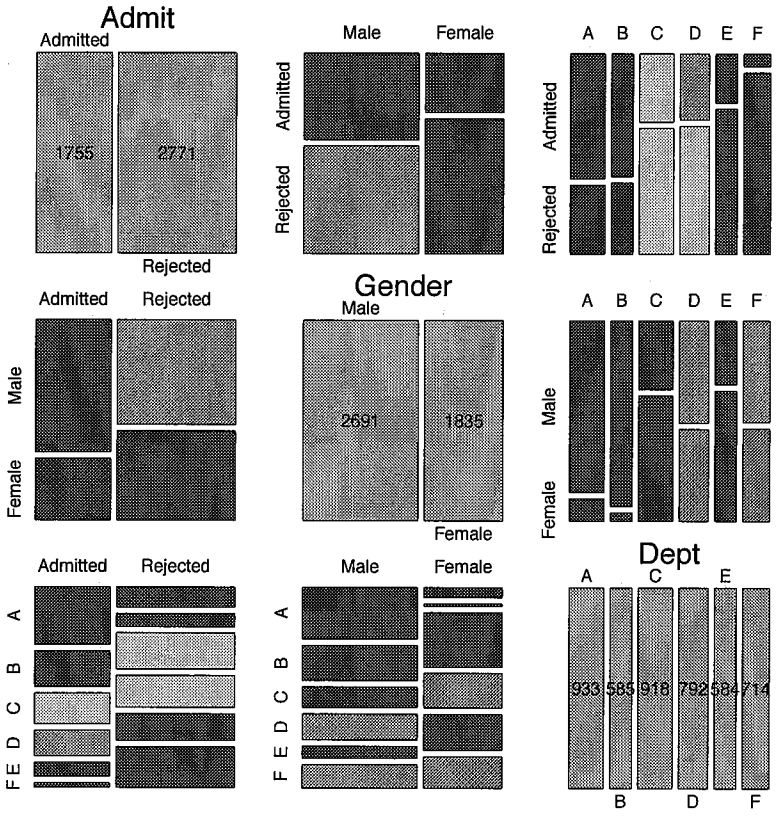


Figure 5.24: Mosaic matrix of the UCBAdmissions data showing bivariate marginal relations.

The panel in row 2, column 1 shows that Admission and Gender are strongly associated marginally, as we saw in Figure 4.5, and overall, males are more often admitted. The diagonally opposite panel (row 1, column 2) shows the same relation, splitting first by gender.[14]

The panels in the third column (and third row) provide the explanation for the paradoxical result

[14]Note that this is different than just the transpose or interchange of horizontal and vertical dimensions as in a scatterplot matrix, because the mosaic display splits the total frequency first by the horizontal variable and then (conditionally) by the vertical variable. The areas of all corresponding tiles are the same in each diagonally opposite pair, however, as are the residuals shown by color and shading.

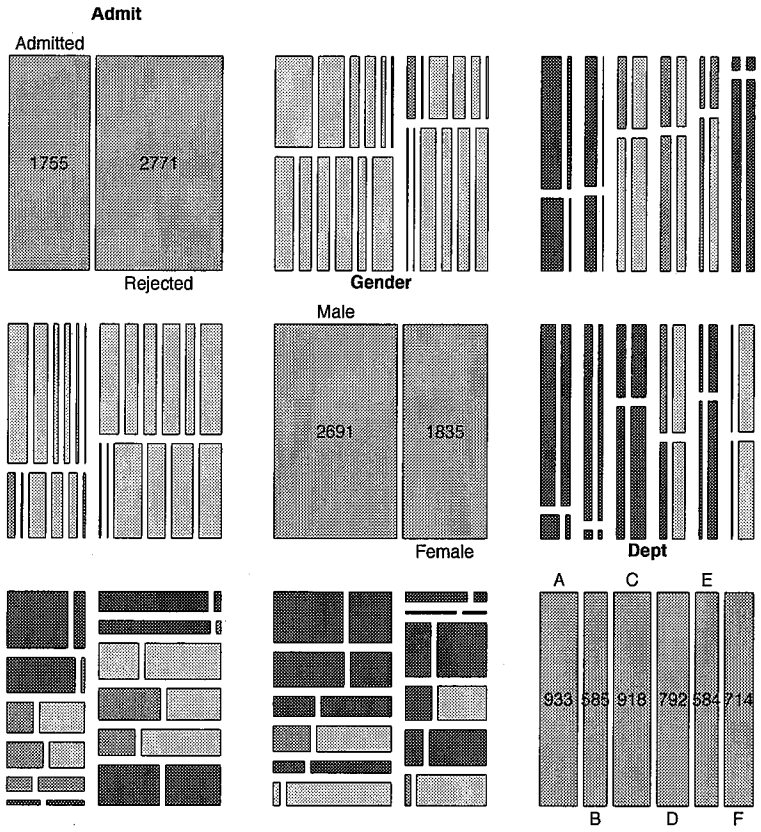


Figure 5.26: Generalized mosaic matrix of the UCBAdmissions data. The off-diagonal plots fit models of conditional indpendence.

boxplots to show the distributions of age for each of the categorical variables. The last column shows these same variables as stripplots (or "barcodes"), which show all the individual observations. In the (1, 4) and (4, 1) panels, it can be seen that younger patients are more likely to report no improvement. The other panels in the first row (and column) show that improvement is more likely in the treated condition and greater among women than men. △

5.7 3D mosaics

Mosaic-like displays use the idea of recursive partitioning of a unit square to portray the frequencies in an n-way table by the area of rectangular tiles with (x, y) coordinates. The same idea extends naturally to a 3D graphic. This starts with a unit cube, which is successively subdivided into 3D cuboids along (x, y, z) dimensions, and the frequency in a table cell is then represented by volume.

As in the 2D versions, each cuboid can be shaded to represent some other feature of the data, typically the residual from some model of independence. In principle, the display can accommodate more than 3 variables by using a sequence of split directions along the (x, y, z) axes.

One difficulty in implementing this method is that, short of using a 3D printer, the canvas for a 3D plot on a screen or printer is still projected on a two-dimensional surface, and graphical elements (volumes, lines, text) toward the front of the view will obscure those in the back. In R, a major advance in 3D graphics is available in the rgl (Adler and Murdoch, 2014) package, that mitigates

which

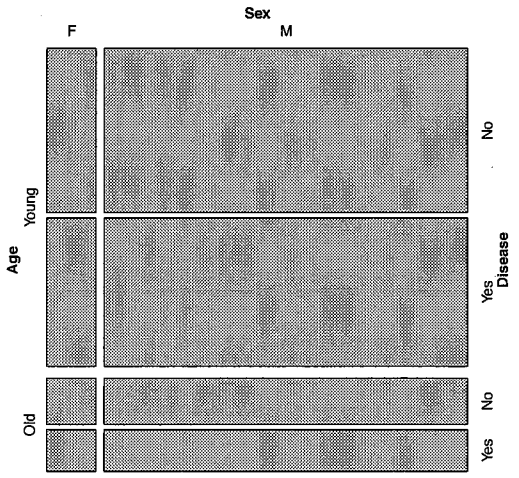


Figure 5.30: Mosaic display for the data on age, sex, and disease, using expected frequencies under mutual independence.

```
> mutual <- loglm(~ Age + Sex + Disease, data = struc, fitted = TRUE)
> fit <- as.table(fitted(mutual))
> structable(fit)

             Sex        F        M
Age   Disease
Young No           34.0991 253.3077
      Yes          30.7992 228.7940
Old   No           10.0365  74.5567
      Yes           9.0652  67.3416
```

These fitted frequencies then have the same one-way margins as the data in `struc`, but have no two-way or higher associations. Then `pairs()` for this table, using `type="total"`, shows the three-way mosaic for each pair of variables, giving the result in Figure 5.31. We use `gp=shading_Friendly` to explicitly indicate the zero residuals in the display,

```
> pairs(fit, gp = shading_Friendly2, type = "total")
```

In this figure the same data are shown in all the off-diagonal panels and the mutual independence model was fitted in each case, but with the table variables permuted. All residuals are exactly zero in all cells, by construction. We see that in each view, the four large tiles corresponding to the first two variables align, indicating that these two variables are marginally independent. For example, in the (1, 2) panel, age and sex are independent, collapsed over disease.

Moreover, comparing the top half to the bottom half in any panel we see that the divisions by the third variable are the same for both levels of the second variable. In the (1, 2) panel, for example, age and disease are independent for both males and females. This means that age and sex are conditionally independent given disease (age $\perp$ sex | disease).

Because this holds in all six panels, we see that mutual independence implies that *all pairs* of variables are conditionally independent, given the remaining one, $(X \perp Y \mid Z)$ for all permutations of variables. A similar argument can be used to show that joint independence also holds, i.e., $((X, Y) \perp Z)$ for all permutations of variables.

Alternatively, you can also visualize these relationships interactively in a 3D mosaic using

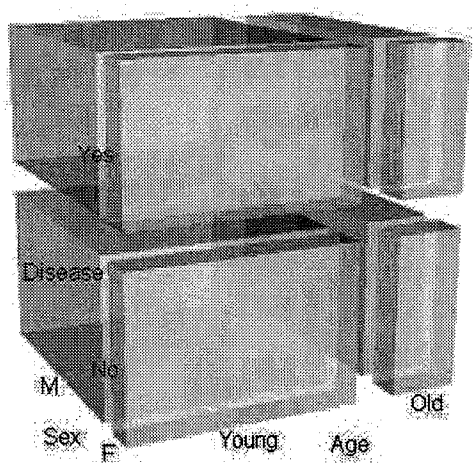


Figure 5.32: 3D mosaic plot of frequencies according to the model of mutual independence. The one-way margins are slices through the unit cube.

```
Old   No          21.542  63.051
      Yes         19.458  56.949
```

The `pairs.table()` plot, now using simpler pairwise plots (`type="pairwise"`), is shown in Figure 5.33.

```
> pairs(fit, gp = shading_Friendly2)
```

This shows, in row 3 and column 3, the anticipated independence of both age and sex with disease, collapsing over the remaining variable. The (1, 2) and (2, 1) panels show that age and sex are still associated when disease is ignored.

5.9 Related visualization methods

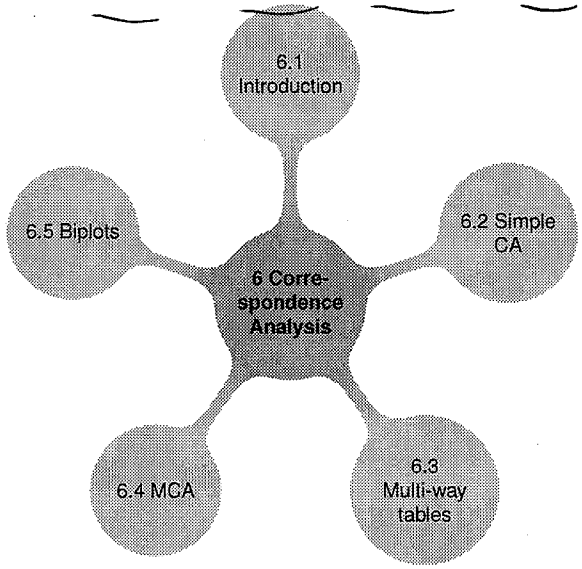
A variety of other graphical methods provide the means for visualizing relationships in multiway frequency tables. We briefly describe a few of these here, without much detail, to give a sense of some alternatives.

5.9.1 Doubledecker plots

Doubledecker plots visualize the dependence of one categorical (typically binary) variable on further categorical variables. Formally, they are mosaic plots with vertical splits for all dimensions (predictors) except the last one, which represents the dependent variable (outcome). The last variable is visualized by horizontal splits, no space between the tiles, and separate colors for the levels.

They have the advantage of making it easier to "read" the differences among the conditional response proportions in relation to combinations of the explanatory variables. Moreover, for a binary response, the difference in these conditional proportions for any two columns has a direct relation to the odds ratio for a positive response in relation to those predictor levels (Hofmann, 2001).

The `doubledecker()` function in vcd takes a formula argument of the form `R ~ E1 + E2 + ...`



6

Correspondence Analysis

Correspondence analysis provides visualizations of associations in a two-way contingency table in a small number of dimensions. Multiple correspondence analysis extends this technique to *n*-way tables. Other graphical methods, including mosaic matrices and biplots provide complementary views of loglinear models for two-way and *n*-way contingency tables, but correspondence analysis methods are particularly useful for a simple visual analysis.

6.1 Introduction

> Whenever a large sample of chaotic elements is taken in hand and marshalled in the order of their magnitude, an unsuspected and most beautiful form of regularity proves to have been latent all along.
>
> Sir Francis Galton, *Natural Inheritance*, London: Macmillan, 1889.

Correspondence analysis (CA) is an exploratory technique that displays the row and column categories in a two-way contingency table as points in a graph, so that the positions of the points represent the associations in the table. Mathematically, correspondence analysis is related to the ***biplot***, to ***canonical correlation***, and to ***principal component analysis***.

This technique finds scores for the row and column categories on a small number of dimensions that account for the greatest proportion of the χ^2 for association between the row and column categories, just as principal components account for maximum variance of quantitative variables. But CA does more—the scores provide a quantification of the categories, and have the property that