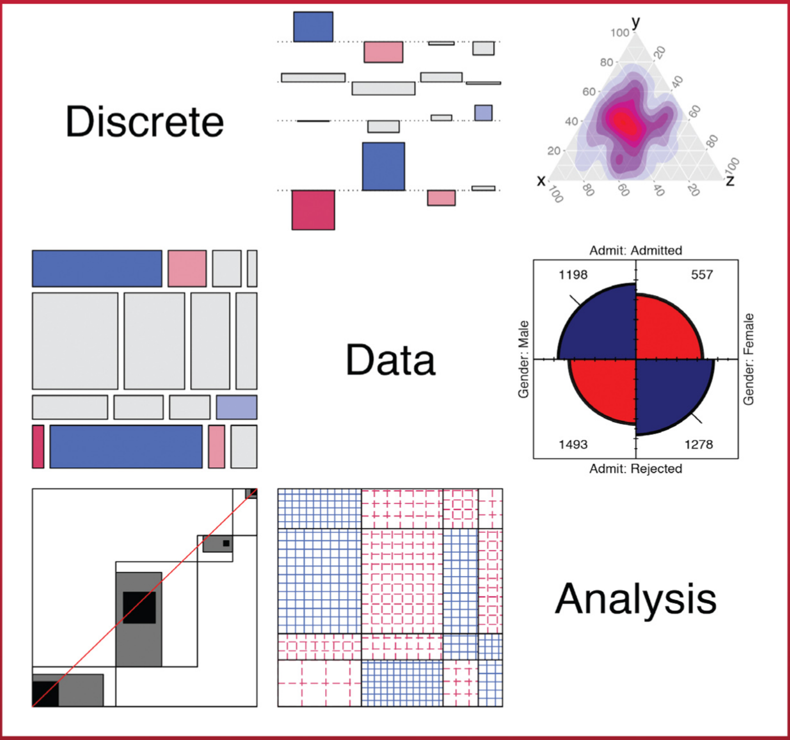
Texts in Statistical Science

Discrete Data Analysis with R

Visualization and Modeling Techniques for Categorical and Count Data



Michael Friendly
David Meyer

CRC Press
Taylor & Francis Group
A CHAPMAN & HALL BOOK

Accessing the E-book edition

Using the VitalSource® ebook

Access to the VitalBook™ ebook accompanying this book is via VitalSource® Bookshelf – an ebook reader which allows you to make and share notes and highlights on your ebooks and search across all of the ebooks that you hold on your VitalSource Bookshelf. You can access the ebook online or offline on your smartphone, tablet or PC/Mac and your notes and highlights will automatically stay in sync no matter where you make them.

1. **Create a VitalSource Bookshelf account at** ***https://online.vitalsource.com/user/new*** or log into your existing account if you already have one.
2. **Redeem the code provided in the panel below to get online access to the ebook.** Log in to Bookshelf and select **Redeem** at the top right of the screen. Enter the redemption code shown on the scratch-off panel below in the **Redeem Code** pop-up and press **Redeem**. Once the code has been redeemed your ebook will download and appear in your library.

No returns if this code has been revealed.

DOWNLOAD AND READ OFFLINE

To use your ebook offline, download BookShelf to your PC, Mac, iOS device, Android device or Kindle Fire, and log in to your Bookshelf account to access your ebook:

On your PC/Mac

Go to ***https://support.vitalsource.com/hc/en-us*** and follow the instructions to download the free **VitalSource Bookshelf** app to your PC or Mac and log into your Bookshelf account.

On your iPhone/iPod Touch/iPad

Download the free **VitalSource Bookshelf** App available via the iTunes App Store and log into your Bookshelf account. You can find more information at ***https://support.vitalsource.com/hc/en-us/categories/200134217-Bookshelf-for-iOS***

On your Android™ smartphone or tablet

Download the free **VitalSource Bookshelf** App available via Google Play and log into your Bookshelf account. You can find more information at ***https://support.vitalsource.com/hc/en-us/categories/200139976-Bookshelf-for-Android-and-Kindle-Fire***

On your Kindle Fire

Download the free **VitalSource Bookshelf** App available from Amazon and log into your Bookshelf account. You can find more information at ***https://support.vitalsource.com/hc/en-us/categories/200139976-Bookshelf-for-Android-and-Kindle-Fire***

N.B. The code in the scratch-off panel can only be used once. When you have created a Bookshelf account and redeemed the code you will be able to access the ebook online or offline on your smartphone, tablet or PC/Mac.

SUPPORT

If you have any questions about downloading Bookshelf, creating your account, or accessing and using your ebook edition, please visit ***http://support.vitalsource.com/***

Discrete Data Analysis with R

Visualization and Modeling Techniques for
Categorical and Count Data

Texts in Statistical Science

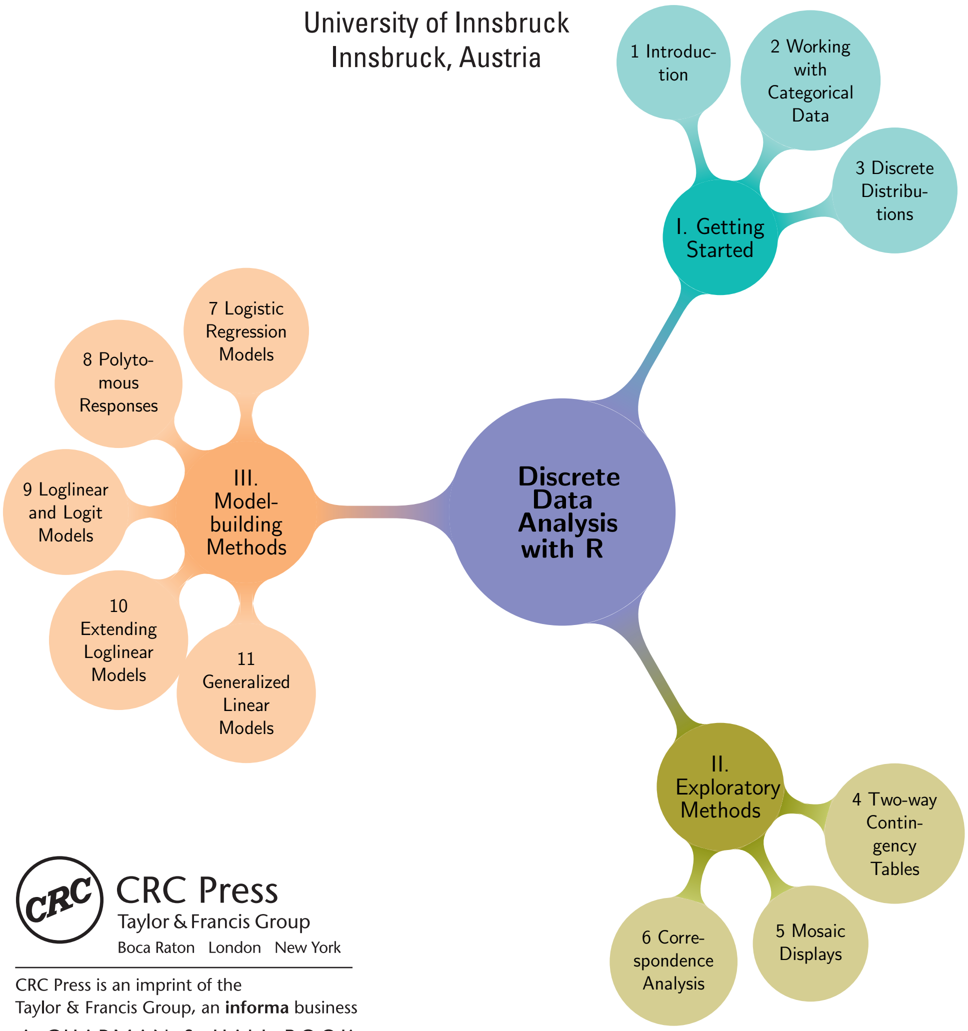
Discrete Data Analysis with R

Visualization and Modeling Techniques for Categorical and Count Data

Michael Friendly
York University
Toronto, Canada

David Meyer
UAS Technikum Wien
Vienna, Austria

with contributions by
Achim Zeileis
University of Innsbruck
Innsbruck, Austria


Discrete Data Analysis with R
I. Getting Started
1 Introduction
2 Working with Categorical Data
3 Discrete Distributions
II. Exploratory Methods
4 Two-way Contingency Tables
5 Mosaic Displays
6 Correspondence Analysis
III. Model-building Methods
7 Logistic Regression Models
8 Polytomous Responses
9 Loglinear and Logit Models
10 Extending Loglinear Models
11 Generalized Linear Models

CRC Press
Taylor & Francis Group
Boca Raton London New York

CRC Press is an imprint of the
Taylor & Francis Group, an **informa** business
A CHAPMAN & HALL BOOK

CRC Press
Taylor & Francis Group
6000 Broken Sound Parkway NW, Suite 300
Boca Raton, FL 33487-2742

© 2016 by Taylor & Francis Group, LLC
CRC Press is an imprint of Taylor & Francis Group, an Informa business

No claim to original U.S. Government works
Version Date: 20151113

International Standard Book Number-13: 978-1-4987-2585-9 (eBook - PDF)

Contents

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 (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, `http://datavis.ca/books/DDAR`.

- 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.

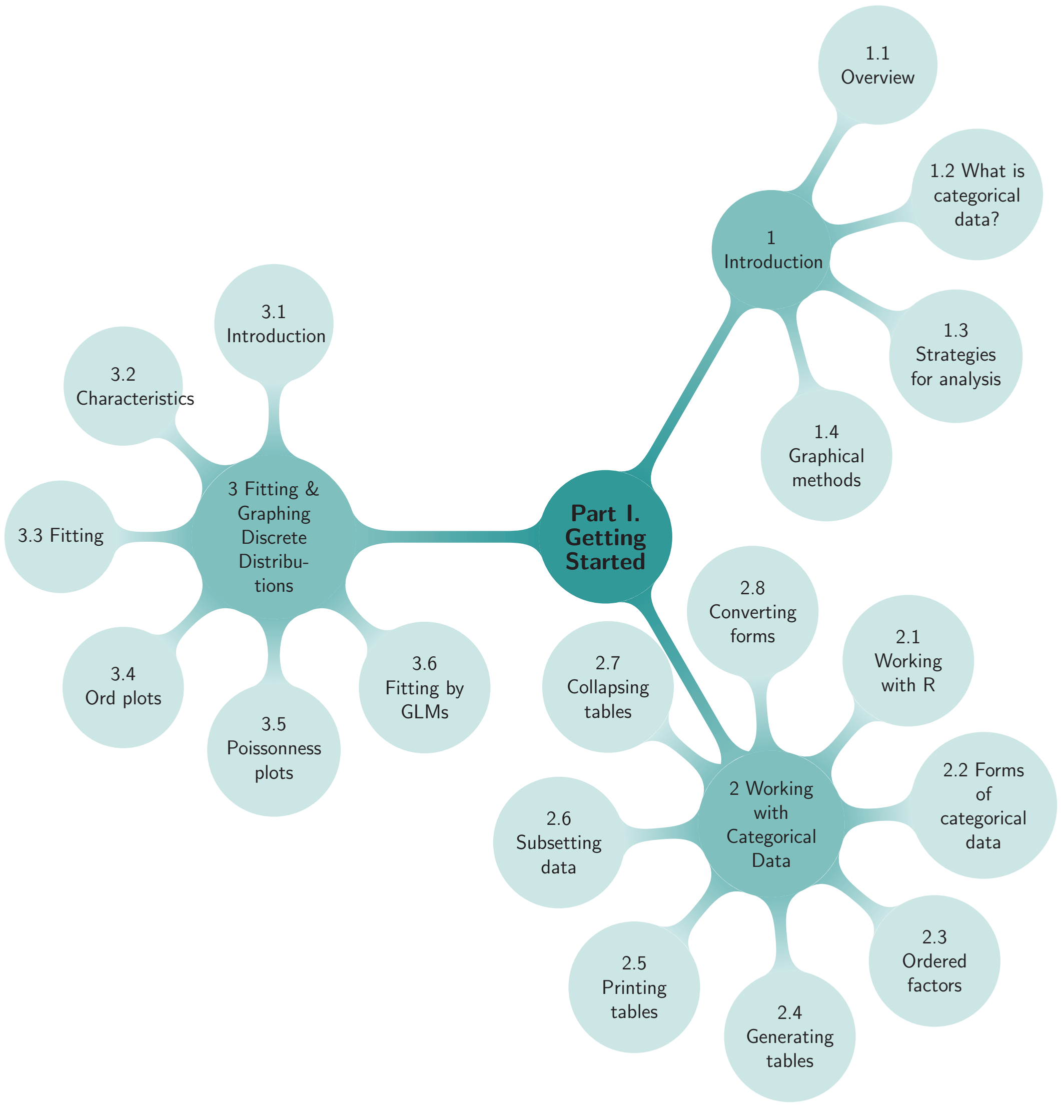
The book was written using the knitr (Xie, 2015) package, allowing a relatively seamless integration of LaTeX text, R code, and R output and graphs, so that any changes in the code were automatically incorporated in the book. Thanks are due to Yihui Xie and all the contributors to the `knitr` project 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.

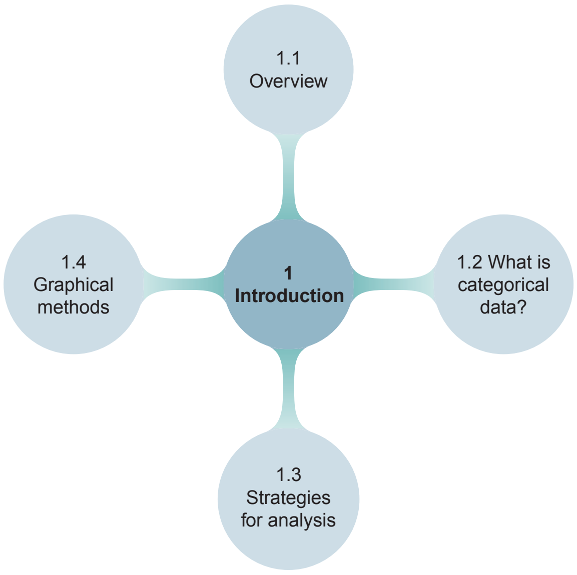
The first author's work on this project was supported by grants from the National Science and Engineering Research Council of Canada (Grant 8150) and a sabbatical leave from York University in 2013–14, during which most of this book was written.

This page intentionally left blank

Part I

Getting Started





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 in the quote from Keats, the overall aims include both beauty and truth, though each of these are only as perceived by the beholder.

Methods for visualizing quantitative data have a long history and are now widely used in both data analysis and in data presentation, and in both popular and scientific media. Graphical methods for categorical data, however, have only a more recent history, and are consequently not as widely used. The goal of this book is to show concretely how data visualization may be usefully applied to categorical data.

"Categorical" means different things in different contexts. We introduce the topic in Section 1.2 with some examples illustrating (a) types of categorical variables: binary, nominal, and ordinal, (b) data in case form vs. frequency form, (c) frequency data vs. count data, (d) univariate, bivariate, and multivariate data, and (e) the distinction between explanatory and response variables.

Statistical methods for the analysis of categorical data also fall into two quite different categories, described and illustrated in Section 1.3: (a) the simple randomization-based methods typified by the classical Pearson chi-squared (χ^2) test, Fisher's exact test, and Cochran–Mantel–Haenszel tests, and (b) the model-based methods represented by logistic regression, loglinear, and generalized linear models. In this book, Chapters 3–6 are mostly related to the randomization-based methods; Chapters 7–9 illustrate the model-based methods.

In Section 1.4 we describe some important similarities and differences between categorical data and quantitative data, and discuss the implications of these differences for visualization techniques. Section 1.4.5 outlines a strategy of data analysis focused on visualization.

In a few cases we show R code or results as illustrations here, but the fuller discussion of using R for categorical data analysis is postponed to Chapter 2.

1.2 What is categorical data?

A ***categorical variable*** is one for which the possible measured or assigned values consist of a discrete set of categories, which may be *ordered* or *unordered*. Some typical examples are:

- `Gender`, with categories "Male," "Female."
- `Marital status`, with categories "Never married," "Married," "Separated," "Divorced," "Widowed."
- `Fielding position` (in baseball), with categories "Pitcher," "Catcher," "1st base," "2nd base," ..., "Left field."
- `Side effects` (in a pharmacological study), with categories "None," "Skin rash," "Sleep disorder," "Anxiety," ....
- `Political attitude`, with categories "Left," "Center," "Right."
- `Party preference` (in Canada), with categories "NDP," "Liberal," "Conservative," "Green."
- `Treatment outcome`, with categories "no improvement," "some improvement," or "marked improvement."
- `Age`, with categories "0–9," "10–19," "20–29," "30–39," ....
- `Number of children`, with categories $0, 1, 2, \ldots$.

As these examples suggest, categorical variables differ in the number of categories: we often distinguish ***binary variables*** (or ***dichotomous variables***) such as `Gender` from those with more than two categories (called ***polytomous variables***). For example, Table 1.1 gives data on $4,526$ applicants to graduate departments at the University of California at Berkeley in 1973, classified by two binary variables, gender and admission status.

Some categorical variables (`Political attitude`, `Treatment outcome`) may have ordered categories (and are called ***ordinal variables***), while others (***nominal variables***) like `Marital`

Table 1.1: Admissions to Berkeley graduate programs

	Admitted	Rejected	Total
Males	1198	1493	2691
Females	557	1278	1835
Total	1755	2771	4526

Table 1.2: Arthritis treatment data

		Improvement			
Treatment	Sex	None	Some	Marked	Total
Active	Female	6	5	16	27
	Male	7	2	5	14
Placebo	Female	19	7	6	32
	Male	10	0	1	11
Total		42	14	28	84

`status` have unordered categories.[1] For example, Table 1.2 shows a $2 \times 2 \times 3$ table of ordered outcomes ("none," "some," or "marked" improvement) to an active treatment for rheumatoid arthritis compared to a placebo for men and women.

Finally, such variables differ in the fineness or level to which some underlying observation has been categorized for a particular purpose. From one point of view, *all* data may be considered categorical because the precision of measurement is necessarily finite, or an inherently continuous variable may be recorded only to limited precision.

But this view is not helpful for the applied researcher because it neglects the phrase "for a particular purpose." Age, for example, might be treated as a quantitative variable in a study of native language vocabulary, or as an ordered categorical variable with decade groups (0–10, 11–20, 20–30, . . .) in terms of the efficacy or side-effects of treatment for depression, or even as a binary variable ("child" vs. "adult") in an analysis of survival following an epidemic or natural disaster. In the analysis of data using categorical methods, continuous variables are often recoded into ordered categories with a small set of categories for some purpose.[2]

1.2.1 Case form vs. frequency form

In many circumstances, data is recorded on each individual or experimental unit. Data in this form is called case data, or data in ***case form***. The data in Table 1.2, for example, were derived from the individual data listed in the data set *`Arthritis`* from the vcd package. The following lines show the first five of $N = 84$ cases in the *`Arthritis`* data,

```
  ID Treatment  Sex Age Improved
1 57   Treated Male  27     Some
```

[1]An ordinal variable may be defined as one whose categories are *unambiguously* ordered along a *single* underlying dimension. Both marital status and fielding position may be weakly ordered, but not on a single dimension, and not unambiguously.

[2]This may be a waste of information available in the original variable, and should be done for substantive reasons, not mere convenience. For example, some researchers unfamiliar with regression methods often perform a "median-split" on quantitative predictors so they can use ANOVA methods. Doing this precludes the possibility of determining if those variables have nonlinear relations with the outcome while also decreasing statistical power.

```
2 46   Treated Male   29     None
3 77   Treated Male   30     None
4 17   Treated Male   32   Marked
5 36   Treated Male   46   Marked
```

Whether or not the data variables, and the questions we ask, call for categorical or quantitative data analysis, when the data are in case form, we can always trace any observation back to its individual identifier or data record (for example, if the case with `ID` equal to 57 turns out to be unusual or noteworthy).

Data in ***frequency form*** has already been tabulated, by counting over the categories of the table variables. The same data shown as a table in Table 1.2 appear in frequency form as shown below.

```
   Treatment    Sex Improved Freq
1    Placebo Female     None   19
2    Treated Female     None    6
3    Placebo   Male     None   10
4    Treated   Male     None    7
5    Placebo Female     Some    7
6    Treated Female     Some    5
7    Placebo   Male     Some    0
8    Treated   Male     Some    2
9    Placebo Female   Marked    6
10   Treated Female   Marked   16
11   Placebo   Male   Marked    1
12   Treated   Male   Marked    5
```

Data in frequency form may be analyzed by methods for quantitative data if there is a quantitative response variable (weighting each group by the cell frequency, with a weight variable). Otherwise, such data are generally best analyzed by methods for categorical data, where statistical models are often expressed as models for the frequency variable, in the form of an R formula like `Freq ~ .`.

In any case, an observation in a data set in frequency form refers to all cases in the cell collectively, and these cannot be identified individually. Data in case form can always be reduced to frequency form, but the reverse is rarely possible. In Chapter 2, we identify a third format, ***table form***, which is the R representation of a table like Table 1.2.

1.2.2 Frequency data vs. count data

In many cases the observations representing the classifications of events (or variables) are recorded from *operationally independent* experimental units or individuals, typically a sample from some population. The tabulated data may be called ***frequency data***. The data in Table 1.1 and Table 1.2 are both examples of frequency data because each tabulated observation comes from a different person.

However, if several events or variables are observed for the same units or individuals, those events are not operationally independent, and it is useful to use the term ***count data*** in this situation. These terms (following Lindsey (1995)) are by no means standard, but the distinction is often important, particularly in statistical models for categorical data.

For example, in a tabulation of the number of male children within families (Table 1.3, described in Section 1.2.3 below), the number of male children in a given family would be a *count* variable, taking values $0, 1, 2, \ldots$. The number of independent families with a given number of male children is a *frequency* variable. Count data also arise when we tabulate a sequence of events over time or under different circumstances in a number of individuals.

Table 1.3: Number of Males in 6115 Saxony Families of Size 12

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

1.2.3 Univariate, bivariate, and multivariate data

Another distinction concerns the number of variables: one, two, or (potentially) many shown in a data set or table, or used in some analysis. Table 1.1 is an example of a bivariate (two-way) contingency table and Table 1.2 classifies the observations by three variables. Yet, we will see later that the Berkeley admissions data also recorded the department to which potential students applied (giving a three-way table), and in the arthritis data, the age of subjects was also recorded.

Any contingency table (in frequency or table form) therefore records the *marginal totals*, summed over all variables not represented in the table. For data in case form, this means simply ignoring (or not recording) one or more variables; the "observations" remain the same. Data in frequency form, however, result in smaller tables when any variable is ignored; the "observations" are the cells of the contingency table. For example, in the `Arthritis` data, ignoring `Sex` gives the smaller 2×3 table for `Treatment` and `Improved`.

```
  Treatment Improved Freq
1   Placebo     None   29
2   Treated     None   13
3   Placebo     Some    7
4   Treated     Some    7
5   Placebo   Marked    7
6   Treated   Marked   21
```

In the limiting case, only one table variable may be recorded or available, giving the categorical equivalent of univariate data. For example, Table 1.3 gives data on the distribution of the number of male children in families with 12 children (discussed further in Example 3.2). These data were part of a large tabulation of the sex distribution of families in Saxony in the 19^{th} century, but the data in Table 1.3 have only one discrete classification variable, number of males. Without further information, the only statistical questions concern the form of the distribution. We discuss methods for fitting and graphing such discrete distributions in Chapter 3. The remaining chapters relate to bivariate and multivariate data.

1.2.4 Explanatory vs. response variables

Most statistical models make a distinction between ***response variables*** (or *dependent*, or *criterion* variables) and ***explanatory variables*** (or *independent*, or *predictor* variables).

In the standard (classical) linear models for regression and analysis of variance (ANOVA), for instance, we treat one (or more) variables as responses, to be explained by the other, explanatory variables. The explanatory variables may be quantitative or categorical (e.g., factors in R). This affects only the details of how the model is specified or how coefficients are interpreted for `lm()` or `glm()`. In these classical models, the response variable ("treatment outcome," for example), must be considered quantitative, and the model attempts to describe how the *mean* of the distribution of responses changes with the values or levels of the explanatory variables, such as age or gender.

When the response variable is categorical, however, the standard linear models do not apply, because they assume a normal (Gaussian) distribution for the model residuals. For example, in Table 1.2 the response variable is `Improvement`, and even if numerical scores were assigned to the categories "none," "some," "marked," it may be unlikely that the assumptions of the classical linear models could be met.

Hence, a categorical *response* variable generally requires analysis using methods for categorical data, but categorical *explanatory* variables may be readily handled by either method.

The distinction between response and explanatory variables also becomes important in the use of loglinear models for frequency tables (described in Chapter 9), where models can be specified in a simpler way (as equivalent logit models) by focusing on the response variable.

1.3 Strategies for categorical data analysis

Data analysis typically begins with exploratory and graphical methods designed to expose features of the data, followed by statistical analysis designed to summarize results, answer questions, and draw conclusions. Statistical methods for the analysis of categorical data can be classified into two broad categories: those concerned with *hypothesis testing* per se versus those concerned with *model building*.

1.3.1 Hypothesis testing approaches

In many studies, the questions of substantive interest translate readily into questions concerning hypotheses about ***association*** between variables, a more general idea than that of correlation (*linear* association) for quantitative variables. If a non-zero association exists, we may wish to characterize the strength of the association numerically and understand the pattern or nature of the association.

For example, in Table 1.1, a main question is: "Is there evidence of gender-bias in admission to graduate school?" Another way to frame this: "Are males more likely to be admitted?" These questions can be expressed in terms of an association between gender and admission status in a 2×2 contingency table of applicants classified by these two variables. If there is evidence for an association, we can assess its strength by a variety of measures, including the difference in proportions admitted for men and women or the ratio of the odds of admission for men compared to women, as described in Section 4.2.2.

Similarly, in Table 1.2, questions about the efficacy of the treatment for rheumatoid arthritis can be answered in terms of hypotheses about the associations among the table variables: `Treatment`, `Sex`, and the `Improvement` categories. Although the main concern might be focused on the overall association between `Treatment` and `Improvement`, one would also wish to know if this association is the same for men and women. A ***stratified analysis*** (Section 4.3) controls for the effects of background variables like Sex, tests for ***homogeneity of association***, and helps to determine if these associations are equal.

Questions involving tests of such hypotheses are answered most easily using a large variety of specific statistical tests, often based on randomization arguments. These include the familiar Pearson chi-squared test for two-way tables, the Cochran–Mantel–Haenszel test statistics, Fisher's exact test, and a wide range of measures of strength of association. These tests make minimal assumptions, principally requiring that subjects or experimental units have been randomly assigned to the categories of experimental factors. The hypothesis testing approach is illustrated in Chapters 4–6, though the emphasis is on graphical methods that help us to understand the nature of association between variables.

EXAMPLE 1.1: Hair color and eye color

The data set *`HairEye`* below records data on the relationship between hair color and eye color in a sample of nearly 600 students.

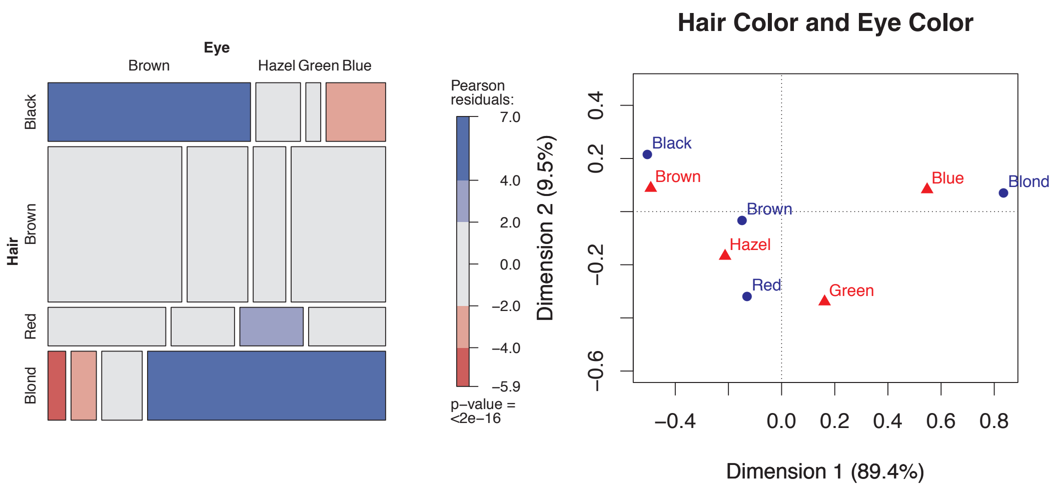


Figure 1.1: Graphical displays for the hair color and eye color data. Left: mosaic display; right: correspondence analysis plot.

```
       Eye
Hair    Brown Blue Hazel Green
  Black    68   20    15     5
  Brown   119   84    54    29
  Red      26   17    14    14
  Blond     7   94    10    16
```

The standard analysis (with `chisq.test()` or `assocstats()`) gives a Pearson χ^2 of 138.3 with nine degrees of freedom, indicating substantial departure from independence. Among the measures of strength of association, ***Cramer's V***, $V = \sqrt{\chi^2/N \min(r-1, c-1)} = 0.279$, indicates a substantial relationship between hair and eye color.[3]

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

Phi-Coefficient   : NA
Contingency Coeff.: 0.435
Cramer's V        : 0.279
```

The further (and perhaps more interesting question) is how do we understand the *nature* of this association between hair and eye color? Two graphical methods related to the hypothesis testing approach are shown in Figure 1.1.

The left panel of Figure 1.1 is a ***mosaic display*** (Chapter 5), constructed so that the size of each rectangle is proportional to the observed cell frequency. The shading reflects the cell contribution to the χ^2 statistic—shades of blue when the observed frequency is substantially greater than the expected frequency under independence, shades of red when the observed frequency is substantially less, as shown in the legend.

The right panel of this figure shows the results of a correspondence analysis (Chapter 6), where the deviations of the hair color and eye color points from the origin accounts for as much of the χ^2 as possible in two dimensions.

[3]Cramer's V varies from 0 (no association) to 1 (perfect association).

We observe that both the hair colors and the eye colors are ordered from dark to light in the mosaic display and along Dimension 1 in the correspondence analysis plot. The deviations between observed and expected frequencies have an opposite-corner pattern in the mosaic display, except for the combination of red hair and green eyes, which also stand out as the largest values on Dimension 2 in the Correspondence analysis plot. Displays such as these provide a means to understand *how* the variables are related. △

1.3.2 Model building approaches

Model-based methods provide tests of equivalent hypotheses about associations, but offer additional advantages (at the cost of additional assumptions) not provided by the simpler hypotheses-testing approaches. Among these advantages, model-based methods provide estimates, standard errors and confidence intervals for parameters, and the ability to obtain predicted (fitted/expected) values with associated measures of precision.

We illustrate this approach here for a dichotomous response variable, where it is often convenient to construct a model relating a function of the probability, π, of one event to a linear combination of the explanatory variables. Logistic regression uses the ***logit function***,

$$\text{logit}(\pi) \equiv \log_e \left(\frac{\pi}{1-\pi} \right) ,$$

which may be interpreted as the ***log odds*** of the given event. A linear logistic model can then be expressed as

$$\text{logit}(\pi) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \dots$$

Statistical inferences from model-based methods provide tests of hypotheses for the effects of the predictors, $x_1, x_2, \dots$, but they also provide estimates of parameters in the model, $\beta_1, \beta_2, \dots$ and associated confidence intervals. Standard modeling tools allow us to graphically display the fitted response surface (with confidence or prediction intervals) and even to extrapolate these predictions beyond the given data. A particular advantage of the logit representation in the logistic regression model is that estimates of odds ratios (Section 4.2.2) may be obtained directly from the parameter estimates.

Example 1.2: Space shuttle disaster

To illustrate the model-based approach, the graph in Figure 1.2 is based on a logistic regression model predicting the probability of a failure in one of the O-ring seals used in the 24 NASA space shuttles prior to the disastrous launch of the *Challenger* in January, 1986. The explanatory variable is the ambient temperature (in Fahrenheit) at the time of the flight. The sad story behind these data, and the lessons to be learned for graphical data display, are related in Example 1.10.

Here, we simply note that the fitted model, shown by the solid line in Figure 1.2, corresponds to the prediction equation (with standard errors shown in parentheses),

$$\text{logit}(\text{Failure}) = \underset{(3.06)}{5.09} - \underset{(0.047)}{0.116}\ \text{Temperature}$$

A hypothesis test that failure probability is unassociated with temperature is equivalent to the test that the coefficient for temperature in this model equals 0; this test has a p-value of 0.014, convincing evidence for rejection.

The parameter estimate for temperature, -0.116, however, gives more information. Each 1° increase in temperature decreases the log odds of failure by 0.116, with 95% confidence interval $[-0.208, -0.0235]$. The equivalent odds ratio is $\exp(-0.116) = 0.891$ $[0.812, 0.977]$. Equivalently, a 10° *decrease* in temperature corresponds to an odds ratio of a failure of $\exp(10 \times 0.116) = 3.18$, more than tripling the odds of a failure.

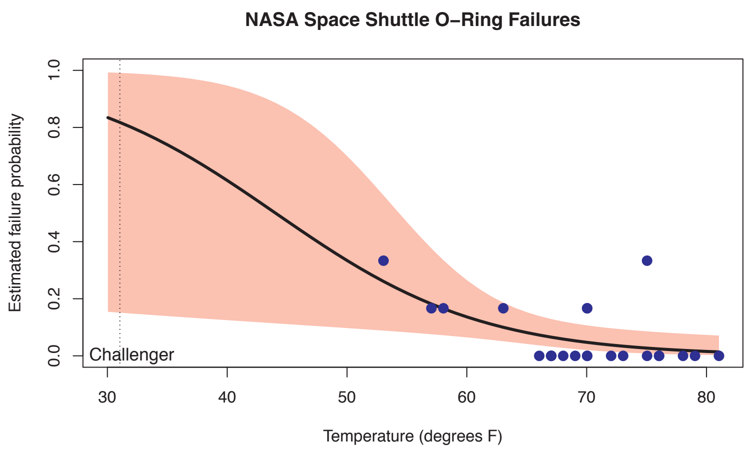


Figure 1.2: Space shuttle O-ring failure, observed and predicted probabilities. The dotted vertical line at 31° shows the prediction for the launch of the *Challenger*.

When the *Challenger* was launched, the temperature was only 31°. The shaded region in Figure 1.2 shows 95% prediction intervals for failure probability. All previous shuttles (shown by the points in the figure) had been launched at much warmer temperatures, so the prediction interval (the dashed vertical line) at 31° represents a considerable extrapolation beyond the available data. Nonetheless, the model building approach does provide such predictions along with measures of their uncertainty. Figure 1.2 is a graph that might have saved lives.

△

Example 1.3: Donner Party

In April–May of 1846 (three years before the California gold rush), the Donner and Reed families set out for California from the American Mid-west in a wagon train to seek a new life and perhaps their fortune in the new American frontier. By mid-July, a large group had reached a site in present-day Wyoming; George Donner was elected to lead what was to be called the "Donner Party," which eventually numbered 87 people in 23 wagons, along with their oxen, cattle, horses, and worldly possessions.

They were determined to reach California as quickly as possible. Lansford Hastings, a self-proclaimed trailblazer (retrospectively, of dubious distinction), proposed that the party follow him through a shorter path through the Wasatch Mountains. Their choice of "Hastings's Cutoff" proved disastrous: Hastings had never actually crossed that route himself, and the winter of of 1846 was to be one of the worst on record.

In October, 1846, heavy snow stranded them in the eastern Sierra Nevada, just to the east of a pass that bears their name today. The party made numerous attempts to seek rescue, most turned back by blizzard conditions. Relief parties in March–April 1847 rescued 40, but discovered grisly evidence that those who survived had cannibalized those who died.

Here we briefly examine how statistical models and graphical evidence can shed light on the question of who survived in the Donner party.

Figure 1.3 is an example of what we call a *data-centric, model-based* graph of a discrete (binary) outcome: lived (1) versus died (0). That is, it shows both the data and a statistical summary based on a fitted statistical model. The statistical model provides a smoothing of the discrete data.

The jittered points at the top and bottom of the graph show survival in relation to age of the person. You can see that there were more people who survived among the young, and more who died among the old. The blue curve in the plot shows the fitted probability of survival from a

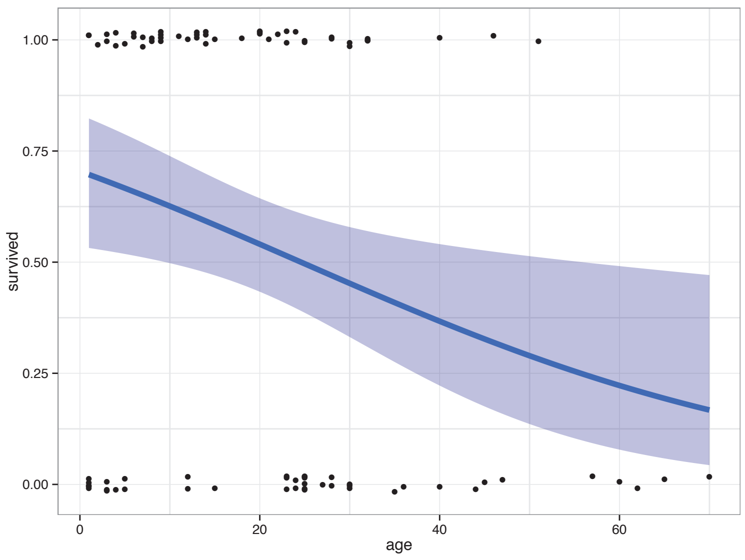


Figure 1.3: Donner party data, showing the relationship between age and survival. The blue curve and confidence band give the predicted probability of survival from a linear logistic regression model.

linear logistic regression model for these data with a 95% confidence band for the predictions. The prediction equation for this model can be given as:

$$\text{logit}(\text{survived}) = \underset{(0.372)}{0.868} - \underset{(0.015)}{0.0353}\ \text{age}$$

The equation above implies that the log odds of survival decreases by 0.0352 with each additional year of age or by $10 \times 0.0352 = 0.352$ for an additional decade. Another way to say this is that the odds of survival is multiplied by $\exp(0.353) = .702$ with each 10 years of age, a 30% decrease.

Of course, these visual and statistical summaries depend on the validity of the fitted model. For

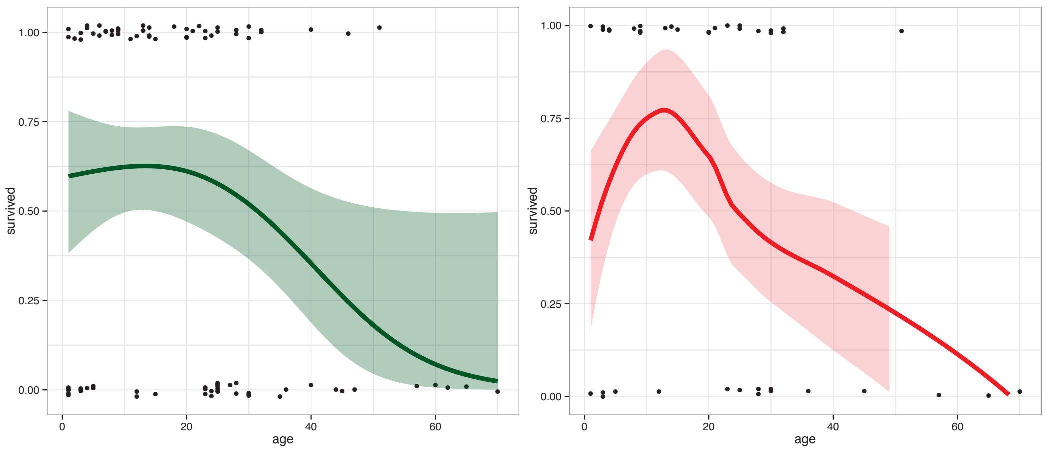


Figure 1.4: Donner party data, showing other model-based smoothers for the relationship between age and survival. Left: using a natural spline; right: using a non-parametric loess smoother.

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.

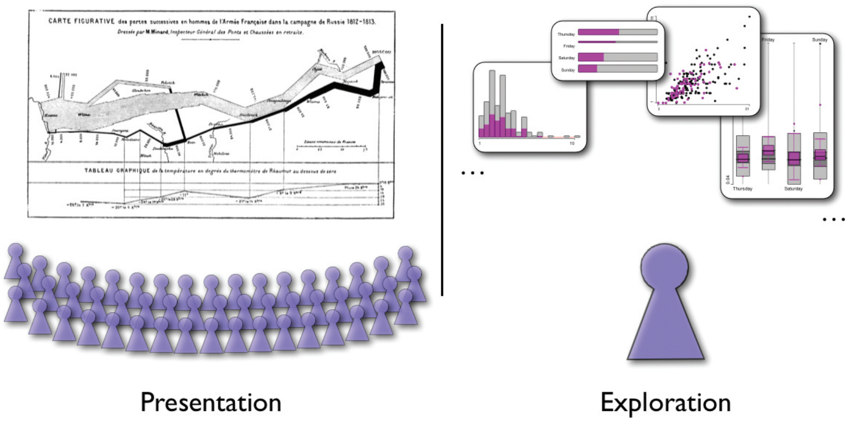


Figure 1.5: Different communication purposes require different graphs. For presentations, a single, carefully crafted graph may appeal best to a large audience; for exploratory analysis, many related images from different perspectives for a narrow audience (often you!). *Source*: Adapted from a blog entry by Martin Theus, `http://www.theusrus.de/blog/presentation-vs-exploration/`.

In this view, an effective graphical display, like good writing, requires an understanding of its *purpose*—what aspects of the data are to be communicated to the viewer. In writing we communicate most effectively when we know our audience and tailor the message appropriately. So too, we may construct a graph in different ways to: (a) use ourselves, (b) present at a conference or meeting of our colleagues, (c) publish in a research report, or (d) communicate to a general audience (Friendly (1991, Ch. 1), Friendly and Kwan (2011)). Figure 1.5 illustrates a basic contrast between graphs for presentation purposes, designed to appeal persuasively to a large audience (one-to-many) and the use of perhaps many graphs we might make for ourselves for exploratory data analysis (many-to-one).

Figure 1.6 shows one organization of visualization methods in terms of the *primary* use or

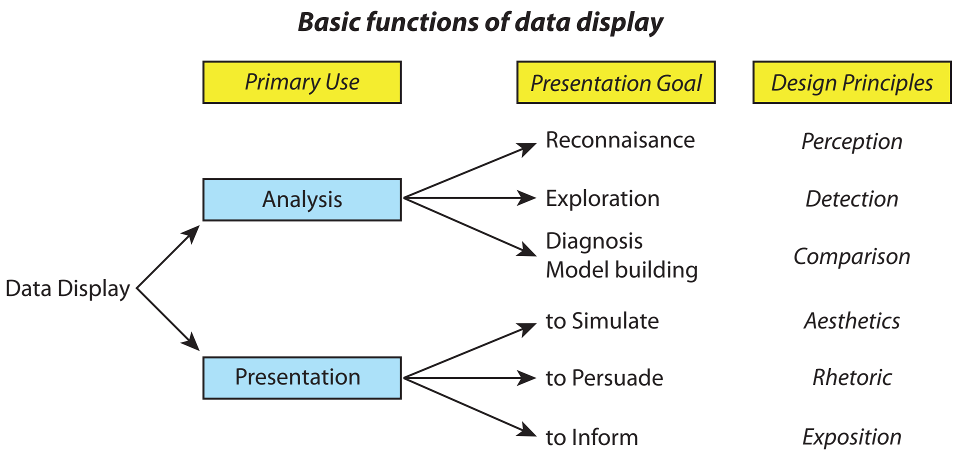


Figure 1.6: A taxonomy of the basic functions of data display by intended use, presentation goal, and design principles.

intended communication goal, the functional *presentation goal*, and suggested corresponding *design principles*.

We illustrate these ideas and distinctions in the examples below, most of which are treated again in later chapters.

EXAMPLE 1.4: Racial profiling: Arrests for marijuana possession

In a case study that will be examined in detail in Chapter 7 (Example 7.10), the *Toronto Star* newspaper studied a huge data base of arrest records by Toronto police for indications of possible racial profiling, i.e., differential treatment of those arrested on the basis of skin color. They focused on the charge of simple possession of a small amount of marijuana, for which enforcement procedures allowed police discretion. An officer could release an arrestee with a summons ("Form 9") to appear in court, or take the person to a police station for questioning ("Form 10") or booking ("Form 11.1"), or order the person held in jail for a bail hearing ("Show cause").

The statistical issue was whether the data on these arrests showed evidence of differential treatment in relation to skin color, particularly in the treatment of blacks vs. whites, controlling, of course, for other factors. Statistical tests on these data (χ^2 tests, loglinear models, logistic regression) showed overwhelming evidence of differential treatment of blacks and whites. However, tables of these results do not reveal the nature of this association.

Figure 1.7 is an example of a graph designed for *analysis*—a mosaic display (Chapter 5) showing the frequencies of those arrested on this charge by skin color and release type. The size of each rectangle shows the frequency and these are shaded in relation to the association between skin color and release—blue for positive associations (more than expected if they were independent) to red for negative associations.

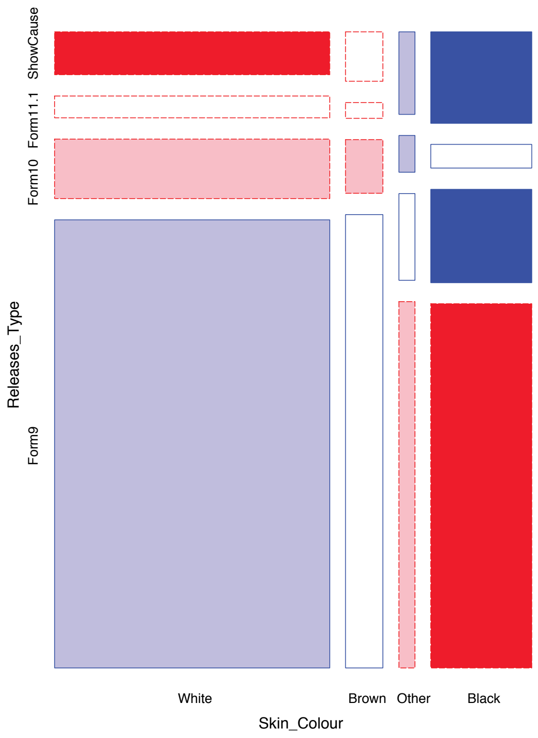


Figure 1.7: Analysis graph: A mosaic display showing the relationship between skin color and release type for those arrested on a charge of simple possession of marijuana in Toronto, 1996–2002.

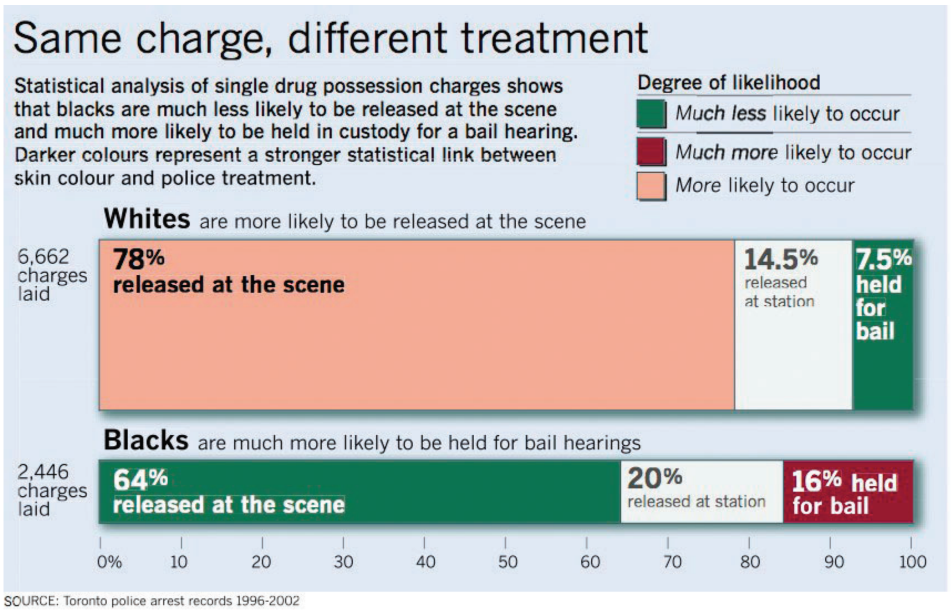


Figure 1.8: Redesign of Figure 1.7 as a presentation graphic. *Source*: Graphics department, *The Toronto Star*, December 11, 2002. Used by permission.

Once you know how to read such graphs, the pattern is clear: blacks were indeed more likely to be held for more severe treatment, whites were more likely to be released with a summons. But 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 and ANOVA models used for quantitative data. These analogies suggest that the graphical methods commonly used for quantitative data may be adapted directly to categorical data.

Happily, it turns out that many of the analysis graphs and diagnostic displays (e.g., effect plots, influence plots, added variable and partial residual plots, etc.) that have become common adjuncts in the analysis of quantitative data have been extended to generalized linear models including logistic regression (Section 7.5) and loglinear models (Section 11.6).

Unhappily, the familiar techniques for displaying raw data are often disappointing when applied to categorical data. The simple scatterplot, for example, widely used to show the relation between quantitative response and predictors, when applied to discrete variables, gives a display of the category combinations, with all identical values overplotted, and no representation of their frequency.

Instead, frequencies of categorical variables are often best represented graphically using *areas* rather than as position along a scale. Friendly (1995) describes conceptual and statistical models that give a rationale for this graphic representation. Figure 1.7 does this in the form of a modified bar chart (mosaic plot), where the widths of the horizontal bars show the proportions of whites and blacks in the data, and the divisions of each group give the percents of each release type. Consequently, the areas of each bar are proportional to the frequency in the cells of this 2×3 table.

As we describe later in this book, using the visual attribute

$$\textbf{area} \sim \textbf{frequency}$$

also allows creating novel graphical displays of frequency data for special circumstances.

Figure 1.9 shows two examples. The left panel gives a ***fourfold display*** of the frequencies of admission and gender in the Berkeley data shown in Table 1.1. What should be seen at a glance is that males are more often admitted and females more often rejected (shaded blue); see Section 4.4 for details.

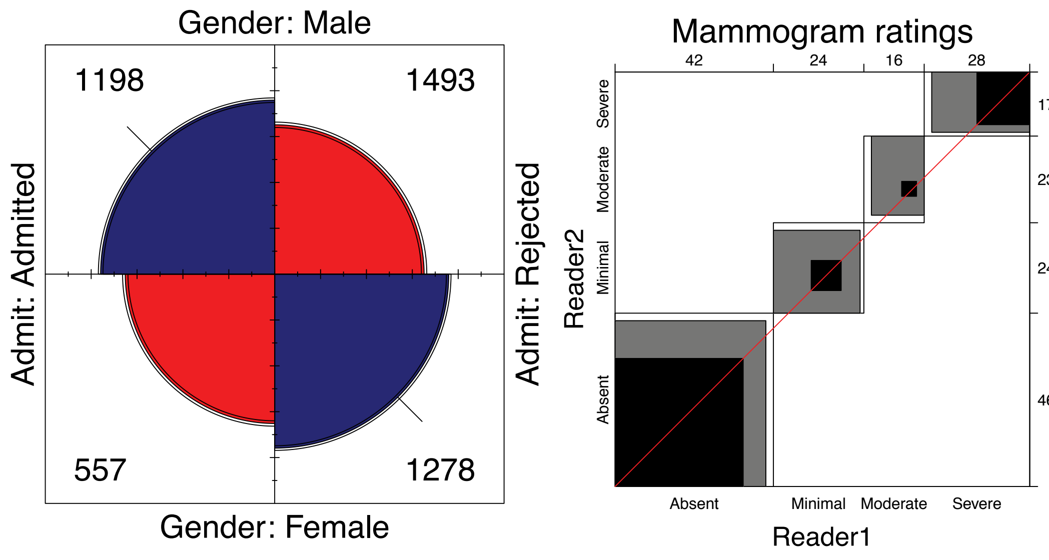


Figure 1.9: Frequencies of categorical variables shown as areas. Left: fourfold display of the relation between gender and admission in the Berkeley data; right: agreement plot for two raters assessing mammograms.

The right panel shows another specialized display, an ***agreement chart*** designed to show the strength of agreement in a square table for two raters (see Section 4.7.2). The example here (Example 4.18) concerns agreement of ratings of breast cancer from mammograms by two raters. The dark squares along the diagonal show exact agreement; the lighter diagonal rectangles allow 1-off agreement, and both are shown in relation to chance agreement (diagonal enclosing rectangles). What should be seen at a glance is that exact agreement is moderately strong and extremely strong if you allow the raters to differ by one rating category.

1.4.3 Effect ordering and rendering for data display

In plots of quantitative variables, standard methods (histograms, scatterplots) automatically position values along ordered scales, facilitating comparison ("which is less/more?") and detection of patterns, trends, and anomalies. However, by its nature, categorical data involves discrete variables such as education level, hair color, geographic region (state or province), or preference for a political party. With alphabetic labels for ordered categories (e.g., education: Low, Medium, High), it is unfortunately all too easy to end up with a nonsensical display with the categories ordered High, Low, Medium. Geographic regions (U.S. states) are often ordered alphabetically by default as are the names of political parties and other categorical variables. This may be useful for lookup, but for the purposes of comparison and detection, this is almost always a bad idea.

Instead, Friendly and Kwan (2003) proposed the principle of ***effect-order sorting*** for visual displays (tables as well as graphs):

sort the data by the effects to be seen to facilitate comparison

For quantitative data, this is often achieved by sorting the data according to means or medians of row and column factors, called ***main-effect ordering***. For categorical data, graphs and tables are often most effective when the categories are arranged in an order reflecting their association, called ***association ordering***.

Another important principle concerns the ***rendering*** of visual attributes of elements in graphical displays (Friendly, 2002). For example, categorical variables in plots (and tables) can be distinguished by any one or more of color, size, shape, or font. The examples below show the use of color to illustrate the precept:

render the data by the effects to be seen to facilitate detection

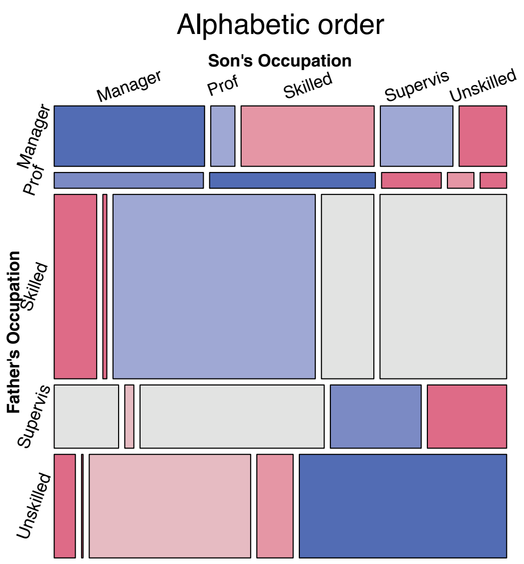
Example 1.5: British social mobility

Bishop et al. (1975, p. 100) analyzed data on the occupations of 3500 British fathers and their sons from a study by Glass (1954), with five occupational categories: Professional, Managerial, Supervisory, Skilled manual, and Unskilled manual.

One would expect, of course, a strong association between a son's occupation and that of his father—the apple doesn't fall very far from the tree. Mosaic plots (detailed in Chapter 5) provide a natural way to show such relationships. Figure 1.10 shows two such plots. The left panel shows the result obtained when the table variables `father` and `son` are read as factors, and therefore ordered alphabetically by default. It is difficult to see any overall pattern, except for the large values in the diagonal cells (shaded blue) corresponding to equal occupational status.

In the right panel, the categories have been arranged in decreasing order of occupational status to show the association according to status. Now you can see a global pattern of shading color, where the tiles become increasingly red as one moves away from the main diagonal, reflecting a greater difference between the occupation of the father and son. The interpretation here is that most sons remain in their father's occupational class, but when they differ, there is little mobility across large steps.

In this example, `father` and `son` are clearly ordinal variables and should be treated as such in both graphs and statistical models. Correspondence analysis (Chapter 6) provides a natural way



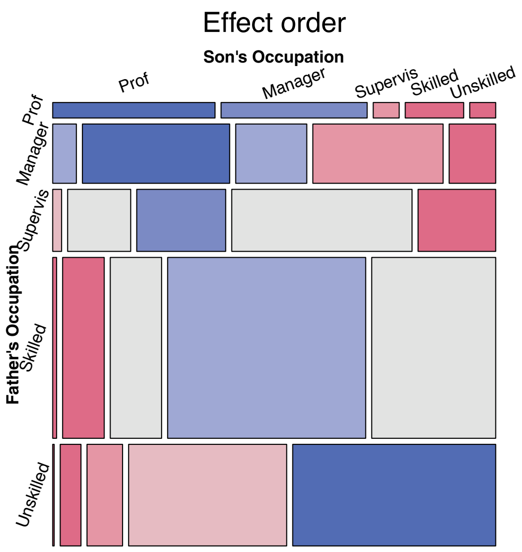


Figure 1.10: Mosaic plots for Glass' mobility table of occupational status. In these displays the area of each tile is proportional to frequency and shading color shows the departure from independence, using blue for positive, red for negative association. Left: default alphabetic ordering of categories; right: occupational categories ordered by status.

to depict association by assigning scores to the categories to optimally represent their relationships. Loglinear models provide special methods for ordinal variables (Section 10.1) and square frequency tables (Section 10.2).

△

The ideas of effect ordering and rendering with color shading to enhance perception can also be used in tabular displays, as illustrated in the next example.

EXAMPLE 1.6: Barley data

The classic `barley` dataset (in lattice (Sarkar, 2015)) from Immer et al. (1934) gives a $10 \times 2 \times 6$ table of yields of 10 varieties of barley in two years (1931, 1932) planted at 6 different sites in Minnesota. Cleveland (1993b) and many others have used this data to illustrate graphical methods, and one surprising finding not revealed in standard tabular displays is that the data for one site (Morris) may have had the values for 1931 and 1932 switched.[4]

To focus attention on this suspicious effect in a tabular display, you can calculate the *yield difference* $\Delta y_{ij} = y_{ij,1931} - y_{ij,1932}$. Table 1.4 shows these values in a 10×6 table with the rows and columns sorted by their means (main-effect ordering). In addition, the table cells have been colored according to the sign and magnitude of the year difference. The shading scheme uses blue for large positive values and red for large negative values, with a white background for intermediate values. The shading intensity values were determined as $|\Delta y_{ij}| > \{2, 3\} \times \widehat{\sigma}(\Delta y_{ij})$.

Effect ordering and color rendering have the result of revealing a new effect, shown as a regular progression in the body of the table. The negative values for Morris now immediately stand out. In addition, the largely positive other values show a lower-triangular pattern, with the size of the yield

[4]This canonical story, like many others in statistics and graphics lore, turns out to be apocryphal on closer examination. Wright (2013) recently took a closer look at the original data and gives an expanded data set as `minnesota.barley.yield` in the agridat (Wright, 2015) package. With a wider range of years (1927–1936), other local effects like weather had a greater impact than the overall year effects seen in 1931–1932, and the results for the Morris site no longer stand out as surprising.

Table 1.4: Barley data, yield differences, 1931-1932, sorted by mean difference, and shaded by value

Variety	**Site**						***Mean***
	Morris	Duluth	University Farm	Grand Rapids	Waseca	Crookston	
No. 475	-22	6	-5	4	6	12	0.1
Wisconsin No. 38	-18	2	1	14	1	14	2.4
Velvet	-13	4	13	-9	13	9	2.9
Peatland	-13	1	5	8	13	16	4.8
Manchuria	-7	6	0	11	15	7	5.5
Trebi	-3	3	7	9	15	5	6.1
Svansota	-9	3	8	13	9	20	7.3
No. 462	-17	6	11	5	21	18	7.4
Glabron	-6	4	6	15	17	12	8.0
No. 457	-15	11	17	13	16	11	8.8
Mean	-12.2	4.6	6.3	8.2	12.5	12.5	5.3

difference increasing with both row and column means. Against this background, one other cell, for Velvet grown at Grand Rapids, stands out with an anomalous negative value.

Although the use of color for graphs is now more common in some journals, color and other rendering details in tables are still difficult. The published version of Table 1.4 (Friendly and Kwan, 2003, Table 3) was forced to use only font shape (normal, italics) to distinguish positive and negative values.

△

Finally, effect ordering is also usefully applied to the variables in multivariate data sets, which by default, are often ordered in data displays according to their position in a data frame or alphabetically.

Example 1.7: Iris data

The classic `iris` data set (Anderson, 1935, Fisher, 1936b) gives the measurements in centimeters of the variables sepal length and width and petal length and width, respectively, for 50 flowers from each of 3 species of iris, *Iris setosa*, *versicolor*, and *virginica*. Such multivariate data are often displayed in ***parallel coordinate plot***s, using a separate vertical axis for each variable, scaled from its minimum to maximum.

The default plot, with variables shown in their data frame order, is shown in the left panel of Figure 1.11, and gives rise to the epithet *spaghetti plot* for such displays because of the large number of line crossings. This feature arises because one variable, sepal width, has negative relations in the species means with the other variables. Simple rearrangement of the variables to put sepal width last (or first) makes the relations among the species and the variables more apparent, as shown in the right panel of Figure 1.11. This plot has also been enhanced by using ***alpha-blending*** (partial transparency) of thicker lines, so that the density of lines is more apparent.

Parallel coordinate plots for categorical data are discussed in an online supplement on the web site for the book. A general method for reordering variables in multivariate data visualizations based on cluster analysis was proposed by Hurley (2004).

△

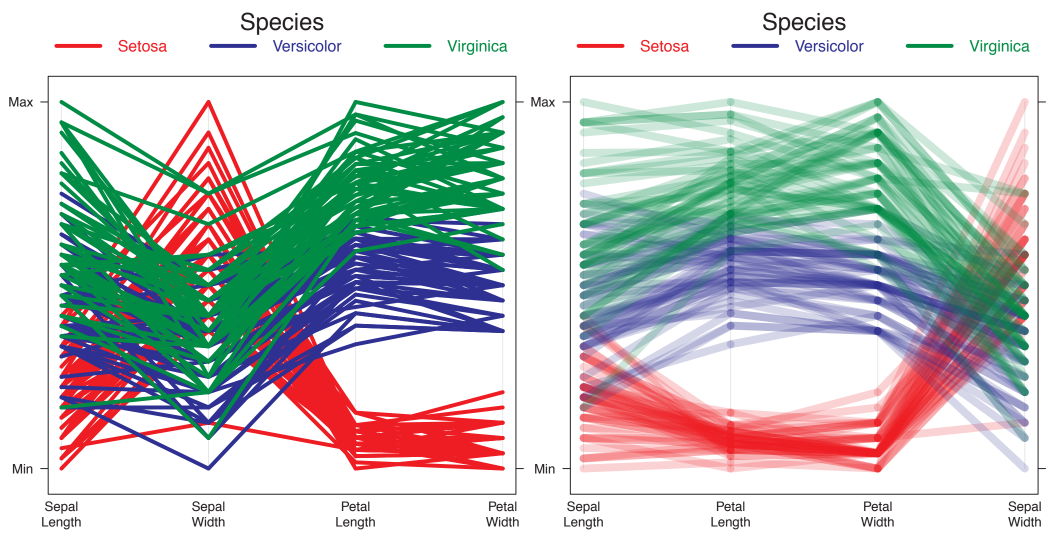


Figure 1.11: Parallel coordinates plots of the Iris data. Left: Default variable order; right: Variables ordered to make the pattern of correlations more coherent.

1.4.4 Interactive and dynamic graphics

Graphics displayed in print form, such as this book, are necessarily static and fixed at the time they are designed and rendered as an image. Yet, recent developments in software, web technology and media alternative to print have created the possibility to extend graphics in far more useful and interesting ways, for both presentation and analysis purposes.

Interactive graphics allow the viewer to directly manipulate the statistical and visual components of graphical display. These range from

- 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*

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

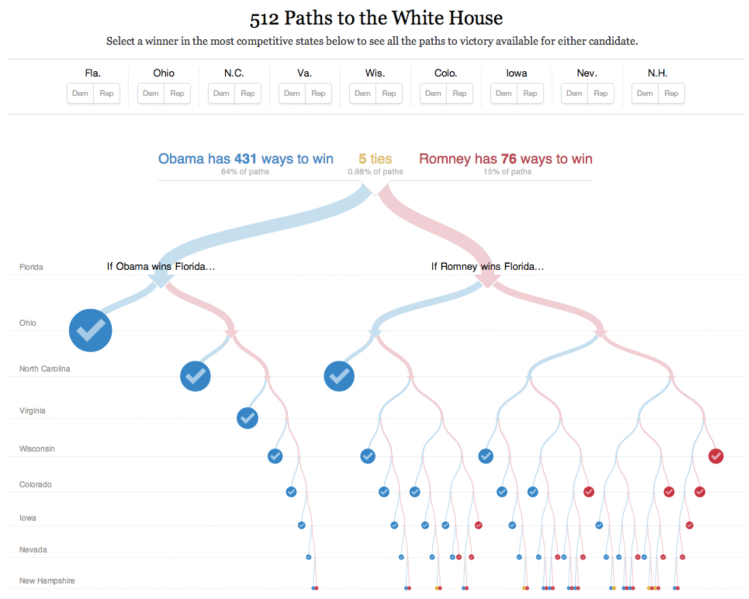


Figure 1.12: 512 paths to the White House. This interactive graphic allows the viewer to select a winner in any one or more of the nine most highly contested U.S. states and highlights the number of paths leading to a win by Obama or Romney, sorted and weighted by the number of Electoral College votes.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

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 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

[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.

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

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.

In contrast, the visualization approach to data analysis is designed to (a) expose information and structure in the data, (b) supplement the information available from numerical summaries, and (c) suggest more adequate models. In general, the visualization approach seeks to serve the needs of both summarization and exposure.

This approach recognizes that both data analysis and graphing are *iterative* processes. You should not expect that any one model captures all features of the data, any more than we should expect that a single graph shows all that may be seen. In most cases, your initial steps should include some graphical display guided by understanding of the subject matter of the data. What you learn from a graph may then help suggest features of the data to be incorporated into a fitted model. Your desire to ensure that the fitted model is an adequate summary may then lead to additional graphs.

The precept here is that

Visualization = Graphing + Fitting + Graphing . . .

where the ellipsis indicates the often iterative nature of this process. Even for descriptive purposes, an initial fit of salient features can be removed from the data, giving residuals (departures from a model). Displaying the residuals may then suggest additional features to account for.

Simple examples of this idea include detrending time series graphs to remove overall and seasonal effects and plots of residuals from main-effect models for ANOVA designs. For categorical data, mosaic plots (Chapter 5) display the unaccounted-for association between variables by shading, as in Figure 1.10. Additional models and plots considered in Section 10.2 can reveal additional structure in square tables beyond the obvious effect that sons tend most often to follow in their fathers' footsteps.

EXAMPLE 1.9: Donner Party

The graphs in Figure 1.3 and Figure 1.4 suggest three different initial descriptions for survival in the Donner party. Yet they ignore all other influences, of which gender and family structure might also be important. A more complete understanding of this data can be achieved by taking these effects into account, both in fitted models and graphs. See Example 7.9 for a continuation of this story. △

EXAMPLE 1.10: Space shuttle disaster

The space shuttle *Challenger* mentioned in Example 1.2 exploded 73 seconds after take-off on January 28, 1986. Subsequent investigation presented to the presidential commission headed by William Rogers determined that the cause was failure of the O-ring seals used to isolate the fuel supply from burning gases. The story behind the *Challenger* disaster is perhaps the most poignant missed opportunity in the history of statistical graphics. See Tufte (1997) for a complete exposition. It may be heartbreaking to find out that some important information was there, but the graphmaker missed it.

Engineers from Morton Thiokol, manufacturers of the rocket motors, had been worried about the effects of unseasonably cold weather on the O-ring seals and recommended aborting the flight.

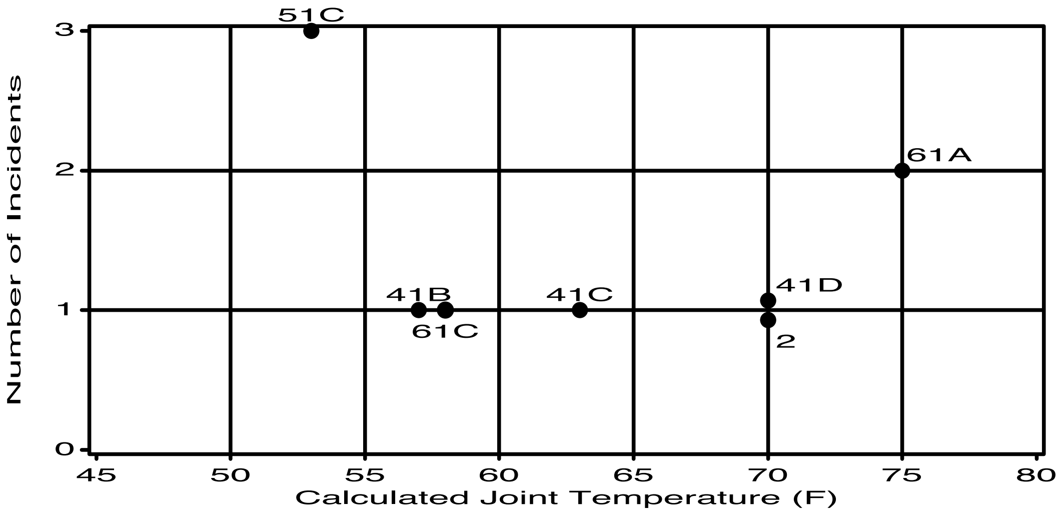


Figure 1.13: NASA Space Shuttle pre-launch graph prepared by the engineers at Morton Thiokol.

NASA staff analyzed the data, tables, and charts submitted by the engineers and concluded that there was insufficient evidence to cancel the flight.

The data relating O-ring failures to temperature were depicted as in Figure 1.13, our candidate for the most misleading graph in history. There had been 23 previous launches of these rockets giving data on the number of O-rings (out of 6) that were seen to have suffered some damage or failure. However, the engineers omitted the observations where no O-rings failed or showed signs of damage, believing that they were uninformative.

Examination of this graph seemed to indicate that there was no relation between ambient temperature and failure. Thus, the decision to launch the *Challenger* was made, in spite of the initial concerns of the Morton Thiokol engineers. Unfortunately, those observations had occurred when the launch temperature was relatively warm ($65 - -80°$F.) and were indeed informative. The coldest temperature at any previous launch was $53°$; when *Challenger* was launched on January 28, the temperature was a frigid $31°$.

These data have been analyzed extensively (Dalal et al., 1989, Lavine, 1991). Tufte (1997) gives a thorough and convincing visual analysis of the evidence available prior to the launch. We consider statistical analysis of these data in Chapter 7, Example 7.4.

But, what if the engineers had simply made a better graph? At the very least, that would entail (a) drawing a smoothed curve to fit the points (to show the trend), and (b) removing the background grid lines (which obscure the data). Figure 1.14 shows a revised version of the same graph, highlighting the non-zero observations and adding a simple quadratic curve to allow for a possible nonlinear relationship. For comparison, the excluded zero observations are also shown in grey. This plot, even showing only the non-zero points, should have caused any engineer to conclude that either: (a) the data were wrong, or (b) there were excessive risks associated with both high and low temperatures. But it is well-known that brittleness of the rubber used in the O-rings is inversely proportional to temperature cubed, so prudent interest might have focussed on the first possibility.[9]

[9]A coda to this story shows the role of visual explanation in practice as well (Tufte, 1997, pp. 50–53). The Rogers Commission contracted the reknowned theoretical physicist Richard Feynman to contribute to their investigation. He determined that the most probable cause of the shuttle failure was the lack of resiliancy of the rubber O-rings at low temperature. But how could he make this point convincingly? At a televised public hearing, he took a piece of the O-ring material, squeezed it in a C-clamp, and plunged it into a glass of ice water. After a few minutes, he released the clamp, and the rubber did not spring back to shape. He mildly said, "... there is no resilience in this particular material when it is at a temperature of 32 degrees. I believe this has some significance for our problem" (Feynman, 1988).

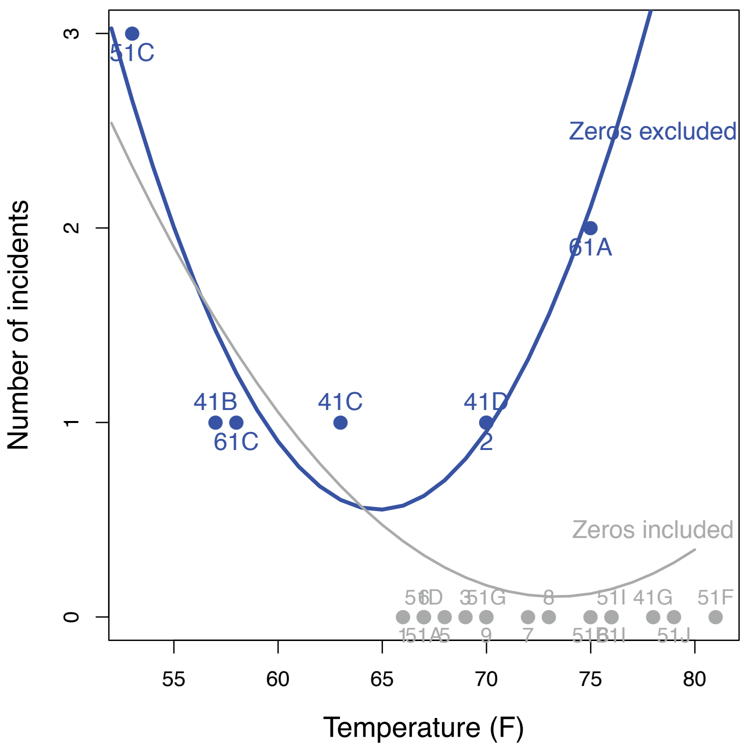


Figure 1.14: Re-drawn version of the NASA pre-launch graph, showing the locations of the excluded observations and with fitted quadratics for both sets of observations.

△

1.4.6 Data plots, model plots, and data+model plots

In this book, we use hundreds of graphs to illustrate aspects of discrete data, methods of analysis, and plots for understanding and explaining results. In addition to the overview of goals and design principles (Section 1.4.1) shown in Figure 1.6, another classification of such graphs is useful to bear in mind as you read this book. We distinguish three kinds of plots:

- **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 simplified 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.

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.

[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.

[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`).

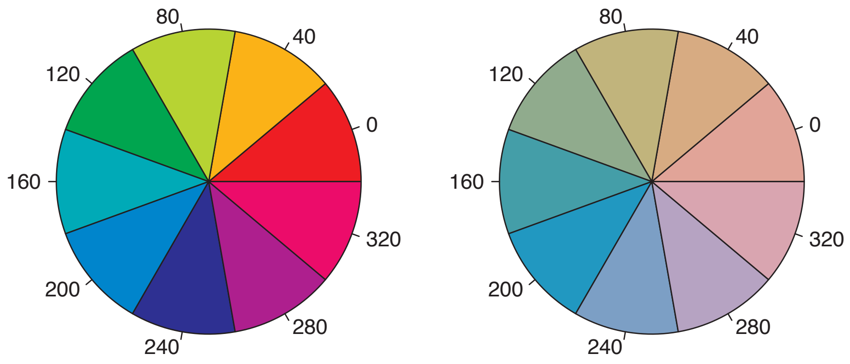


Figure 1.15: Qualitative color palette for the HSV (left) and HCL (right) spaces. The HSV colors are $(H, 100, 100)$ and the HCL colors $(H, 50, 70)$ for the same hues H. Note that in a monochrome version of this page, all pie sectors in the right wheel will be shaded with the same gray, i.e., they will appear to be virtually identical.

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.

1.5 Chapter summary

- Categorical data differs from quantitative data because the variables take on discrete values (ordered or unordered, character or numeric) rather than continuous numerical values. Consequently, such data often appear in aggregated form representing category frequencies or in tables.

- Data analysis methods for categorical data are comprised of those concerned mainly with testing particular hypotheses versus those that fit statistical models. Model building methods have the advantages of providing parameter estimates and model-predicted values, along with measures of uncertainty (standard errors).

- Graphical methods can serve different purposes for different goals (data analysis versus presentation), and these suggest different design principles that a graphic should respect to achieve a given communication goal.

- For categorical data, some graphic forms (bar charts, line graphs, scatterplots) used for quantitative data can be readily adapted to discrete variables. However, frequency data often requires novel graphics using area and other visual attributes.

- Graphics can be far more effective when categorical variables are ordered to facilitate comparison of the effects to be seen and rendered to facilitate detection of patterns, trends or anomalies.

- The visualization approach to data analysis often entails a sequence of intertwined steps involving graphing and model fitting.

- Producing effective graphs for presentation is often hard work, requiring attention to details that support or detract from your communication goal.

1.6 Lab exercises

Exercise 1.1 A web page, “The top ten worst graphs, ” `http://www.biostat.wisc.edu/~kbroman/topten_worstgraphs/` by Karl Broman lists his picks for the worst graphs (and a table) that have appeared in the statistical and scientific literature. Each entry links to graph(s) and a brief discussion of what is wrong and how it could be improved.

(a) Examine a number of recent issues of a scientific or statistical journal in which you have some interest. Find one or more examples of a graph or table that is a particularly bad use of display material to summarize and communicate research findings. Write a few sentences indicating how or why the display fails and how it could be improved.
(b) Do the same task for some popular magazine or newspaper that uses data displays to supplement the text for some story. Again, write a few sentences describing why the display is bad and how it could be improved.

Exercise 1.2 As in the previous exercise, examine the literature in recent issues of some journal of interest to you. Find one or more examples of a graph or table that you feel does a *good* job of summarizing and communicating research findings.

(a) Write a few sentences describing why you chose these displays.
(b) Now take the role of a tough journal reviewer. Are there any features of the display that could be modified to make them more effective?

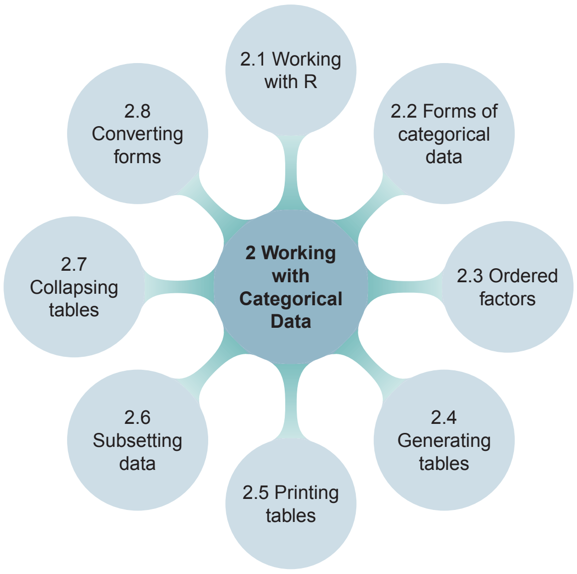
Exercise 1.3 Infographics are another form of visual displays, quite different from the data graphics featured in this book, but often based on some data or analysis. Do a Google image search for the topic “Global warming” to see a rich collection.

(a) Find and study one or two images that attempt some visual explanation of causes and/or effects of global warming. Describe the main message in a sentence or two.
(b) What visual and graphic features are used in these to convey the message?

Exercise 1.4 The Wikipedia web page `en.wikipedia.org/wiki/Portal:Global_warming` gives a few data-based graphics on the topic of global warming. Read the text and study the graphs.

(a) Write a short figure title for each that would announce the conclusion to be drawn in a presentation graphic.
(b) Write a figure caption for each that would explain what is shown and the important graphical details for a reader to understand.

Exercise 1.5 The R Graph Gallery, `http://rgraphgallery.blogspot.com/`, contains a large collection of examples of graphs in R, tagged by type or content, together with the R code to produce them. Explore this collection for the terms (a) association plot (b) bar chart (c) categorical data (d) fluctuation diagram (e) mosaic plot Find one or two you particularly like and write a few sentences saying why you do.



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.

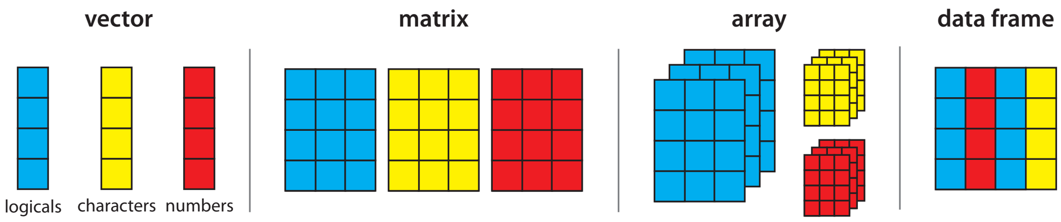


Figure 2.1: Principal data structures and data types in R. Colors represent different data types: numeric, character, logical.

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 `()`.

```
> count <- c(17, 20, 15, 40)                          # assign
> count                                                # print

[1] 17 20 15 40

> (sex <- c("female", "male", "female", "male"))    # both

[1] "female" "male"   "female" "male"

> (passed <- c(TRUE, TRUE, FALSE, FALSE))

[1]  TRUE  TRUE FALSE FALSE
```

Other useful functions for creating vectors are:

- The `:` operator for generating consecutive integer sequences, e.g., `1:10` gives the integers 1 to 10. The `seq()` function is more general, taking the forms `seq(from, to)`, `seq(from, to, by= )`, and `seq(from, to, length.out= )` where the optional argument `by` specifies the interval between adjacent values and `length.out` gives the desired length of the result.
- The `rep()` function generates repeated sequences, replicating its first argument (which may be a vector) a given number of `times`, and individual elements can be repeated with `each` until an optional `length.out` is obtained.

```
> seq(10, 100, by = 10)          # give interval

 [1]  10  20  30  40  50  60  70  80  90 100

> seq(0, 1, length.out = 11)     # give length

 [1] 0.0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0

> (sex <- rep(c("female", "male"), times = 2))

[1] "female" "male"   "female" "male"

> (sex <- rep(c("female", "male"), length.out = 4))   # same

[1] "female" "male"   "female" "male"

> (passed <- rep(c(TRUE, FALSE), each = 2))

[1]  TRUE  TRUE FALSE FALSE
```

2.1.2 Matrices

A **matrix** is a two-dimensional array of elements of the same type composed in a rectangular array of rows and columns. Matrices can be created by the function `matrix(values, nrow, ncol)`, which reshapes the elements in the first argument (`values`) to a matrix with `nrow` rows and `ncol` columns. By default, the elements are filled in columnwise, unless the optional argument `byrow = TRUE` is given.

```
> (matA <- matrix(1:8, nrow = 2, ncol = 4))

     [,1] [,2] [,3] [,4]
[1,]    1    3    5    7
[2,]    2    4    6    8
```

```
> (matB <- matrix(1:8, nrow = 2, ncol = 4, byrow = TRUE))

     [,1] [,2] [,3] [,4]
[1,]    1    2    3    4
[2,]    5    6    7    8

> (matC <- matrix(1:4, nrow = 2, ncol = 4))

     [,1] [,2] [,3] [,4]
[1,]    1    3    1    3
[2,]    2    4    2    4
```

The last example illustrates that the values in the first argument are recycled as necessary to fill the given number of rows and columns.

All matrices have a dimensions attribute, a vector of length two giving the number of rows and columns, retrieved with the function `dim()`. Labels for the rows and columns can be assigned using `dimnames()`,[2] which takes a list of two vectors for the row names and column names, respectively. To see the structure of a matrix (or any other R object) and its attributes, you can use the `str()` function, as shown in the example below.

```
> dim(matA)

[1] 2 4

> str(matA)

 int [1:2, 1:4] 1 2 3 4 5 6 7 8

> dimnames(matA) <- list(c("M", "F"), LETTERS[1:4])
> matA

  A B C D
M 1 3 5 7
F 2 4 6 8

> str(matA)

 int [1:2, 1:4] 1 2 3 4 5 6 7 8
 - attr(*, "dimnames")=List of 2
  ..$ : chr [1:2] "M" "F"
  ..$ : chr [1:4] "A" "B" "C" "D"
```

Additionally, names for the row and column *variables* themselves can also be assigned in the `dimnames` call by giving each dimension vector a name.

```
> dimnames(matA) <- list(sex = c("M", "F"), group = LETTERS[1:4])
> ## or: names(dimnames(matA)) <- c("Sex", "Group")
> matA

   group
sex A B C D
  M 1 3 5 7
  F 2 4 6 8

> str(matA)

 int [1:2, 1:4] 1 2 3 4 5 6 7 8
 - attr(*, "dimnames")=List of 2
  ..$ sex  : chr [1:2] "M" "F"
  ..$ group: chr [1:4] "A" "B" "C" "D"
```

[2]The `dimnames` can also be specified as an optional argument to `matrix()`.

(`LETTERS` is a predefined character vector of the 26 uppercase letters). Matrices can also be created or enlarged by "binding" vectors or matrices together by rows or columns:

- `rbind(a, b, c)` creates a matrix with the vectors `a`, `b`, and `c` as its rows, recycling the elements as necessary to the length of the longest one.
- `cbind(a, b, c)` creates a matrix with the vectors `a`, `b`, and `c` as its columns.
- `rbind(mat, a, b, ...)` and `cbind(mat, a, b, ...)` add additional rows (columns) to a matrix `mat`, recycling or subsetting the elements in the vectors to conform with the size of the matrix.

```
> rbind(matA, c(10, 20))

   A  B  C  D
M  1  3  5  7
F  2  4  6  8
  10 20 10 20

> cbind(matA, c(10, 20))

  A B C D
M 1 3 5 7 10
F 2 4 6 8 20
```

Rows and columns can be swapped (transposed) using `t()`:

```
> t(matA)

     sex
group M F
    A 1 2
    B 3 4
    C 5 6
    D 7 8
```

Finally, we note that basic computations involving matrices are performed *element-wise*:

```
> 2 * matA / 100

   group
sex    A    B    C    D
  M 0.02 0.06 0.10 0.14
  F 0.04 0.08 0.12 0.16
```

Special operators and functions do exist for matrix operations, such as `%*%` for the matrix product.

2.1.3 Arrays

Higher-dimensional arrays are less frequently encountered in traditional data analysis, but they are of great use for categorical data, where frequency tables of three or more variables can be naturally represented as arrays, with one dimension for each table variable.

The function `array(values, dim)` takes the elements in `values` and reshapes these into an array whose dimensions are given in the vector `dim`. The number of dimensions is the length of `dim`. As with matrices, the elements are filled in with the first dimension (rows) varying most rapidly, then by the second dimension (columns) and so on for all further dimensions, which can be considered as layers. A matrix is just the special case of an array with two dimensions.

```
> dims <-  c(2, 4, 2)
> (arrayA <- array(1:16, dim = dims))      # 2 rows, 4 columns, 2 layers

, , 1

     [,1] [,2] [,3] [,4]
[1,]    1    3    5    7
[2,]    2    4    6    8

, , 2

     [,1] [,2] [,3] [,4]
[1,]    9   11   13   15
[2,]   10   12   14   16

> str(arrayA)

 int [1:2, 1:4, 1:2] 1 2 3 4 5 6 7 8 9 10 ...

> (arrayB <- array(1:16, dim = c(2, 8)))    # 2 rows, 8 columns

     [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8]
[1,]    1    3    5    7    9   11   13   15
[2,]    2    4    6    8   10   12   14   16

> str(arrayB)

 int [1:2, 1:8] 1 2 3 4 5 6 7 8 9 10 ...
```

In the same way that we can assign labels to the rows and columns in matrices, we can assign these attributes to `dimnames(arrayA)`, or include this information in a `dimnames=` argument to `array()`.

```
> dimnames(arrayA) <- list(sex = c("M", "F"),
+                          group = letters[1:4],
+                          time = c("Pre", "Post"))
> arrayA

, , time = Pre

   group
sex a b c d
  M 1 3 5 7
  F 2 4 6 8

, , time = Post

   group
sex  a  b  c  d
  M  9 11 13 15
  F 10 12 14 16

> str(arrayA)

 int [1:2, 1:4, 1:2] 1 2 3 4 5 6 7 8 9 10 ...
 - attr(*, "dimnames")=List of 3
  ..$ sex  : chr [1:2] "M" "F"
  ..$ group: chr [1:4] "a" "b" "c" "d"
  ..$ time : chr [1:2] "Pre" "Post"
```

Arrays in R can contain any single type of elements— numbers, character strings, logicals. R also has a variety of functions (e.g., `table()`, `xtabs()`) for creating and manipulating "table" 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 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 `c("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

> GSS <- data.frame(tmp, count = c(279, 165, 73, 47, 225, 191))
> GSS

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

> names(GSS)

[1] "sex"   "party" "count"

> str(GSS)

'data.frame': 6 obs. of  3 variables:
 $ sex  : Factor w/ 2 levels "female","male": 1 2 1 2 1 2
 $ party: Factor w/ 3 levels "dem","indep",..: 1 1 2 2 3 3
 $ count: num  279 165 73 47 225 191

> sum(GSS$count)

[1] 980
```

The last line above shows that there are 980 cases represented in the frequency table. △

2.2.3 Table form

Table form data is represented as a matrix, array, or table object whose elements are the frequencies in an n-way table. The number of dimensions of the table is the length, `length(dim(X))`, of its `dim` (or `dimnames`) attribute, and the sizes of the dimensions in the table are the elements of `dim(X)`. The total number of observations represented is the sum of all the frequencies, `sum(X)`.

Example 2.4: Hair color and eye color

A classic data set on frequencies of hair color, eye color, and sex is given in table form in *`HairEyeColor`* in the datasets package, reporting the frequencies of these categories for 592 students in a statistics course.

```
> data("HairEyeColor", package = "datasets")    # load the data
> str(HairEyeColor)                            # show the structure

 table [1:4, 1:4, 1:2] 32 53 10 3 11 50 10 30 10 25 ...
 - attr(*, "dimnames")=List of 3
  ..$ Hair: chr [1:4] "Black" "Brown" "Red" "Blond"
  ..$ Eye : chr [1:4] "Brown" "Blue" "Hazel" "Green"
  ..$ Sex : chr [1:2] "Male" "Female"

> dim(HairEyeColor)                            # table dimension sizes

[1] 4 4 2

> dimnames(HairEyeColor)                       # variable and level names
```

```
$Hair
[1] "Black" "Brown" "Red"   "Blond"

$Eye
[1] "Brown" "Blue"  "Hazel" "Green"

$Sex
[1] "Male"   "Female"

> sum(HairEyeColor)                  # number of cases

[1] 592
```

Three-way (and higher-way) tables can be printed in a more convenient form using `structable()` and `ftable()` as described below in Section 2.5. △

Tables are often created from raw data in case form or frequency form using the functions `table()` and `xtabs()` described in Section 2.4. For smallish frequency tables that are already in tabular form, you can enter the frequencies in a matrix, and then assign `dimnames` and other attributes.

To illustrate, we create the GSS data as a table below, entering the values in the table by rows (`byrow=TRUE`), as they appear in printed form.

```
> GSS.tab <- matrix(c(279, 73, 225,
+                    165, 47, 191),
+                  nrow = 2, ncol = 3, byrow = TRUE)
> dimnames(GSS.tab) <- list(sex = c("female", "male"),
+                          party = c("dem", "indep", "rep"))
> GSS.tab

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

`GSS.tab` is a matrix, not an object of `class("table")`, and some functions are happier with tables than matrices.[5] You should therefore coerce it to a table with `as.table()`,

```
> GSS.tab <- as.table(GSS.tab)
> str(GSS.tab)

 table [1:2, 1:3] 279 165 73 47 225 191
 - attr(*, "dimnames")=List of 2
  ..$ sex  : chr [1:2] "female" "male"
  ..$ party: chr [1:3] "dem" "indep" "rep"
```

EXAMPLE 2.5: Job satisfaction

Here is another similar example, entering data on job satisfaction classified by `income` and level of `satisfaction` from a 4×4 table given by Agresti (2002, Table 2.8, p. 57).

```
> ## A 4 x 4 table  Agresti (2002, Table 2.8, p. 57) Job Satisfaction
> JobSat <- matrix(c(1, 2, 1, 0,
+                   3, 3, 6, 1,
+                   10, 10, 14, 9,
+                   6, 7, 12, 11),
```

[5]There are quite a few functions in R with specialized methods for "table" objects. For example, `plot(GSS.tab)` gives a mosaic plot and `barchart(GSS.tab)` gives a divided bar chart.

```
+                    nrow = 4, ncol = 4)
> dimnames(JobSat) <-
+   list(income = c("< 15k", "15-25k", "25-40k", "> 40k"),
+         satisfaction = c("VeryD", "LittleD", "ModerateS", "VeryS"))
> JobSat <- as.table(JobSat)
> JobSat

        satisfaction
income    VeryD LittleD ModerateS VeryS
  < 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"
```

Finally, there are situations where, particularly for display purposes, you want to re-order the *dimensions* of an n-way table, and/or change the labels for the variables or levels. This is easy when the data are in table form: `aperm()` permutes the dimensions, and assigning to `names` and `dimnames` changes variable names and level labels, respectively.

```
> str(UCBAdmissions)

 table [1:2, 1:2, 1:6] 512 313 89 19 353 207 17 8 120 205 ...
 - attr(*, "dimnames")=List of 3
  ..$ Admit : chr [1:2] "Admitted" "Rejected"
  ..$ Gender: chr [1:2] "Male" "Female"
  ..$ Dept  : chr [1:6] "A" "B" "C" "D" ...

> # vary along the 2nd, 1st, and 3rd dimension in UCBAdmissions
> UCB <- aperm(UCBAdmissions, c(2, 1, 3))
> dimnames(UCB)$Admit <- c("Yes", "No")
> names(dimnames(UCB)) <- c("Sex", "Admitted", "Department")
> str(UCB)

 table [1:2, 1:2, 1:6] 512 89 313 19 353 17 207 8 120 202 ...
 - attr(*, "dimnames")=List of 3
  ..$ Sex       : chr [1:2] "Male" "Female"
  ..$ Admitted  : chr [1:2] "Yes" "No"
  ..$ Department: chr [1:6] "A" "B" "C" "D" ...
```

2.4 Generating tables with `table()` and `xtabs()`

With data in case form or frequency form, you can generate frequency tables from factor variables in data frames using the `table()` function; for tables of proportions, use the `prop.table()` function, and for marginal frequencies (summing over some variables) use `margin.table()`. The examples below use the same case-form data frame `mydata` used earlier (Section 2.1.4).

```
> 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)
```

2.4.1 `table()`

`table(...)` takes a list of variables interpreted as factors, or a data frame whose columns are so interpreted. It does not take a `data=` argument, so either supply the names of columns in the data frame (possibly using `with()` for convenience), or select the variables using column indexes:

```
> # 2-Way Frequency Table
> table(mydata$A, mydata$B)          # A will be rows, B will be columns


     b1 b2
  a1 18 30
  a2 22 30

> ## same: with(mydata, table(A, B))
> (mytab <- table(mydata[,1:2]))      # same

    B
A    b1 b2
  a1 18 30
  a2 22 30
```

We can use `margin.table(X, margin)` to sum a table `X` for the indices in `margin`, i.e., over the dimensions not included in `margin`. A related function is `addmargins(X, margin, FUN = sum)`, which extends the dimensions of a table or array with the marginal values calculated by `FUN`.

```
> margin.table(mytab)        # sum over A & B

[1] 100

> margin.table(mytab, 1)     # A frequencies (summed over B)

A
a1 a2
48 52

> margin.table(mytab, 2)     # B frequencies (summed over A)

B
b1 b2
40 60

> addmargins(mytab)          # show all marginal totals

     B
A       b1  b2 Sum
  a1    18  30  48
  a2    22  30  52
  Sum   40  60 100
```

The function `prop.table()` expresses the table entries as a fraction of a given marginal table.

```
> prop.table(mytab)          # cell proportions

    B
A       b1   b2
  a1 0.18 0.30
  a2 0.22 0.30

> prop.table(mytab, 1)       # row proportions

    B
A          b1      b2
  a1 0.37500 0.62500
  a2 0.42308 0.57692
```

```
> prop.table(mytab, 2)      # column proportions

    B
A       b1   b2
  a1 0.45 0.50
  a2 0.55 0.50
```

`table()` can also generate multidimensional tables based on 3 or more categorical variables. In this case, use the `ftable()` or `structable()` function to print the results more attractively as a "flat" (2-way) table.

```
> # 3-Way Frequency Table
> mytab <- table(mydata[,c("A", "B", "sex")])
> ftable(mytab)

      sex  F  M
A  B
a1 b1      9  9
   b2     15 15
a2 b1     12 10
   b2     19 11
```

`table()` ignores missing values by default, but has optional arguments `useNA` and `exclude` that can be used to control this. See `help(table)` for the details.

2.4.2 `xtabs()`

The `xtabs()` function allows you to create cross tabulations of data using formula style input. This typically works with case-form or frequency-form data supplied in a data frame or a matrix. The result is a contingency table in array format, whose dimensions are determined by the terms on the right side of the formula. As shown below, the `summary` method for tables produces a simple χ^2 test of independence of all factors, and indicates the number of cases and dimensions.

```
> # 3-Way Frequency Table
> mytable <- xtabs(~ A + B + sex, data = mydata)
> ftable(mytable)     # print table

      sex  F  M
A  B
a1 b1      9  9
   b2     15 15
a2 b1     12 10
   b2     19 11

> summary(mytable)    # chi-squared test of independence

Call: xtabs(formula = ~A + B + sex, data = mydata)
Number of cases in table: 100
Number of factors: 3
Test for independence of all factors:
Chisq = 1.54, df = 4, p-value = 0.82
```

When the data have already been tabulated in frequency form, include the frequency variable (usually `count` or `Freq`) on the left side of the formula, as shown in the example below for the GSS data.

```
> (GSStab <- xtabs(count ~ sex + party, data = GSS))
```

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

> summary(GSStab)

Call: xtabs(formula = count ~ sex + party, data = GSS)
Number of cases in table: 980
Number of factors: 2
Test for independence of all factors:
Chisq = 7, df = 2, p-value = 0.03
```

For "table" objects, the `plot` method produces basic mosaic plots using the `mosaicplot()` function from the graphics package.

2.5 Printing tables with `structable()` and `ftable()`

2.5.1 Text output

For 3-way and larger tables, the functions `ftable()` (in the stats package) and `structable()` (in vcd) provide a convenient and flexible tabular display in a "flat" (2-way) format.

With `ftable(X, row.vars=, col.vars=)`, variables assigned to the rows and/or columns of the result can be specified as the integer numbers or character names of the variables in the array `X`. By default, the last variable is used for the columns. The formula method, in the form `ftable(colvars ~ rowvars, data)` allows a formula where the left- and right-hand side of formula specify the column and row variables, respectively.

```
>  ftable(UCB)                          # default

                Department   A   B   C   D   E   F
Sex    Admitted
Male   Yes                 512 353 120 138  53  22
       No                  313 207 205 279 138 351
Female Yes                  89  17 202 131  94  24
       No                   19   8 391 244 299 317

> #ftable(UCB, row.vars = 1:2)       # same result
>  ftable(Admitted + Sex ~ Department, data = UCB)    # formula method

           Admitted  Yes          No
           Sex      Male Female Male Female
Department
A                    512     89  313     19
B                    353     17  207      8
C                    120    202  205    391
D                    138    131  279    244
E                     53     94  138    299
F                     22     24  351    317
```

The `structable()` function is similar, but more general, and uses recursive splits in the vertical or horizontal directions (similar to the construction of mosaic displays). It works with both data frames and table objects.

```
> library(vcd)
> structable(HairEyeColor)                      # show the table: default

             Eye Brown Blue Hazel Green
```

```
Hair  Sex
Black Male          32   11    10     3
      Female        36    9     5     2
Brown Male          53   50    25    15
      Female        66   34    29    14
Red   Male          10   10     7     7
      Female        16    7     7     7
Blond Male           3   30     5     8
      Female         4   64     5     8

> structable(Hair + Sex ~ Eye, HairEyeColor) # specify col ~ row variables

      Hair Black        Brown          Red         Blond
      Sex    Male Female  Male Female Male Female  Male Female
Eye
Brown          32     36    53     66   10     16     3      4
Blue           11      9    50     34   10      7    30     64
Hazel          10      5    25     29    7      7     5      5
Green           3      2    15     14    7      7     8      8
```

It also returns an object of class `"structable"` for which there are a variety of special methods. For example, the transpose function `t()` interchanges rows and columns, so that a call like `t(structable(HairEyeColor))` produces the second result shown just above. There are also plot methods: for example, `plot()` produces mosaic plots from the vcd package.

2.6 Subsetting data

Often, the analysis of some data set is focused on a subset only. For example, the *HairEyeColor* data set introduced above tabulates frequencies of hair and eye colors for male and female students—the analysis could concentrate on one group only, or compare both groups in a stratified analysis. This section deals with extracting subsets of data in tables, structables, or data frames.

2.6.1 Subsetting tables

If data are available in tabular form created with `table()` or `xtabs()`, resulting in `table` objects, subsetting is done via indexing, either with integers or character strings corresponding to the factor levels. The following code extracts the female data from the `HairEyeColor` data set:

```
> HairEyeColor[,,"Female"]

       Eye
Hair    Brown Blue Hazel Green
  Black    36    9     5     2
  Brown    66   34    29    14
  Red      16    7     7     7
  Blond     4   64     5     8

> ##same using index: HairEyeColor[,,2]
```

Empty indices stand for taking all data of the corresponding dimension. The third one (Sex) is fixed at the second (“`Female`”) level. Note that in this case, the dimensionality is reduced to a two-way table, since dimensions with only one level are dropped by default. Functions like `apply()` can iterate through all levels of one or several dimensions and apply a function to each subset. The following calculates the total amount of male and female students:

```
> apply(HairEyeColor, 3, sum)

  Male Female 
   279    313 
```

It is of course possible to select more than one level:

```
> HairEyeColor[c("Black", "Brown"), c("Hazel", "Green"),]

, , Sex = Male

       Eye
Hair     Hazel Green
  Black     10     3
  Brown     25    15

, , Sex = Female

       Eye
Hair     Hazel Green
  Black      5     2
  Brown     29    14
```

2.6.2 Subsetting structables

Structables work in a similar way, but take into account the hierarchical structure imposed by the "flattened" format, and also distinguish explicitly between subsetting levels and subsetting tables. In the following example, compare the different effects of applying the `[` and `[[` operators to the structable:

```
> hec <- structable(Eye ~ Sex + Hair, data = HairEyeColor)
> hec

             Eye Brown Blue Hazel Green
Sex    Hair                            
Male   Black        32   11    10     3
       Brown        53   50    25    15
       Red          10   10     7     7
       Blond         3   30     5     8
Female Black        36    9     5     2
       Brown        66   34    29    14
       Red          16    7     7     7
       Blond         4   64     5     8

> hec["Male",]

           Eye Brown Blue Hazel Green
Sex  Hair                            
Male Black        32   11    10     3
     Brown        53   50    25    15
     Red          10   10     7     7
     Blond         3   30     5     8

> hec[["Male",]]

      Eye Brown Blue Hazel Green
Hair                            
Black        32   11    10     3
Brown        53   50    25    15
Red          10   10     7     7
Blond         3   30     5     8
```

The first form keeps the dimensionality, whereas the second conditions on the "Male" level and returns the corresponding subtable. The following does this twice, once for Sex, and once for Hair (restricted to the Male level):

```
> hec[[c("Male", "Brown"),]]

Eye Brown Blue Hazel Green

       53    50     25     15
```

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 analyze, 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 that are not shown in the plots.

The appropriate tools to use again depend on the form in which the data are represented—a case-form data frame, a frequency-form data frame (aggregate()), or a table-form array or table object (margin.table() or apply()).

When the data are in frequency form, and we want to produce another frequency data frame, aggregate() is a handy tool, using the argument FUN = sum to sum the frequency variable over the factors *not* mentioned in the formula.

EXAMPLE 2.6: Dayton survey

The data frame *DaytonSurvey* in the vcdExtra package represents a 2^5 table giving the frequencies of reported use ("ever used?") of alcohol, cigarettes, and marijuana in a sample of 2276 high school seniors, also classified by sex and race.

```
> data("DaytonSurvey", package = "vcdExtra")
> str(DaytonSurvey)

'data.frame': 32 obs. of  6 variables:
 $ cigarette: Factor w/ 2 levels "Yes","No": 1 2 1 2 1 2 1 2 1 2 ...
 $ alcohol  : Factor w/ 2 levels "Yes","No": 1 1 2 2 1 1 2 2 1 1 ...
 $ marijuana: Factor w/ 2 levels "Yes","No": 1 1 1 1 2 2 2 2 1 1 ...
 $ sex      : Factor w/ 2 levels "female","male": 1 1 1 1 1 1 1 1 2 2 ...
 $ race     : Factor w/ 2 levels "white","other": 1 1 1 1 1 1 1 1 1 1 ...
 $ Freq     : num  405 13 1 1 268 218 17 117 453 28 ...

> head(DaytonSurvey)

  cigarette alcohol marijuana    sex  race Freq
1       Yes     Yes       Yes female white  405
2        No     Yes       Yes female white   13
3       Yes      No       Yes female white    1
4        No      No       Yes female white    1
5       Yes     Yes        No female white  268
6        No     Yes        No female white  218
```

To focus on the associations among the substances, we want to collapse over sex and race. The right-hand side of the formula used in the call to aggregate() gives the factors to be retained in the new frequency data frame, Dayton_ACM_df. The left-hand side is the frequency variable (Freq), and we aggregate using the FUN = sum.

```
> # data in frequency form: collapse over sex and race
> Dayton_ACM_df <- aggregate(Freq ~ cigarette + alcohol + marijuana,
+                            data = DaytonSurvey, FUN = sum)
> Dayton_ACM_df

  cigarette alcohol marijuana Freq
1       Yes     Yes       Yes  911
2        No     Yes       Yes   44
3       Yes      No       Yes    3
```

```
4        No       No        Yes    2
5       Yes      Yes         No  538
6        No      Yes         No  456
7       Yes       No         No   43
8        No       No         No  279
```

△

When the data are in table form, and we want to produce another table, `apply()` with `FUN = sum` can be used in a similar way to sum the table over dimensions not mentioned in the `MARGIN` argument. `margin.table()` is just a wrapper for `apply()` using the `sum()` function.

EXAMPLE 2.7: Dayton survey

To illustrate, we first convert the *DaytonSurvey* to a 5-way table using `xtabs()`, giving `Dayton_tab`.

```
> # convert to table form
> Dayton_tab <- xtabs(Freq ~ cigarette + alcohol + marijuana + sex + race,
+                     data = DaytonSurvey)
> structable(cigarette + alcohol + marijuana ~ sex + race,
+            data = Dayton_tab)

             cigarette Yes                 No
             alcohol   Yes      No      Yes      No
             marijuana Yes  No Yes  No Yes  No Yes  No
sex    race
female white           405 268   1  17  13 218   1 117
       other            23  23   0   1   2  19   0  12
male   white           453 228   1  17  28 201   1 133
       other            30  19   1   8   1  18   0  17
```

Then, use `apply()` on `Dayton_tab` to give the 3-way table `Dayton_ACM_tab` summed over sex and race. The elements in this new table are the column sums for `Dayton.tab` shown by `structable()` just above.

```
> # collapse over sex and race
> Dayton_ACM_tab <- apply(Dayton_tab, MARGIN = 1:3, FUN = sum)
> Dayton_ACM_tab <- margin.table(Dayton_tab, 1:3)   # same result
> structable(cigarette + alcohol ~ marijuana, data = Dayton_ACM_tab)

          cigarette Yes      No
          alcohol   Yes  No Yes  No
marijuana
Yes                 911   3  44   2
No                  538  43 456 279
```

△

(Note that `structable()` would do the collapsing job for us anyway.)

Many of these operations can be performed using the `**ply()` functions in the plyr (Wickham, 2014a) package. For example, with the data in a frequency form data frame, use `ddply()` to collapse over unmentioned factors, and `summarise()` as the function to be applied to each piece.

```
> library(plyr)
> Dayton_ACM_df <- ddply(DaytonSurvey, .(cigarette, alcohol, marijuana),
+                        summarise, Freq = sum(Freq))
```

2.7.2 Collapsing table levels: `collapse.table()`

A related problem arises when we have a table or array and for some purpose we want to reduce the number of levels of some factors by summing subsets of the frequencies. For example, we may have initially coded Age in 10-year intervals, and decide that, either for analysis or display purposes, we want to reduce Age to 20-year intervals. The `collapse.table()` function in vcdExtra was designed for this purpose.

EXAMPLE 2.8: Collapsing categories

Create a 3-way table, and collapse Age from 10-year to 20-year intervals and Education from three levels to two. To illustrate, we first generate a $2 \times 6 \times 3$ table of random counts from a Poisson distribution with mean of 100, with factors `sex`, `age`, and `education`.

```
> # 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.

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

	To this		
From 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)`	—

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.

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.

[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".

```
> 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
```

Then, `weighted.mean()` works as we would like:

```
> weighted.mean(horse.df$nDeaths, weights=horse.df$Freq)

[1] 2
```

△

2.8.2 Case form to table form

Going the other way, we use `table()` to convert from case form to table form.

EXAMPLE 2.11: Arthritis treatment

Convert the *`Arthritis`* data in case form to a 3-way table of `Treatment` × `Sex` × `Improved`. We select the desired columns with their names, but could also use column numbers, e.g., `table(Arthritis[,c(2,3,5)])`.

```
> Art.tab <- table(Arthritis[,c("Treatment", "Sex", "Improved")])
> str(Art.tab)

 'table' int [1:2, 1:2, 1:3] 19 6 10 7 7 5 0 2 6 16 ...
 - attr(*, "dimnames")=List of 3
  ..$ Treatment: chr [1:2] "Placebo" "Treated"
  ..$ Sex      : chr [1:2] "Female" "Male"
  ..$ Improved : chr [1:3] "None" "Some" "Marked"

> ftable(Art.tab)

                 Improved None Some Marked
Treatment Sex
Placebo   Female            19    7      6
          Male              10    0      1
Treated   Female             6    5     16
          Male               7    2      5
```

△

2.8.3 Table form to case form

There may also be times that you will need an equivalent case form data frame with factors representing the table variables rather than the frequency table. For example, the `mca()` function in package MASS (for multiple correspondence analysis) only operates on data in this format. The function `expand.dft()`[7] in vcdExtra does this, converting a table into a case form.

Example 2.12: Arthritis treatment
Convert the *Arthritis* data in table form (`Art.tab`) back to a `data.frame` in case form, with factors `Treatment`, `Sex`, and `Improved`.

```
> library(vcdExtra)
> Art.df <- expand.dft(Art.tab)
> str(Art.df)

'data.frame':  84 obs. of  3 variables:
 $ Treatment: Factor w/ 2 levels "Placebo","Treated": 1 1 1 1 1 1 1 1 1 1 ...
 $ Sex      : Factor w/ 2 levels "Female","Male": 1 1 1 1 1 1 1 1 1 1 ...
 $ Improved : Factor w/ 3 levels "Marked","None",..: 2 2 2 2 2 2 2 2 2 2 ...
```

△

2.8.4 Publishing tables to LaTeX or HTML

OK, you've read your data into R, done some analysis, and now want to include some tables in a LaTeX document or in a web page in HTML format. Formatting tables for these purposes is often tedious and error-prone.

There are a great many packages in R that provide for nicely formatted, publishable tables for a wide variety of purposes; indeed, most of the tables in this book are generated using these tools. See Leifeld (2013) for a description of the texreg (Leifeld, 2014) package and a comparison with some of the other packages.

Here, we simply illustrate the xtable (Dahl, 2014) package, which, along with capabilities for statistical model summaries, time-series data, and so forth, has a `xtable.table` method for one-way and two-way table objects.

The *HorseKicks* data is a small one-way frequency table described in Example 3.4 and contains the frequencies of 0, 1, 2, 3, 4 deaths per corps-year by horse-kick among soldiers in 20 corps in the Prussian army.

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

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

By default, `xtable()` formats this in LaTeX as a vertical table, and prints the LaTeX markup to the R console. This output is shown below.

```
> library(xtable)
> xtable(HorseKicks)

% latex table generated in R 3.2.1 by xtable 1.8-0 package
% Thu Nov 12 13:05:37 2015
\begin{table}[ht]
\centering
\begin{tabular}{rr}
  \hline
 & nDeaths \\
  \hline
0 & 109 \\
  1 &  65 \\
```

[7]The original code for this function was provided by Marc Schwarz on the R-Help mailing list.

```
  2 &  22 \\
  3 &   3 \\
  4 &   1 \\
   \hline
\end{tabular}
\end{table}
```

When this is rendered in a LaTeX document, the result of `xtable()` appears as shown in the table below.

```
> xtable(HorseKicks)
```

	nDeaths
0	109
1	65
2	22
3	3
4	1

The table above isn't quite right, because the column label "nDeaths" belongs to the first column, and the second column should be labeled "Freq." To correct that, we convert the *`HorseKicks`* table to a data frame (see Section 2.8 for details), add the appropriate `colnames`, and use the `print.xtable` method to supply some other options.

```
> 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, giving the version shown in Table 2.2: (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; (e) 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, label="tab:xtable5",
+       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.

2.9 A complex example: TV viewing data*

If you have followed so far, congratulations! You are ready for a more complicated example that puts together a variety of the skills developed in this chapter: (a) reading raw data, (b) creating tables, (c) assigning level names to factors and (d) collapsing levels or variables for use in analysis.

For an illustration of these steps, we use the dataset `tv.dat`, supplied with the initial implementation of mosaic displays in R by Jay Emerson. In turn, they were derived from an early, compelling example of mosaic displays (Hartigan and Kleiner, 1984) that illustrated the method with data on a large sample of TV viewers whose behavior had been recorded for the Neilsen ratings. This data set contains sample television audience data from Neilsen Media Research for the week starting November 6, 1995.

The data file, `tv.dat`, is stored in frequency form as a file with 825 rows and 5 columns. There is no header line in the file, so when we use `read.table()` below, the variables will be named `V1` – `V5`. This data represents a 4-way table of size $5 \times 11 \times 5 \times 3 = 825$ where the table variables are `V1` – `V4`, and the cell frequency is read as `V5`.

The table variables are:
`V1`— values 1:5 correspond to the days Monday–Friday;
`V2`— values 1:11 correspond to the quarter-hour times 8:00 pm through 10:30 pm;
`V3`— values 1:5 correspond to ABC, CBS, NBC, Fox, and non-network choices;
`V4`— values 1:3 correspond to transition states: turn the television Off, Switch channels, or Persist in viewing the current channel.

2.9.1 Creating data frames and arrays

The file `tv.dat` is stored in the `doc/extdata` directory of vcdExtra; it can be read as follows:

```
> tv_data <- read.table(system.file("doc", "extdata", "tv.dat",
+                                   package = "vcdExtra"))
> str(tv_data)

'data.frame': 825 obs. of  5 variables:
 $ V1: int  1 2 3 4 5 1 2 3 4 5 ...
 $ V2: int  1 1 1 1 1 2 2 2 2 2 ...
 $ V3: int  1 1 1 1 1 1 1 1 1 1 ...
 $ V4: int  1 1 1 1 1 1 1 1 1 1 ...
 $ V5: int  6 18 6 2 11 6 29 25 17 29 ...
```

```
> head(tv_data, 5)

  V1 V2 V3 V4 V5
1  1  1  1  1  6
2  2  1  1  1 18
3  3  1  1  1  6
4  4  1  1  1  2
5  5  1  1  1 11
```

To read such data from a local file, just use `read.table()` in this form:

```
> tv_data <- read.table("C:/R/data/tv.dat")
```

or, to select the path using the file-chooser tool,

```
> 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]

```
> TV <- TV[,,1:3,]      # keep only ABC, CBS, NBC
> TV <- TV[,,,3]        # keep only Persist -- now a 3 way table
> structable(TV)

             Time 8:00 8:15 8:30 8:45 9:00 9:15 9:30 9:45 10:00 10:15 10:30
Day Network
Mon ABC            146  151  156   83  325  350  386  340   352   280   278
    CBS            337  293  304  233  311  251  241  164   252   265   272
    NBC            263  219  236  140  226  235  239  246   279   263   283
Tue ABC            244  181  231  205  385  283  345  192   329   351   364
    CBS            173  180  184  109  218  235  256  250   274   263   261
    NBC            315  254  280  241  370  214  195  111   188   190   210
Wed ABC            233  161  194  156  339  264  279  140   237   228   203
    CBS            158  126  207   59   98  103  122   86   109   105   110
    NBC            134  146  166   66  194  230  264  143   274   289   306
Thu ABC            174  183  197  181  187  198  211   86   110   122   117
    CBS            196  185  195  104  106  116  116   47   102    84    84
    NBC            515  463  472  477  590  473  446  349   649   705   747
Fri ABC            294  281  305  239  278  246  245  138   246   232   233
    CBS            130  144  154   81  129  153  136  126   138   136   152
    NBC            195  220  248  160  172  164  169   85   183   198   204
```

Finally, for some purposes, we might also want to collapse the 11 `Time`'s into a smaller number. Here, we use `collapse.table()` (see Section 2.7.2), which was designed for this purpose.

```
> TV2 <- collapse.table(TV,
+                       Time = c(rep("8:00-8:59", 4),
+                                rep("9:00-9:59", 4),
+                                rep("10:00-10:44", 3)))
> structable(Day ~ Time + Network, TV2)

                     Day  Mon  Tue  Wed  Thu  Fri
Time        Network
8:00-8:59   ABC           536  861  744  735 1119
            CBS          1167  646  550  680  509
            NBC           858 1090  512 1927  823
9:00-9:59   ABC          1401 1205 1022  682  907
            CBS           967  959  409  385  544
            NBC           946  890  831 1858  590
10:00-10:44 ABC           910 1044  668  349  711
            CBS           789  798  324  270  426
            NBC           825  588  869 2101  585
```

Congratulations! If you followed the operations described above, you are ready for the material described in the rest of the book. If not, try working through some of exercises below.

2.10 Lab exercises

Exercise 2.1 The packages vcd and vcdExtra contain many data sets with some examples of analysis and graphical display. The goal of this exercise is to familiarize yourself with these resources.

[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.

You can get a brief summary of these using the function `datasets()` from vcdExtra. Use the following to get a list of these with some characteristics and titles.

```
> ds <- datasets(package = c("vcd", "vcdExtra"))
> str(ds, vec.len = 2)

'data.frame': 74 obs. of  5 variables:
 $ Package: chr  "vcd" "vcd" ...
 $ Item   : chr  "Arthritis" "Baseball" ...
 $ class  : chr  "data.frame" "data.frame" ...
 $ dim    : chr  "84x5" "322x25" ...
 $ Title  : chr  "Arthritis Treatment Data" "Baseball Data" ...
```

(a) How many data sets are there altogether? How many are there in each package?
(b) Make a tabular display of the frequencies by `Package` and `class`.
(c) Choose one or two data sets from this list, and examine their help files (e.g., `help(Arthritis)` or `?Arthritis`). You can use, e.g., `example(Arthritis)` to run the R code for a given example.

Exercise 2.2 For each of the following data sets in the vcdExtra package, identify which are response variable(s) and which are explanatory. For factor variables, which are unordered (nominal) and which should be treated as ordered? Write a sentence or two describing substantitive questions of interest for analysis of the data. (*Hint*: use `data(foo, package="vcdExtra")` to load, and `str(foo)`, `help(foo)` to examine data set `foo`.)

(a) Abortion opinion data: *`Abortion`*
(b) Caesarian Births: *`Caesar`*
(c) Dayton Survey: *`DaytonSurvey`*
(d) Minnesota High School Graduates: *`Hoyt`*

Exercise 2.3 The data set *`UCBAdmissions`* is a 3-way table of frequencies classified by `Admit`, `Gender`, and `Dept`.

(a) Find the total number of cases contained in this table.
(b) For each department, find the total number of applicants.
(c) For each department, find the overall proportion of applicants who were admitted.
(d) Construct a tabular display of department (rows) and gender (columns), showing the proportion of applicants in each cell who were admitted relative to the total applicants in that cell.

Exercise 2.4 The data set *`DanishWelfare`* in vcd gives a 4-way, $3 \times 4 \times 3 \times 5$ table as a data frame in frequency form, containing the variable `Freq` and four factors, `Alcohol`, `Income`, `Status`, and `Urban`. The variable `Alcohol` can be considered as the response variable, and the others as possible predictors.

(a) Find the total number of cases represented in this table.
(b) In this form, the variables `Alcohol` and `Income` should arguably be considered *ordered* factors. Change them to make them ordered.
(c) Convert this data frame to table form, `DanishWelfare.tab`, a 4-way array containing the frequencies with appropriate variable names and level names.
(d) The variable `Urban` has 5 categories. Find the total frequencies in each of these. How would you collapse the table to have only two categories, `City`, `Non-city`?
(e) Use `structable()` or `ftable()` to produce a pleasing flattened display of the frequencies in the 4-way table. Choose the variables used as row and column variables to make it easier to compare levels of `Alcohol` across the other factors.

Exercise 2.5 The data set *UKSoccer* in vcd gives the distributions of number of goals scored by the 20 teams in the 1995/96 season of the Premier League of the UK Football Association.

```
> data("UKSoccer", package = "vcd")
> ftable(UKSoccer)

     Away  0  1  2  3  4
Home
0         27 29 10  8  2
1         59 53 14 12  4
2         28 32 14 12  4
3         19 14  7  4  1
4          7  8 10  2  0
```

This two-way table classifies all $20 \times 19 = 380$ games by the joint outcome (Home, Away), the number of goals scored by the `Home` and `Away` teams. The value `4` in this table actually represents 4 or more goals.

(a) Verify that the total number of games represented in this table is 380.
(b) Find the marginal total of the number of goals scored by each of the home and away teams.
(c) Express each of the marginal totals as proportions.
(d) Comment on the distribution of the numbers of home-team and away-team goals. Is there any evidence that home teams score more goals on average?

Exercise 2.6 The one-way frequency table *Saxony* in vcd records the frequencies of families with 0, 1, 2, ... 12 male children, among 6115 families with 12 children. This data set is used extensively in Chapter 3.

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

nMales
   0    1    2    3    4    5    6    7    8    9   10   11   12
   3   24  104  286  670 1033 1343 1112  829  478  181   45    7
```

Another data set, *Geissler*, in the vcdExtra package, gives the complete tabulation of all combinations of `boys` and `girls` in families with a given total number of children (`size`). The task here is to create an equivalent table, `Saxony12` from the *Geissler* data.

```
> data("Geissler", package = "vcdExtra")
> str(Geissler)

'data.frame':	90 obs. of  4 variables:
 $ boys : int  0 0 0 0 0 0 0 0 0 0 ...
 $ girls: num  1 2 3 4 5 6 7 8 9 10 ...
 $ size : num  1 2 3 4 5 6 7 8 9 10 ...
 $ Freq : int  108719 42860 17395 7004 2839 1096 436 161 66 30 ...
```

(a) Use `subset()` to create a data frame, `sax12` containing the *Geissler* observations in families with `size==12`.
(b) Select the columns for `boys` and `Freq`.
(c) Use `xtabs()` with a formula, `Freq ~ boys`, to create the one-way table.
(d) Do the same steps again to create a one-way table, `Saxony11`, containing similar frequencies for families of `size==11`.

Exercise 2.7 * *Interactive coding of table factors*: Some statistical and graphical methods for contingency tables are implemented only for two-way tables, but can be extended to 3+-way tables by recoding the factors to interactive combinations along the rows and/or columns, in a way similar to what `ftable()` and `structable()` do for printed displays.

For the *`UCBAdmissions`* data, produce a two-way table object, `UCB.tab2`, that has the combinations of `Admit` and `Gender` as the rows, and `Dept` as its columns, to look like the result below:

```
                 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.

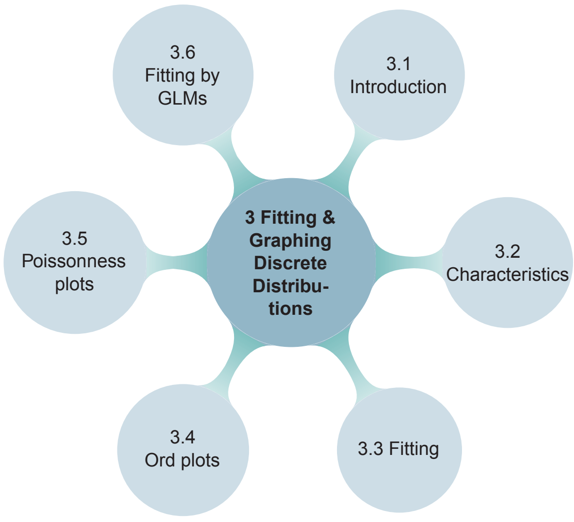
Exercise 2.8 The data set *`VisualAcuity`* in vcd gives a $4 \times 4 \times 2$ table as a frequency data frame.

```
> 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.

This page intentionally left blank



3 Fitting and Graphing Discrete Distributions

Discrete data often follow various theoretical probability models. Graphic displays are used to visualize goodness of fit, to diagnose an appropriate model, and determine the impact of individual observations on estimated parameters.

Not everything that counts can be counted, and not everything that can be counted counts.

Albert Einstein

Discrete frequency distributions often involve counts of occurrences of events, such as accident fatalities, incidents of terrorism or suicide, words in passages of text, or blood cells with some characteristic. Often interest is focused on how closely such data follow a particular probability distribution, such as the binomial, Poisson, or geometric distribution, which provide the basis for generating mechanisms that might give rise to the data. Understanding and visualizing such distributions in the simplest case of an unstructured sample provides a building block for generalized linear models (Chapter 11) where they serve as one component. They also provide the basis for a variety of recent extensions of regression models for count data (Chapter 11), some of which account for the excess counts of zeros (zero-inflated models) caused by left- or right-truncation often encountered in statistical practice.

This chapter describes the well-known discrete frequency distributions: the binomial, Poisson, negative binomial, geometric, and logarithmic series distributions in the simplest case of an unstructured sample. The chapter begins with simple graphical displays (line graphs and bar charts) to view the distributions of empirical data and theoretical frequencies from a specified discrete distribution.

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

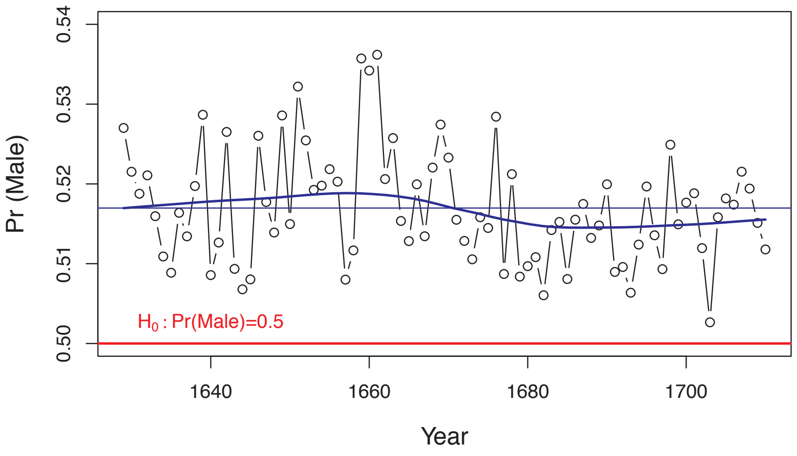


Figure 3.1: Arbuthnot's data on male/female sex ratios in London, 1629–1710, together with a (loess) smoothed curve (blue) over time and the mean Pr(Male).

HistData (Friendly, 2014a) package. For now, we simply display a plot of the probability of a male birth over time. The plot in Figure 3.1 shows the proportion of males over years, with horizontal lines at $\Pr(\text{Male}) = 0.5$ and the mean, $\Pr(\text{Male}) = 0.517$. Also shown is a (loess) smoothed curve, which suggests that any deviation from a constant sex ratio is relatively small, but also showed some systematic trend over time.

```
> data("Arbuthnot", package = "HistData")
> with(Arbuthnot, {
+   prob = Males / (Males + Females)
+   plot(x = Year, y = prob, type = "b",
+        ylim = c(0.5, 0.54), ylab = "Pr (Male)")
+   abline(h = 0.5, col = "red", lwd = 2)
+   abline(h = mean(prob), col = "blue")
+   lines(loess.smooth(Year, prob), col = "blue", lwd = 2)
+   text(x = 1640, y = 0.5, expression(H[0]: "Pr(Male)=0.5"),
+        pos = 3, col = "red")
+   })
```

Exercise 3.1 invites you to consider some other plots for this data. We return to this data in a later chapter where we ask whether the variation around the mean can be explained by any other considerations, or should just be considered random variation (see Exercise 7.1). △

Example 3.2: Families in Saxony

A related example of sex ratio data that ought to follow a binomial distribution comes from a classic study by A. Geissler (1889). Geissler listed the data on the distributions of boys and girls in families in Saxony for the period 1876–1885. In total, over four million births were recorded, and the sex distribution in the family was available because the parents had to state the sex of all their children on the birth certificate.

The complete data, classified by number of boys and number of girls (each 0–12) appear in Edwards (1958, Table 1).[1] Lindsey (1995, Table 6.2) selected only the 6115 families with 12 children, and listed the frequencies by number of males. The data are shown in table form in Table 3.1 in the

[1] Edwards (1958) notes that over these 10 years, many parents will have had several children, and their family composition is therefore recorded more than once. However, in families with a given number of children, each family can appear only once.

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
Families (n_k)	3	24	104	286	670	1,033	1,343	1,112	829	478	181	45	7

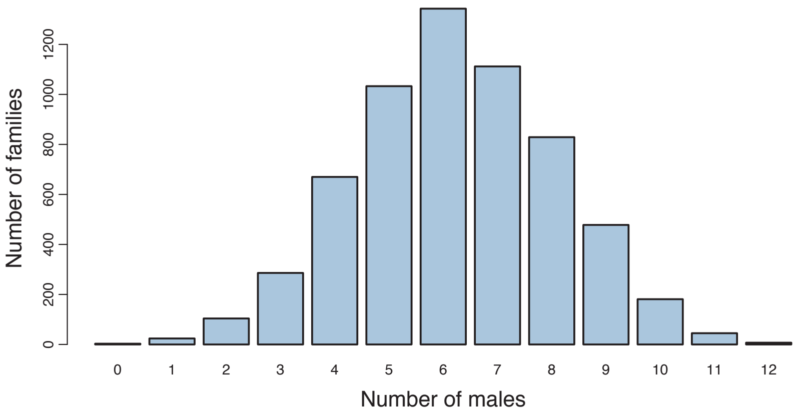


Figure 3.2: Number of males in Saxony families of size 12.

standard form of a complete discrete distribution. The basic outcome variable, $k = 0, 1, \dots, 12$, 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+
Frequency (n_k)	185	1,149	3,265	5,475	6,114	5,194	3,067	1,331	403	105	18

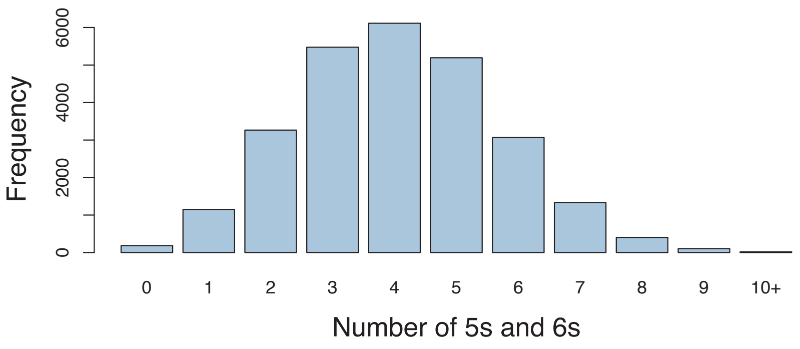


Figure 3.3: Weldon's dice data, frequency distribution of 5s and 6s in throws of 12 dice.

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 3.3: von Bortkiewicz's data on deaths by horse kicks

Number of deaths (k)	0	1	2	3	4	Sum
Frequency (n_k)	109	65	22	3	1	200

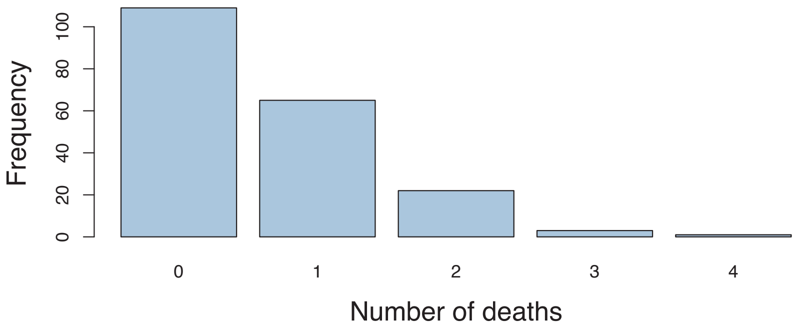


Figure 3.4: HorseKicks data, distribution of the number of deaths in 200 corps-years.

shown in Table 3.3. Ladislaus von Bortkiewicz, an economist and statistician, tabulated the number of soldiers in each of 14 army corps in the 20 years from 1875–1894 who died after being kicked by a horse (Andrews and Herzberg, 1985, p. 18). Table 3.3 shows the data used by Fisher (1925) for 10 of these army corps, summed over 20 years, giving 200 'corps-year' observations. In 109 corps-years, no deaths occurred; 65 corps-years had one death, etc.

The data set is available as *HorseKicks* in the vcd package. The distribution is plotted in Figure 3.4.

```
> data("HorseKicks", package = "vcd")
> barplot(HorseKicks, xlab = "Number of deaths", ylab = "Frequency",
+         col = "lightblue", cex.lab = 1.5)
```

△

EXAMPLE 3.5: Federalist Papers

In 1787–1788, Alexander Hamilton, John Jay, and James Madison wrote a series of newspaper essays to persuade the voters of New York State to ratify the U.S. Constitution. The essays were titled *The Federalist Papers* and all were signed with the pseudonym "Publius." Of the 77 papers published, the author(s) of 65 are known, but *both* Hamilton and Madison later claimed sole authorship of the remaining 12. Mosteller and Wallace (1963, 1984) investigated the use of statistical methods to identify authors of disputed works based on the frequency distributions of certain key function words, and concluded that Madison had indeed authored the 12 disputed papers.[3]

Table 3.4 shows the distribution of the occurrence of one of these "marker" words, the word *may* in 262 blocks of text (each about 200 words long) from issues of the *Federalist Papers* and other essays known to be written by James Madison. Read the table as follows: in 156 blocks, the word *may* did not occur; it occurred once in 63 blocks, etc. The distribution is plotted in Figure 3.5.

[3]It should be noted that this is a landmark work in the development of statistical methods and their application to the analysis of texts and cases of disputed authorship. In addition to *may*, they considered many such marker words, such as *any*, *by*, *from*, *upon*, and so forth. Among these, the word *upon* was the best discriminator between the works known by Hamilton (3 per 1000 words) and Madison (0.16 per 1000 words). In this work, they pioneered the use of Bayesian discriminant analysis, and the use of cross-validation to assess the stability of estimates and their conclusions.

Table 3.4: Number of occurrences of the word *may* in texts written by James Madison

Occurrences of *may* (k)	0	1	2	3	4	5	6	Sum
Blocks of text (n_k)	156	63	29	8	4	1	1	262

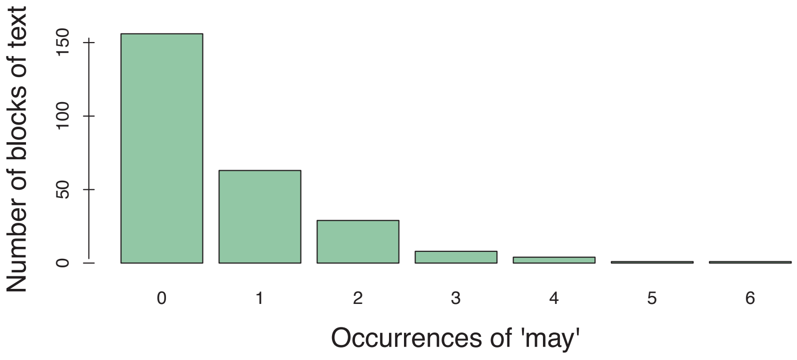


Figure 3.5: Federalist Papers data, distribution of the uses of the word *may*.

```
> data("Federalist", package = "vcd")
> barplot(Federalist,
+          xlab = "Occurrences of 'may'",
+          ylab = "Number of blocks of text",
+          col = "lightgreen", cex.lab = 1.5)
```

△

EXAMPLE 3.6: London cycling deaths

Aberdein and Spiegelhalter (2013) observed that from November 5–13, 2013, six people were killed while cycling in London. How unusual is this number of deaths in less than a two-week period? Was this a freak occurrence, or should Londoners petition for cycling lanes and greater road safety? To answer these questions, they obtained data from the UK Department of Transport *Road Safety Data* from 2005–2012 and selected all accident fatalities of cyclists within the city of London.

It seems reasonable to assume that, in any short period of time, deaths of people riding bicycles are independent events. If, in addition, the probability of such events is constant over this time span, the Poisson distribution should describe the distribution of $0, 1, 2, 3, \ldots$ deaths. Then, an answer to the main question can be given in terms of the probability of six (or more) deaths in a comparable period of time.

Their data, comprising 208 counts of deaths in the fortnightly periods from January 2005 to December 2012, are contained in the data set *CyclingDeaths* in vcdExtra. To work with the distribution, we first convert this to a one-way table.

```
> data("CyclingDeaths", package = "vcdExtra")
> CyclingDeaths.tab <- table(CyclingDeaths$deaths)
> CyclingDeaths.tab


  0   1   2   3
114  75  14   5
```

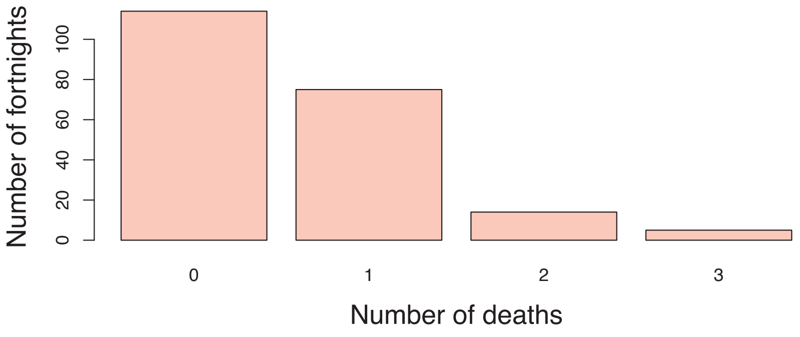


Figure 3.6: Frequencies of number of cyclist deaths in two-week periods in London, 2005–2012.

The maximum number of deaths was 3, which occurred in only 5 two-week periods. The distribution is plotted in Figure 3.6.

```
> barplot(CyclingDeaths.tab,
+         xlab = "Number of deaths", ylab = "Number of fortnights",
+         col = "pink", cex.lab = 1.5)
```

We return to this data in Example 3.10 and answer the question of how unusual six or more deaths would be in a Poisson distribution.

△

3.1.3 Type-token distributions

There are a variety of other types of discrete data distributions. One important class is ***type-token*** distributions, where the basic count k is the number of distinct types of some observed event, $k = 1, 2, \ldots$ and the frequency, n_k, is the number of different instances observed. For example, distinct words in a book, words that subjects list as members of the semantic category "fruit," musical notes that appear in a score, and species of animals caught in traps can be considered as types, and the occurrences of of those type comprise tokens.

This class differs from the Poisson type considered above in that the frequency for value $k = 0$ is *unobserved*. Thus, questions like (a) How many words did Shakespeare know? (b) How many words in the English language are members of the "fruit" category? (c) How many wolves remain in Canada's Northwest territories? depend on the unobserved count for $k = 0$. They cannot easily be answered without appeal to additional information or statistical theory.

Example 3.7: Butterfly species in Malaya

In studies of the diversity of animal species, individuals are collected and classified by species. The distribution of the number of species (types) where $k = 1, 2, \ldots$ individuals (tokens) were collected forms a kind of type-token distribution. An early example of this kind of distribution was presented by Fisher et al. (1943). Table 3.5 lists the number of individuals of each of 501 species of butterfly collected in Malaya. There were thus 118 species for which just a single instance was found, 74 species for which two individuals were found, down to 3 species for which 24 individuals were collected. Fisher et al. note, however, that the distribution was truncated at $k = 24$. Type-token distributions are often J-shaped, with a long upper tail, as we see in Figure 3.7.

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

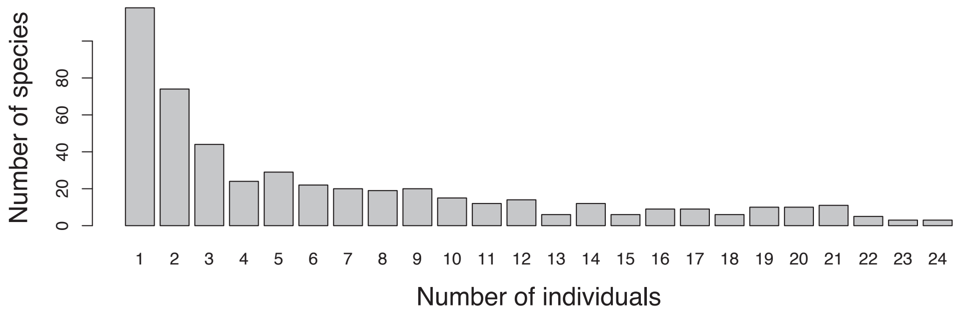


Figure 3.7: Butterfly species in Malaya.

```
> 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

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

Table 3.7: R functions for discrete probability distributions

Discrete distribution	Density (pmf) function	Cumulative (CDF)	Quantile CDF^{-1}	Random # generator
Binomial	`dbinom()`	`pbinom()`	`qbinom()`	`rbinom()`
Poisson	`dpois()`	`ppois()`	`qpois()`	`rpois()`
Negative binomial	`dnbinom()`	`pnbinom()`	`qnbinom()`	`rnbinom()`
Geometric	`dgeom()`	`pgeom()`	`qgeom()`	`rgeom()`
Logarithmic series	`dlogseries()`	`plogseries()`	`qlogseries()`	`rlogseries()`

parameters can be estimated and how to plot the frequency distribution. R has a wealth of functions for a wide variety of distributions. For ease of reference, their names and types for the distributions covered here are shown in Table 3.7. The naming scheme is simple and easy to remember: for each distribution, there are functions, with a prefix letter, `d`, `p`, `q`, `r`, followed by the name for that class of distribution:[4]

d a density function,[5] $\Pr\{X = k\} \equiv p(k)$ for the probability that the variable X takes the value k.
p a cumulative probability/density function, or CDF, $F(k) = \sum_{X \leq k} p(k)$.
q a quantile function, the inverse of the CDF, $k = F^{-1}(p)$. The quantile is defined as the smallest value x such that $F(k) \geq p$.
r a random number generating function for that distribution.

In the R console, `help(Distributions)` gives an overview listing of the distribution functions available in the stats package.

3.2.1 The binomial distribution

The binomial distribution, $\text{Bin}(n, p)$, arises as the distribution of the number k of events of interest that occur in n independent trials when the probability of the event on any one trial is the constant value $p = \Pr(\text{event})$. For example, if 15% of the population has red hair, the number of redheads in randomly sampled groups of $n = 10$ might follow a binomial distribution, $\text{Bin}(10, 0.15)$; in Weldon's dice data (Example 3.3), the probability of a 5 or 6 should be $\frac{1}{3}$ on any one trial, and the number of 5s or 6s in tosses of 12 dice would follow $\text{Bin}(12, \frac{1}{3})$.

Over n independent trials, the number of events k may range from 0 to n; if X is a random variable with a binomial distribution, the probability that $X = k$ is given by

$$\text{Bin}(n, p) : \Pr\{X = k\} \equiv p(k) = \binom{n}{k} p^k (1-p)^{n-k} \qquad k = 0, 1, \ldots, n\,, \tag{3.1}$$

where $\binom{n}{k} = n!/k!(n-k)!$ is the number of ways of choosing k out of n.

The first three (central) moments of the binomial distribution are as follows (letting $q = 1 - p$),

$$\begin{aligned} \text{Mean}(X) &= np \\ \text{Var}(X) &= npq \\ \text{Skew}(X) &= npq(q-p)\,. \end{aligned}$$

[4]The CRAN Task View on Probability Distributions, `http://cran.r-project.org/web/views/Distributions.html`, provides a general overview and lists a wide variety of contributed packages for specialized distributions, discrete and continuous.

[5]For discrete random variables this is usually called the probability mass function (pmf).

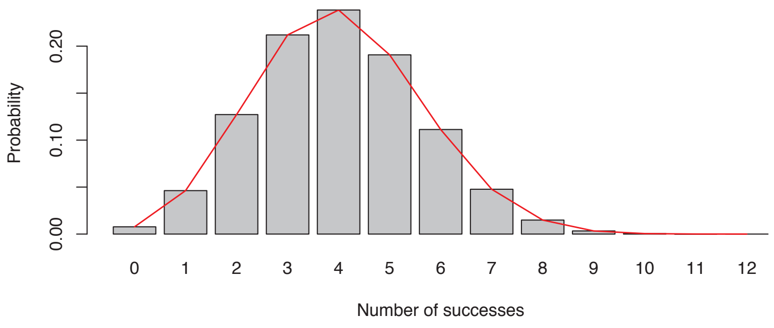


Figure 3.8: Binomial distribution for $k = 0, \dots, 12$ successes in 12 trials and p=1/3.

It is easy to verify that the binomial distribution has its maximum variance when $p = \frac{1}{2}$. It is symmetric (Skew(x)=0) when $p = \frac{1}{2}$, and negatively (positively) skewed when $p < \frac{1}{2}$ ($p > \frac{1}{2}$).

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 $k = 0, \dots, 12$ 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.

[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.

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.

```
> Weldon_df <- as.data.frame(WeldonDice) # convert to data frame
>
> k <- 0 : 12                          # same as seq(0, 12)
> Pk <- dbinom(k, 12, 1/3)            # binomial probabilities
> Pk <- c(Pk[1:10], sum(Pk[11:13])) # sum values for 10+
> Exp <- round(26306 * Pk, 5)         # expected frequencies
> Diff <- Weldon_df$Freq - Exp         # raw residuals
> Chisq <- Diff^2 / Exp
> data.frame(Weldon_df, Prob = round(Pk, 5), Exp, Diff, Chisq)

   n56 Freq    Prob      Exp      Diff   Chisq
1    0  185 0.00771  202.749  -17.7495 1.55386
2    1 1149 0.04624 1216.497  -67.4968 3.74503
3    2 3265 0.12717 3345.366  -80.3661 1.93064
4    3 5475 0.21195 5575.610 -100.6102 1.81548
5    4 6114 0.23845 6272.561 -158.5614 4.00821
6    5 5194 0.19076 5018.049  175.9509 6.16947
7    6 3067 0.11127 2927.195  139.8047 6.67716
8    7 1331 0.04769 1254.512   76.4877 4.66346
9    8  403 0.01490  392.035   10.9649 0.30668
10   9  105 0.00331   87.119   17.8811 3.67008
11 10+   18 0.00054   14.305    3.6947 0.95424
```

△

Finally, we can use programming features in R to calculate and plot probabilities for binomial distributions over a range of both k and p as follows, for the purposes of graphing the distributions as one or both varies. The following code uses outer() to create a 13×4 matrix Prob containing the result of dbinom() for all combinations of k = 0:12 and p = c(1/6, 1/3, 1/2, 2/3). These values are then supplied as arguments to dbinom().

```
> p <- c(1/6, 1/3, 1/2, 2/3)
> k <- 0 : 12
> Prob <- outer(k, p, function(k, p) dbinom(k, 12, p))
> str(Prob)

 num [1:13, 1:4] 0.1122 0.2692 0.2961 0.1974 0.0888 ...
```

In this form, each column of Prob can be most easily plotted against k using matplot(). The following code generates Figure 3.9.

```
> col <- palette()[2:5]
> matplot(k, Prob,
+         type = "o", pch = 15 : 17, col = col, lty = 1,
+         xlab = "Number of Successes", ylab = "Probability")
> legend("topright", legend = c("1/6","1/3","1/2","2/3"),
+        pch = 15 : 17, lty = 1, col = col, title = "Pr(Success)")
```

3.2.2 The Poisson distribution

The Poisson distribution gives the probability of an event occurring $k = 0, 1, 2, \ldots$ times over a large number of independent "trials," when the probability, p, that the event occurs on any one trial (in time or space) is small and constant. Hence, the Poisson distribution is usually applied to the study of rare events such as highway accidents at a particular location, deaths from horse kicks,

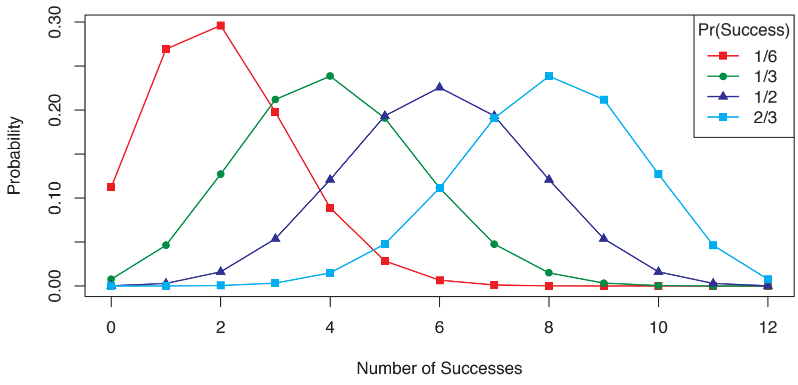


Figure 3.9: Binomial distributions for $k = 0, \dots, 12$ successes in $n = 12$ trials, and four values of p.

or defects in a well-controlled manufacturing process. Other applications include: the number of customers contacting a call center per unit time; the number of insurance claims per unit region or unit time; number of particles emitted from a small radioactive sample.

For the Poisson distribution, the probability function is

$$\text{Pois}(\lambda) : \Pr\{X = k\} \equiv p(k) = \frac{e^{-\lambda}\,\lambda^k}{k!} \qquad k = 0, 1, \dots \tag{3.2}$$

where the rate parameter, λ (> 0), turns out to be the mean of the distribution. The first three (central) moments of the Poisson distribution are:

$$\begin{aligned} \text{Mean}(X) &= \lambda \\ \text{Var}(X) &= \lambda \\ \text{Skew}(X) &= \lambda^{-1/2} \end{aligned}$$

So, the mean and variance of the Poisson distribution are always the same, which is sometimes used to identify a distribution as Poisson. For the binomial distribution, the mean (Np) is always greater than the variance (Npq); for other distributions (negative binomial and geometric) the mean is less than the variance. The Poisson distribution is always positively skewed, but skewness decreases as λ increases.

The maximum likelihood estimator of the parameter λ in Eqn. (3.2) is just the mean of the distribution,

$$\hat{\lambda} = \bar{x} = \frac{\sum_k k\, n_k}{\sum_k n_k} \,. \tag{3.3}$$

Hence, the expected frequencies can be estimated by substituting the sample mean into Eqn. (3.2) and multiplying by the total sample size N.

There are many useful properties of the Poisson distribution.[8] Among these are:

- Poisson variables have a nice reproductive property: if $X_1, X_2, \dots X_m$ are independent Poisson variables with the same parameter λ, then their sum, $\sum X_i$ is a Poisson variate with parameter $m\lambda$; if the Poisson parameters differ, the sum is still Poisson with parameter $\sum \lambda_i$.
- For two or more independent Poisson variables, $X_1 \sim \text{Pois}(\lambda_1), X_2 \sim \text{Pois}(\lambda_2), \dots$, with rate parameters $\lambda_1, \lambda_2 \dots$, the distribution of any X_i, *conditional on their sum*, $\sum_j X_j = n$, is binomial, $\text{Bin}(n, p)$, where $p = \lambda_i / \sum_j \lambda_j$.

[8]See: `http://en.wikipedia.org/wiki/Poisson_distribution`.

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

- As λ increases, the Poisson distribution becomes increasingly symmetric, and approaches the normal distribution $N(\lambda, \lambda)$ with mean and variance λ as $\lambda \to \infty$. The approximation is quite good with $\lambda > 20$.
- 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.

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

As a preliminary check of the distributions for the home and away goals, we can determine if the

[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."

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 frequency form. We use within() to convert Home and Away to numeric variables, and calculate Total as their sum.

```
> data("UKSoccer", package = "vcd")
>
> soccer.df <- as.data.frame(UKSoccer, stringsAsFactors = FALSE)
> soccer.df <- within(soccer.df, {
+   Home <- as.numeric(Home)        # make numeric
+   Away <- as.numeric(Away)        # make numeric
+   Total <- Home + Away             # total goals
+ })
> str(soccer.df)

'data.frame': 25 obs. of  4 variables:
 $ Home : num  0 1 2 3 4 0 1 2 3 4 ...
 $ Away : num  0 0 0 0 0 1 1 1 1 1 ...
 $ Freq : num  27 59 28 19 7 29 53 32 14 8 ...
 $ Total: num  0 1 2 3 4 1 2 3 4 5 ...
```

To calculate the mean and variance of these variables, first expand the data frame to 380 individual observations using expand.dft(). Then use apply() over the rows to calculate the mean and variance in each column.

```
> soccer.df <- expand.dft(soccer.df)    # expand to ungrouped form
> apply(soccer.df, 2, FUN = function(x) c(mean = mean(x), var = var(x)))

       Home   Away  Total
mean 1.4868 1.0632 2.5500
var  1.3164 1.1728 2.6175
```

The means are all approximately equal to the corresponding variances. More to the point, the variance of the Total score is approximately equal to the sum of the individual variances. Note also there does appear to be an advantage for the home team, of nearly half a goal.

△

EXAMPLE 3.10: London cycling deaths

A quick check of whether the number of deaths among London cyclists follows the Poisson distribution can be carried out by calculating the mean and variance. The ***index of dispersion***, the ratio of the variance to the mean, is commonly used to quantify whether a set of observed frequencies is more or less dispersed than a reference (Poisson) distribution.

```
> with(CyclingDeaths, c(mean = mean(deaths),
+                        var = var(deaths),
+                        ratio = mean(deaths) / var(deaths)))

   mean     var   ratio
0.56731 0.52685 1.07679
```

Thus, there was an average of about 0.57 deaths per fortnight, or a bit more than 1 per month, and no evidence for over- or underdispersion.

We can now answer the question of whether it was an extraordinary event to observe six deaths in a two-week period, by calculating the probability of more than 5 deaths using ppois().

```
> mean.deaths <- mean(CyclingDeaths$deaths)
> ppois(5, mean.deaths, lower.tail = FALSE)

[1] 2.8543e-05
```

This probability is extremely small, so we conclude that the occurrence of six deaths was a singular event. The interpretation of this result might indicate an increased risk to cycling in London, and might prompt further study of road safety. △

3.2.2.1 Calculation and visualization

For the Poisson distribution, you can generate probabilities using `dpois(x, lambda)` for the numbers of events in `x` with rate parameter `lambda`. As we did earlier for the binomial distribution, we can calculate these for a collection of values of `lambda` by using `expand.grid()` to create all combinations of the values of `x` we wish to plot.

EXAMPLE 3.11: Plotting styles for discrete distributions

In this example, we illustrate some additional styles for plotting discrete distributions, using both lattice `xyplot()` and the ggplot2 package. The goal here is to visualize a collection of Poisson distributions for varying values of λ.

We first create the 63 combinations of `x = 0:20` for three values of λ, `lambda = c(1, 4, 10)`, and use these columns as arguments to `dpois()`. Again, `lambda` is a numeric variable, but the plotting methods are easier if this variable is converted to a factor.

```
> KL <- expand.grid(k = 0 : 20, lambda = c(1, 4, 10))
> pois_df <- data.frame(KL, prob = dpois(KL$k, KL$lambda))
> pois_df$lambda = factor(pois_df$lambda)
> str(pois_df)

'data.frame': 63 obs. of  3 variables:
 $ k     : int  0 1 2 3 4 5 6 7 8 9 ...
 $ lambda: Factor w/ 3 levels "1","4","10": 1 1 1 1 1 1 1 1 1 1 ...
 $ prob  : num  0.3679 0.3679 0.1839 0.0613 0.0153 ...
```

Discrete distributions are often plotted as bar charts or in histogram-like form, as we did for the examples in Section 3.1, rather than the line-graph form used for the binomial distribution in Figure 3.9. With `xyplot()`, the plot style is controlled by the `type` argument, and the code below uses `type = c("h", "p")` to get *both* histogram-like lines to the origin and points. As well, the plot formula, `prob ~ k | lambda` instructs `xyplot()` to produce a multi-panel plot, conditioned on values of `lambda`. These lines produce Figure 3.10.

```
> library(lattice)
> xyplot(prob ~ k | lambda, data = pois_df,
+   type = c("h", "p"), pch = 16, lwd = 4, cex = 1.25, layout = c(3, 1),
+   xlab = list("Number of events (k)", cex = 1.25),
+   ylab = list("Probability", cex = 1.25))
```

The line-graph plot style of Figure 3.9 has the advantage that it is easier to compare the separate distributions in a single plot (using the `groups` argument) than across multiple panels (using a conditioning formula). It has the disadvantages that (a) a proper legend is difficult to construct with lattice, and (b) is difficult to read, because you have to visually coordinate the curves in the plot with the values shown in the legend. Figure 3.11 solves both problems using the directlabels package.

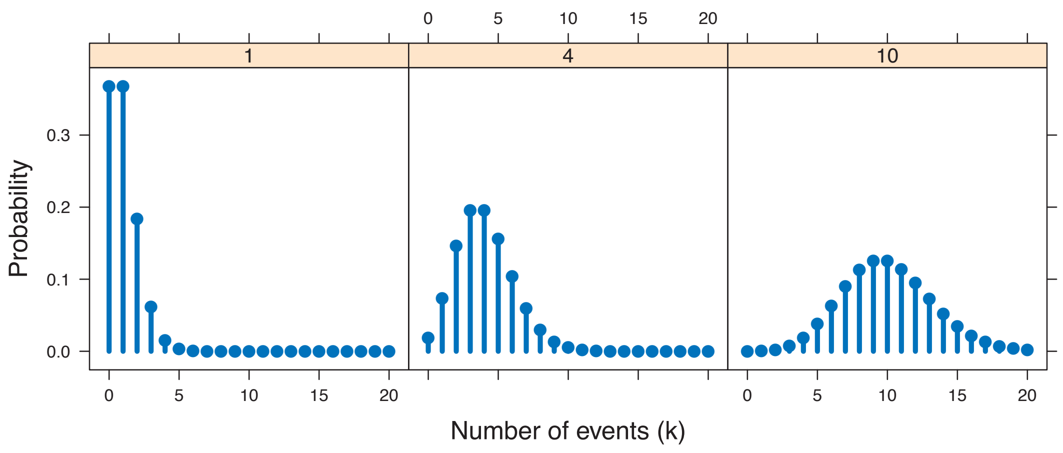


Figure 3.10: Poisson distributions for $\lambda = 1, 4, 10$, in a multi-panel display.

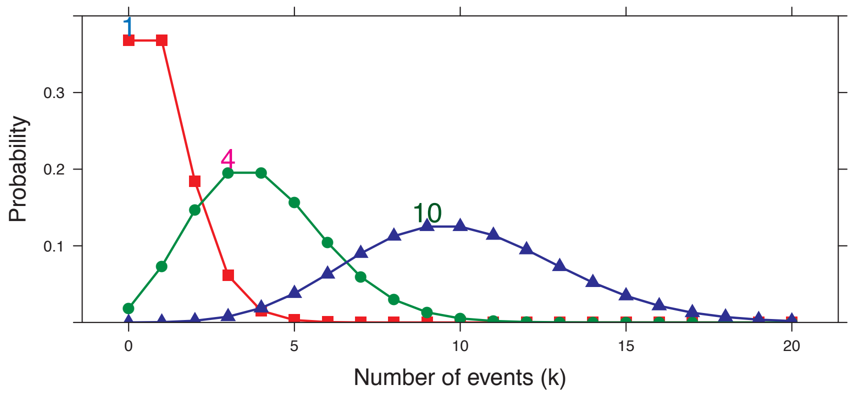


Figure 3.11: Poisson distributions for $\lambda = 1, 4, 10$, using direct labels.

```
> mycol <- palette()[2:4]
> plt <- xyplot(prob ~ k, data = pois_df, groups = lambda,
+   type = "b", pch = 15 : 17, lwd = 2, cex = 1.25, col = mycol,
+   xlab = list("Number of events (k)", cex = 1.25),
+   ylab = list("Probability",  cex = 1.25),
+   ylim = c(0, 0.4))
>
> library(directlabels)
> direct.label(plt, list("top.points", cex = 1.5, dl.trans(y = y + 0.1)))
```

Note that the plot constructed by `xyplot()` is saved as a ("trellis") object, `plt`. The function `direct.label()` massages this to add the labels directly to each curve. In the second argument above, `"top.points"` says to locate these at the maximum value on each curve.

Finally, we illustrate the use of ggplot2 to produce a single-panel, multi-line plot of these distributions. The basic plot uses `aes(x = k, y = prob, ...)` to produce a plot of `prob` vs. `k`, assigning color and shape attributes to the values of `lambda`.

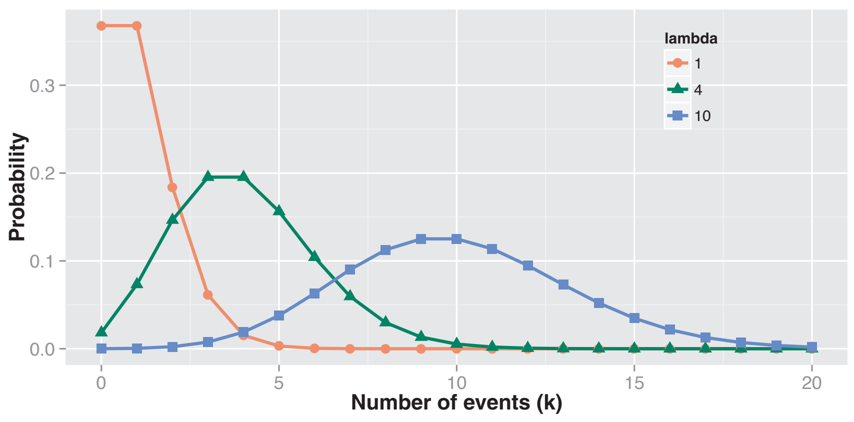


Figure 3.12: Poisson distributions for $\lambda = 1, 4, 10$, using ggplot2.

```
> library(ggplot2)
> gplt <- ggplot(pois_df,
+              aes(x = k, y = prob, colour = lambda, shape = lambda)) +
+   geom_line(size = 1) + geom_point(size = 3) +
+   xlab("Number of events (k)") +
+   ylab("Probability")
```

ggplot2 allows most details of the plot to be modified using `theme()`. Here we use this to move the legend inside the plot, and enlarge the axis labels and titles (producing Figure 3.12).

```
> gplt + theme(legend.position = c(0.8, 0.8)) +  # manually move legend
+        theme(axis.text = element_text(size = 12),
+              axis.title = element_text(size = 14, face = "bold"))
```

△

3.2.3 The negative binomial distribution

The negative binomial distribution is a type of waiting-time distribution, but also arises in statistical applications as a generalization of the Poisson distribution, allowing for ***overdispersion*** (variance > mean). See Hilbe (2011) for a comprehensive treatment of negative binomial statistical models with many applications in R.

One form of the negative binomial distribution (also called the ***Pascal distribution***) arises when a series of independent Bernoulli trials is observed with constant probability p of some event, and we ask how many non-events (failures), k, it takes to observe n successful events. For example, in tossing one die repeatedly, we may consider the outcome "1" as a "success" (with $p = \frac{1}{6}$) and ask about the probability of observing $k = 0, 1, 2, \ldots$ failures before getting $n = 3$ 1s.

The probability function with parameters n (a positive integer, $0 < n < \infty$) and p $(0 < p < 1)$ gives the probability that k non-events (failures) are observed before the n-th event (success), and can be written[10]

$$\text{NBin}(n,p) : \Pr\{X = k\} \equiv p(k) = \binom{n+k-1}{k} p^n (1-p)^k \qquad k = 0, 1, \ldots, \infty \tag{3.4}$$

This formulation makes clear that a given sequence of events involves a total of $n + k$ trials, of which there are n successes, with probability p^n, and k are failures, with probability $(1 - p)^k$. The binomial coefficient $\binom{n+k-1}{k}$ gives the number of ways to choose the k successes from the remaining $n + k - 1$ trials preceding the last success.

The first three central moments of the negative binomial distribution are:

$$\begin{aligned} \text{Mean}(X) &= nq/p = \mu \\ \text{Var}(X) &= nq/p^2 \\ \text{Skew}(X) &= \frac{2-p}{\sqrt{nq}}\,, \end{aligned}$$

where $q = 1 - p$. The variance of X is therefore greater than the mean, and the distribution is always positively skewed.

A more general form of the negative binomial distribution (the ***Polya distribution***) allows n to take non-integer values and to be an unknown parameter. In this case, the combinatorial coefficient, $\binom{n+k-1}{k}$ in Eqn. (3.4), is calculated using the gamma function, $\Gamma(\bullet)$, a generalization of the factorial for non-integer values, defined so that $\Gamma(x + 1) = x!$ when x is an integer.

Then the probability function Eqn. (3.4) becomes

$$\Pr\{X = k\} \equiv p(k) = \frac{\Gamma(n+k)}{\Gamma(n)\Gamma(k+1)} p^n (1-p)^k \qquad k = 0, 1, \ldots, \infty\,. \tag{3.5}$$

Greenwood and Yule (1920) developed the negative binomial distribution as a model for accident proneness or susceptibility of individuals to repeated attacks of disease. They assumed that for any individual, i, the number of accidents or disease occurrences has a Poisson distribution with parameter λ_i. If individuals vary in proneness, so that the λ_i have a gamma distribution, the resulting distribution is the negative binomial.

In this form, the negative binomial distribution is frequently used as an alternative to the Poisson distribution when the assumptions of the Poisson (constant probability and independence) are not satisfied, or when the variance of the distribution is greater than the mean (overdispersion). This gives rise to an alternative parameterization in terms of the mean (μ) of the distribution and its relation to the variance. From the relation of the mean and variance to the parameters n, p given above,

$$\text{Mean}(X) = \mu = \frac{n(1-p)}{p} \quad \Longrightarrow \quad p = \frac{n}{n+\mu}\,, \tag{3.6}$$

$$\text{Var}(X) = \frac{n(1-p)}{p^2} \quad \Longrightarrow \quad \text{Var}(X) = \mu + \frac{\mu^2}{n}\,. \tag{3.7}$$

This formulation allows the variance of the distribution to exceed the mean, and in these terms, the "size" parameter n is called the ***dispersion parameter***.[11] Increasing this parameter corresponds to less heterogeneity, variance closer to the mean, and therefore greater applicability of the Poisson distribution.

[10]There are a variety of other parameterizations of the negative binomial distribution, but all of these can be converted to the form shown here, which is relatively standard, and consistent with R. They differ in whether the parameter n relates to the number of successes or the total number of trials, and whether the stopping criterion is defined in terms of failures or successes. See: `http://en.wikipedia.org/wiki/Negative_binomial_distribution` for details on these variations.

[11]Other terms are "shape parameter," with reference to the mixing distribution of Poissons with varying λ, "heterogeneity parameter," or "aggregation parameter."

3.2.3.1 Calculation and visualization

In R, the density (pmf), distribution (CDF), quantile, and random number functions for the negative binomial distribution are a bit special, in that the parameterization can be specified using either (n, p) or (n, μ) forms, where $\mu = n(1-p)/p$. In our notation, probabilities can be calculated using `dnbinom()` using the call `dbinom(k, n, p)` or the call `dbinom(k, n, mu=)`, as illustrated below:

```
> k <- 2
> n <- 2 : 4
> p <- 0.2
> dnbinom(k, n, p)

[1] 0.07680 0.03072 0.01024

> (mu <- n * (1 - p) / p)

[1]  8 12 16

> dnbinom(k, n, mu = mu)

[1] 0.07680 0.03072 0.01024
```

Thus, for the distribution with `k = 2` failures and `n = 2:4` successes with probability `p = 0.2`, the values `n = 2:4` correspond to means $\mu = 8, 12, 16$ as shown above.

As before, we can calculate these probabilities for a range of the combinations of arguments using `expand.grid()`. In the example below, we allow three values for each of `n` and `p` and calculate all probabilities for all values of `k` from 0 to 20. The result, `nbin_df`, is like a 3-way, $21 \times 3 \times 3$ array of `prob` values, but in data frame format.

```
> XN <- expand.grid(k = 0 : 20, n = c(2, 4, 6), p = c(0.2, 0.3, 0.4))
> nbin_df <- data.frame(XN, prob = dnbinom(XN$k, XN$n, XN$p))
> nbin_df$n <- factor(nbin_df$n)
> nbin_df$p <- factor(nbin_df$p)
> str(nbin_df)

'data.frame': 189 obs. of  4 variables:
 $ k   : int  0 1 2 3 4 5 6 7 8 9 ...
 $ n   : Factor w/ 3 levels "2","4","6": 1 1 1 1 1 1 1 1 1 1 ...
 $ p   : Factor w/ 3 levels "0.2","0.3","0.4": 1 1 1 1 1 1 1 1 1 1 ...
 $ prob: num  0.04 0.064 0.0768 0.0819 0.0819 ...
```

With 9 combinations of the parameters, it is most convenient to plot these in separate panels, in a 3×3 display, as in Figure 3.13. The formula `prob ~ k | n + p` in the call to `xyplot()` constructs plots of `prob` vs. `k` conditioned on the combinations of `n` and `p`.

```
> xyplot(prob ~ k | n + p, data = nbin_df,
+   xlab = list("Number of failures (k)", cex = 1.25),
+   ylab = list("Probability",  cex = 1.25),
+   type = c("h", "p"), pch = 16, lwd = 2,
+   strip = strip.custom(strip.names = TRUE)
+   )
```

It can be readily seen that the mean increases from left to right with `n`, and increases from top to bottom with decreasing `p`. For these distributions, we can also calculate the theory-implied means, μ, across the entire distributions, $k = 0, 1, \ldots \infty$, as shown below.

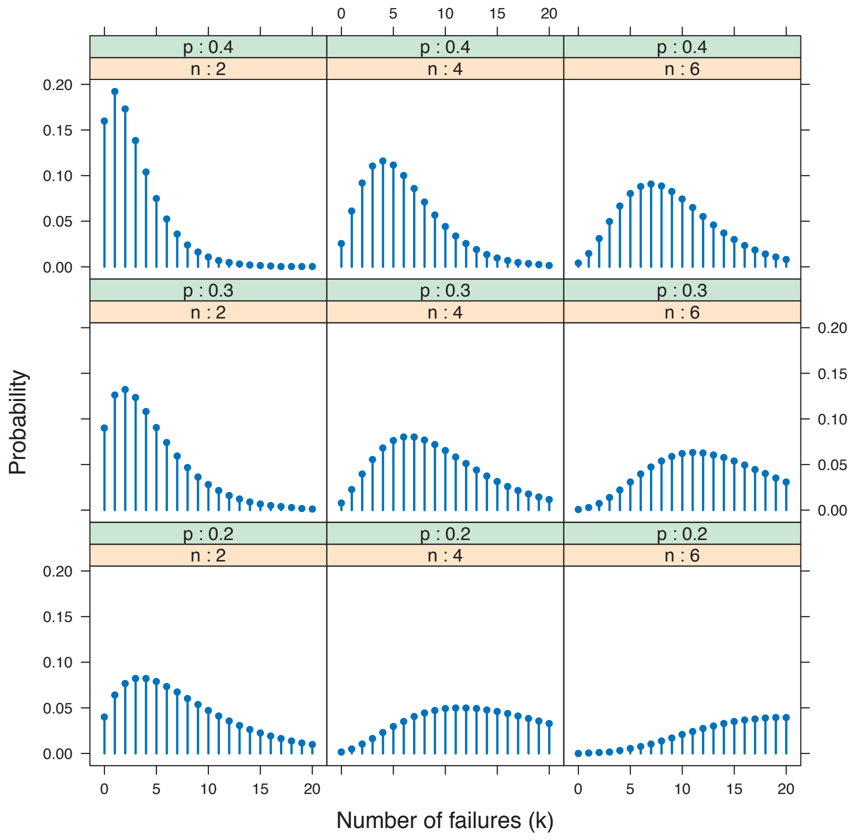


Figure 3.13: Negative binomial distributions for $n = 2, 4, 6$ and $p = 0.2, 0.3, 0.4$, using `xyplot()`.

```
> n <- c(2, 4, 6)
> p <- c(0.2, 0.3, 0.4)
> NP <- outer(n, p, function(n, p) n * (1 - p) / p)
> dimnames(NP) <- list(n = n, p = p)
> NP

   p
n   0.2     0.3 0.4
  2   8  4.6667   3
  4  16  9.3333   6
  6  24 14.0000   9
```

3.2.4 The geometric distribution

The special case of the negative binomial distribution when $n = 1$ is a geometric distribution. We observe a series of independent trials and count the number of non-events (failures) preceding the first successful event. The probability that there will be k failures before the first success is given by

$$\text{Geom}(p) : \Pr\{X = k\} \equiv p(k) = p(1 - p)^k \qquad k = 0, 1, \dots . \tag{3.8}$$

For this distribution the central moments are:

$$\text{Mean}(X) \;=\; 1/p$$

$$\begin{aligned}\text{Var}(X) &= (1-p)/p^2 \\ \text{Skew}(X) &= (2-p)/\sqrt{1-p}\end{aligned}$$

Note that estimation of the parameter p for the geometric distribution can be handled as the special case of the negative binomial by fixing $n = 1$, so no special software is needed. Going the other way, if $X_1, X_2, \ldots X_n$ are independent geometrically distributed as $\text{Geom}(p)$, then their sum, $Y = \sum_j^n X_j$ is distributed as $\text{NBin}(p, n)$.

In R, the standard set of functions for the geometric distribution are available as `dgeom(x, prob)`, `pgeom(q, prob)`, `qgeom(p, prob)`, and `rgeom(n, prob)`, where `prob` represents p here. Visualization of the geometric distribution follows the pattern used earlier for other discrete distributions.

3.2.5 The logarithmic series distribution

The logarithmic series distribution is a long-tailed distribution introduced by Fisher et al. (1943) in connection with data on the abundance of individuals classified by species of the type shown for the distribution of butterfly species in Table 3.5.

The probability distribution function with parameter p is given by

$$\text{LogSer}(p) : \Pr\{X = k\} \equiv p(k) = \frac{p^k}{-(k \log(1-p))} = \alpha\, p^k/k \qquad k = 1, 2, \ldots, \infty\,, \tag{3.9}$$

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

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	$\theta^k/[-k\log(1-\theta)]$	$\theta = \theta$	$-\log(1-\theta)$	$1/k$

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).

3.3 Fitting discrete distributions

In applications to discrete data such as the examples in Section 3.1, interest is often focused on how closely such data follow a particular distribution, such as the Poisson, binomial, or geometric distribution. A close fit provides for interpretation in terms of the underlying mechanism for the distribution; conversely, a bad fit can suggest the possibility for improvement by relaxing one or more of the assumptions. We examine more detailed and nuanced methods for diagnosing and testing discrete distributions in Section 3.4 and Section 3.5 below.

Fitting a discrete distribution involves three basic steps:

1. Estimating the parameter(s) of the distribution from the data; for example, p for the binomial, λ for the Poisson, n and p for the negative binomial. Typically, this is carried out by maximum likelihood methods, or a simpler method of moments, which equates sample moments (mean, variance, skewness) to those of the theoretical distribution, and solves for the parameter estimates. These methods are illustrated in Section 3.3.1.
2. From this, we can calculate the fitted probabilities, $\hat{p}_k$ that apply for the given distribution, or equivalently, the model expected frequencies, $N\hat{p}_k$, where N is the total sample size.
3. Finally, we can calculate goodness-of-fit tests measuring the departure between the observed and fitted frequencies.

Often goodness-of-fit is examined with a classical (Pearson) ***goodness-of-fit*** (GOF) chi-squared test,

$$X^2 = \sum_{k=1}^{K} \frac{(n_k - N\hat{p}_k)^2}{N\hat{p}_k} \sim \chi^2_{(K-s-1)} \, , \tag{3.10}$$

where there are K frequency classes, s parameters have been estimated from the data, and $\hat{p}_k$ is the estimated probability of each basic count, under the null hypothesis that the data follows the chosen distribution.

An alternative test statistic is the likelihood-ratio G^2 statistic,

$$G^2 = \sum_{k=1}^{K} n_k \log(n_k / N\hat{p}_k) \ , \tag{3.11}$$

when the $\hat{p}_k$ are estimated by maximum likelihood, which also has an asymptotic $\chi^2_{(K-s-1)}$ distribution. “Asymptotic” means that these are *large sample tests*, meaning that the test statistic follows the χ^2 distribution increasingly well as $N \rightarrow \infty$. A common rule of thumb is that all *expected* frequencies should exceed one and that fewer than 20% should be less than 5.

EXAMPLE 3.12: Death by horse kick

We illustrate the basic ideas of goodness-of fit tests with the *HorseKicks* data, where we expect a Poisson distribution with parameter λ = mean number of deaths. As shown in Eqn. (3.3), this is calculated as the frequency (n_k) weighted mean of the k values, here, number of deaths.

In R, such one-way frequency distributions should be converted to data frames with numeric variables. The calculation below uses `weighted.mean()` with the frequencies as weights, and finds $\lambda = 0.61$ as the mean number of deaths per corps-year.

```
> # goodness-of-fit test
> tab <- as.data.frame(HorseKicks, stringsAsFactors = FALSE)
> colnames(tab) <- c("nDeaths", "Freq")
> str(tab)

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

> (lambda <- weighted.mean(as.numeric(tab$nDeaths), w = tab$Freq))

[1] 0.61
```

From this, we can calculate the probabilities (`phat`) of `k = 0:4` deaths, and hence the expected (`exp`) frequencies in a Poisson distribution.

```
> phat <- dpois(0 : 4, lambda = lambda)
> exp <- sum(tab$Freq) * phat
> chisq <- (tab$Freq - exp)^2 / exp
>
> GOF <- data.frame(tab, phat, exp, chisq)
> GOF

  nDeaths Freq      phat       exp     chisq
1       0  109 0.5433509 108.67017 0.0010011
2       1   65 0.3314440  66.28881 0.0250573
3       2   22 0.1010904  20.21809 0.1570484
4       3    3 0.0205551   4.11101 0.3002534
5       4    1 0.0031346   0.62693 0.2220057
```

Finally, the Pearson χ^2 is just the sum of the `chisq` values and `pchisq()` is used to calculate the p-value of this test statistic—the probability of obtaining this χ^2 or a more extreme value if our assumption on the underlying distribution is true.

```
> sum(chisq)  # chi-square value

[1] 0.70537

> pchisq(sum(chisq), df = nrow(tab) - 2, lower.tail = FALSE)

[1] 0.87194
```

The result, $\chi^2_3 = 0.70537$, shows an extremely good fit of these data to the Poisson distribution, perhaps exceptionally so.[12]

△

3.3.1 R tools for discrete distributions

In R, the function `fitdistr()` in the MASS package is a basic work horse for fitting a variety of distributions by maximum likelihood and other methods, giving parameter estimates and standard errors. Among discrete distributions, the binomial, Poisson and geometric distributions have closed-form maximum likelihood estimates; the negative binomial distribution, parameterized by (n, μ), is estimated iteratively by direct optimization.

These basic calculations are extended and enhanced for one-way discrete distributions in the vcd function `goodfit()`, which computes the fitted values of a discrete distribution (either Poisson, binomial, or negative binomial) to the count data. If the parameters are not specified they are estimated either by maximum likelihood (ML) or Minimum Chi-squared. `print()` and `summary()` methods for the "goodfit" objects give, respectively, a table of observed and fitted frequencies, and the Pearson and/or likelihood ratio goodness-of-fit statistics. Plotting methods for visualizing the discrepancies between observed and fitted frequencies are described and illustrated in Section 3.3.2.

EXAMPLE 3.13: Families in Saxony

This example uses `goodfit()` to fit the binomial to the distribution of the number of male children in families of size 12 in Saxony. Note that for the binomial, both n and p are considered as parameters, and by default n is taken as the maximum count.

```
> data("Saxony", package = "vcd")
> Sax_fit <- goodfit(Saxony, type = "binomial")
> unlist(Sax_fit$par) # estimated parameters

    prob     size
 0.51922 12.00000
```

So, we estimate the probability of a male in these families to be $p = 0.519$, a value that is quite close to the value found in Arbuthnot's data ($p = 0.517$).

It is useful to know that `goodfit()` returns a list structure of named components that are used by method functions for class "goodfit" objects. The `print.goodfit()` method prints the table of observed and fitted frequencies. `summary.goodfit()` calculates and prints the likelihood ratio χ^2 GOF test when the ML estimation method is used.

```
> names(Sax_fit)      # components of "goodfit" objects

[1] "observed" "count"    "fitted"   "type"     "method"
[6] "df"       "par"

> Sax_fit             # print method


Observed and fitted values for binomial distribution
with parameters estimated by `ML'

 count observed     fitted pearson residual
     0        3    0.93284          2.14028
```

[12] An exceptionally good fit occurs when the p-value for the test χ^2 statistic is so high, as to suggest that something unreasonable under random sampling might have occurred. The classic example of this is the controversy over Gregor Mendel's experiments of cross-breeding garden peas with various observed (phenotype) characteristics, where R. A. Fisher (1936a) suggested that observed frequencies of combinations like (smooth/wrinkled), (green/yellow) in a second generation were uncomfortably too close to the 3 : 1 ratio predicted by genetic theory.

```
     1       24   12.08884          3.42580
     2      104   71.80317          3.79963
     3      286  258.47513          1.71205
     4      670  628.05501          1.67371
     5     1033 1085.21070         -1.58490
     6     1343 1367.27936         -0.65661
     7     1112 1265.63031         -4.31841
     8      829  854.24665         -0.86380
     9      478  410.01256          3.35761
    10      181  132.83570          4.17896
    11       45   26.08246          3.70417
    12        7    2.34727          3.03687

> summary(Sax_fit)    # summary method


 Goodness-of-fit test for binomial distribution

                      X^2 df    P(> X^2)
Likelihood Ratio 97.007 11 6.9782e-16
```

Note that the GOF test gives a highly significant p-value (pratically zero), indicating significant lack of fit to the binomial distribution.[13] Some further analysis of this result is explored in examples below. △

Example 3.14: Weldon's dice

Weldon's dice data, explored in Example 3.3, are also expected to follow a binomial distribution, here with $p = \frac{1}{3}$. However, as given in the data set *WeldonDice*, the frequencies for counts 10–12 were grouped as "10+." In this case, it is necessary to supply the correct value of $n = 12$ as the value of the `size` parameter in the call to `goodfit()`.

```
> data("WeldonDice", package = "vcd")
> dice_fit <- goodfit(WeldonDice, type = "binomial",
+                     par = list(size = 12))
> unlist(dice_fit$par)

    prob     size
 0.33769 12.00000
```

The probability of a success (a 5 or 6) is estimated as $\hat{p} = 0.3377$, not far from the theoretical value, $p = 1/3$.

```
> print(dice_fit, digits = 0)


Observed and fitted values for binomial distribution
with parameters estimated by `ML'

 count observed fitted pearson residual
     0      185    187               -0
     1     1149   1147                0
     2     3265   3216                1
     3     5475   5465                0
     4     6114   6269               -2
     5     5194   5114                1
     6     3067   3042                0
```

[13] A handy rule-of-thumb is to think of the ratio of χ^2/df, because, under the null hypothesis of acceptable fit, $\mathcal{E}(\chi^2/df) = 1$, so ratios exceeding ≈ 2.5 are troubling. Here, the ratio is $97/11 = 8.8$, so the lack of fit is substantial.

```
      7      1331   1330                 0
      8       403    424                -1
      9       105     96                 1
     10        18     15                 1
     11         0      1                -1
     12         0      0                -0

> summary(dice_fit)


 Goodness-of-fit test for binomial distribution

                      X^2 df  P(> X^2)
Likelihood Ratio 11.506  9   0.2426
```

Here, we find an acceptable fit for the binomial distribution. △

EXAMPLE 3.15: Death by horse kick

This example reproduces the calculations done "manually" in Example 3.12 above. We fit the Poisson distribution to the *HorseKicks* data by specifying `type = "poisson"` (actually, that is the default for `goodfit()`).

```
> data("HorseKicks", package = "vcd")
> HK_fit <- goodfit(HorseKicks, type = "poisson")
> HK_fit$par

$lambda
[1] 0.61

> HK_fit


Observed and fitted values for poisson distribution
with parameters estimated by `ML'

 count observed    fitted pearson residual
     0      109 108.67017          0.03164
     1       65  66.28881         -0.15830
     2       22  20.21809          0.39629
     3        3   4.11101         -0.54795
     4        1   0.62693          0.34142
```

The `summary` method uses the LR test by default, so the `X^2` value reported below differs slightly from the Pearson χ^2 value shown earlier.

```
> summary(HK_fit)


 Goodness-of-fit test for poisson distribution

                       X^2 df  P(> X^2)
Likelihood Ratio 0.86822  3   0.83309
```

△

EXAMPLE 3.16: Federalist Papers

In Example 3.5 we examined the distribution of the marker word "may" in blocks of text in the *Federalist Papers* written by James Madison. A naive hypothesis is that these occurrences might follow a Poisson distribution, that is, as independent occurrences with constant probability across the 262 blocks of text. Using the same methods as above, we fit these data to the Poisson distribution:

```
> data("Federalist", package = "vcd")
> Fed_fit0 <- goodfit(Federalist, type = "poisson")
> unlist(Fed_fit0$par)

 lambda
0.65649

> Fed_fit0


Observed and fitted values for poisson distribution
with parameters estimated by `ML'

 count observed     fitted pearson residual
     0      156 135.891389         1.724988
     1       63  89.211141        -2.775086
     2       29  29.283046        -0.052306
     3        8   6.407995         0.628903
     4        4   1.051694         2.874934
     5        1   0.138085         2.319483
     6        1   0.015109         7.620613
```

The GOF test below shows a substantial lack of fit, rejecting the assumptions of the Poisson model.

```
> 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.

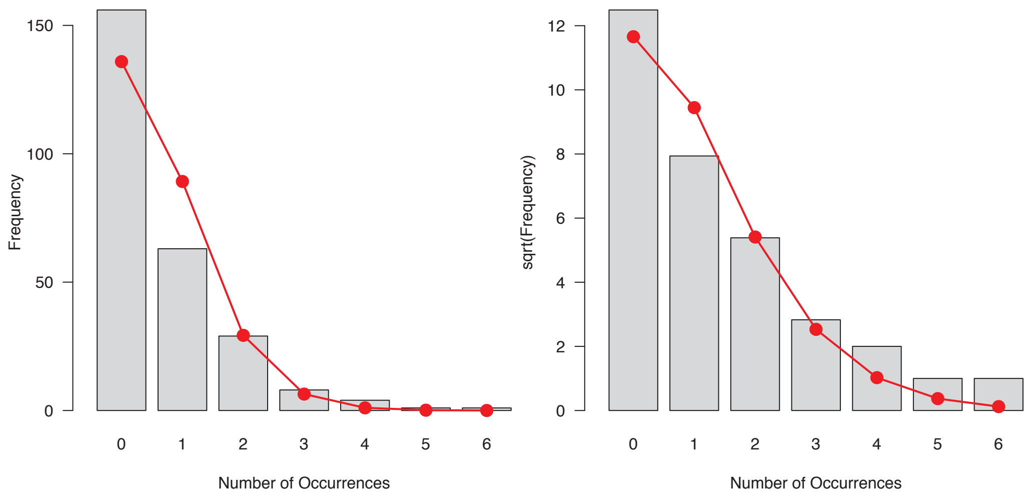


Figure 3.14: Plots for the Federalist Papers data, fitting the Poisson model. Each panel shows the observed frequencies as bars and the fitted frequencies as a smooth curve. Left: raw frequencies; right: plotted on a square-root scale to emphasize smaller frequencies.

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 `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.

```
> plot(Fed_fit0, scale = "raw", type = "standing")
> plot(Fed_fit0, type = "standing")
```

Additional improvements over the standard plot on the scale of raw frequencies are shown in Figure 3.15, both of which use the square root scale. The left panel moves the rootogram bars so their tops are at the expected frequencies (giving a ***hanging rootogram***). This has the advantage that we can more easily judge the pattern of departures against the horizontal reference line at 0, than against the curve.

```
> plot(Fed_fit0, type = "hanging", shade = TRUE)
> plot(Fed_fit0, type = "deviation", shade = TRUE)
```

A final variation is to emphasize the differences between the observed and fitted frequencies by drawing the bars to show the gaps between the 0 line and the (observed−expected) difference (Figure 3.15, right).

[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.

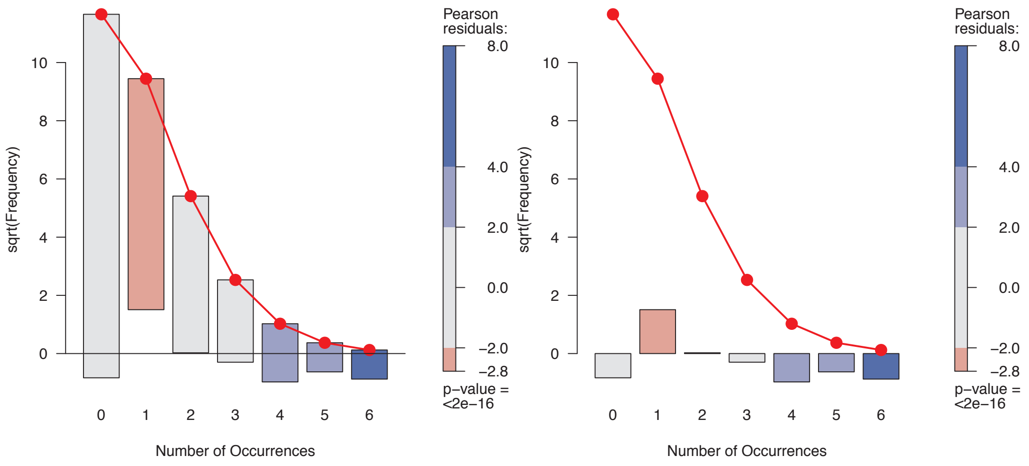


Figure 3.15: Plots for the Federalist Papers data, fitting the Poisson model. Left: hanging rootogram; Right: deviation rootogram. Color reflects the sign and magnitude of the contributions to lack of fit.

All of these plots are actually produced by the `rootogram()` function in vcd, the `plot()` method for "goodfit" objects. The default is `type = "hanging"`, and there are many options to control the plot details. For example, the plots in Figure 3.15 use `shade=TRUE` to color the bars according to the contribution to the Pearson chi-square.[15]

The plots in Figure 3.14 and Figure 3.15 used the ill-fitting Poisson model on purpose to highlight how these plots show the departure between the observed and fitted frequencies. Figure 3.16 compares this with the negative binomial model, `Fed_fit1`, which we saw has a much better, and acceptable fit.

```
> plot(Fed_fit0, main = "Poisson", shade = TRUE, legend = FALSE)
> plot(Fed_fit1, main = "Negative binomial", shade = TRUE, legend = FALSE)
```

Comparing the two plots in Figure 3.16, we can see that the Poisson model overestimates the frequency of counts $k = 1$ and underestimates the larger counts for $k = 4$–6 occurrences. The surprising feature here is that the greatest contribution to lack of fit for the Poisson model is the frequency for $k = 6$. The deviations for the negative binomial are small and unsystematic.

Finally, Figure 3.17 shows hanging rootograms for two atrociously bad models for the data on butterfly species in Malaya considered in Example 3.7. As we will see in Section 3.4, this long-tailed distribution is better approximated by the logarithmic series distribution, but this distribution is presently not handled by `goodfit()`.

```
> data("Butterfly", package = "vcd")
> But_fit1 <- goodfit(Butterfly, type = "poisson")
> But_fit2 <- goodfit(Butterfly, type = "nbinomial")
> plot(But_fit1, main = "Poisson", shade = TRUE, legend = FALSE)
> plot(But_fit2, main = "Negative binomial", shade = TRUE, legend = FALSE)
```

[15]The bipolar color scheme uses blue for positive standard Pearson residuals, $(n_k - N\hat{p}_k)/\sqrt{N\hat{p}_k}$, and red for negative residuals, with shading intensity proportional to the categorized absolute values shown in the legend.

3.4 Diagnosing discrete distributions: Ord plots

Ideally, the general form chosen for a discrete distribution should be dictated by substantive knowledge of a plausible mechanism for generating the data. When such knowledge is lacking, however, we may not know which distribution is most appropriate for some particular set of data. In these cases, the question is often turned around, so that we seek a distribution that fits well, and then try to understand the mechanism in terms of aspects of the underlying probability theory (independent trials, rare events, waiting-time to an occurrence, and so forth).

Although it is possible to fit each of several possibilities, the summary goodness-of-fit statistics can easily be influenced by one or two disparate cells, or additional (ignored or unknown) factors. One simple alternative is a plot suggested by Ord (1967), which may be used to diagnose the form of the discrete distribution.

Ord showed that a linear relationship of the form:

$$\frac{k\,p(k)}{p(k-1)} \equiv \frac{k\,n_k}{n_{k-1}} = a + b\,k \tag{3.12}$$

holds for each of the Poisson, binomial, negative binomial, and logarithmic series distributions, and these distributions are distinguished by the signs of the intercept, a, and slope, b, as shown in Table 3.11.

Table 3.11: Diagnostic slope and intercept for four discrete distributions. The ratios kn_k/n_{k-1} plotted against k should appear as a straight line, whose slope and intercept determine the particular distribution.

Slope	Intercept	Distribution	Parameter
(b)	(a)	(parameter)	estimate
0	$+$	Poisson (λ)	$\lambda = a$
$-$	$+$	Binomial (n, p)	$p = b/(b-1)$
$+$	$+$	Negative binomial (n, p)	$p = 1 - b$
$+$	$-$	Log. series (θ)	$\theta = b$
			$\theta = -a$

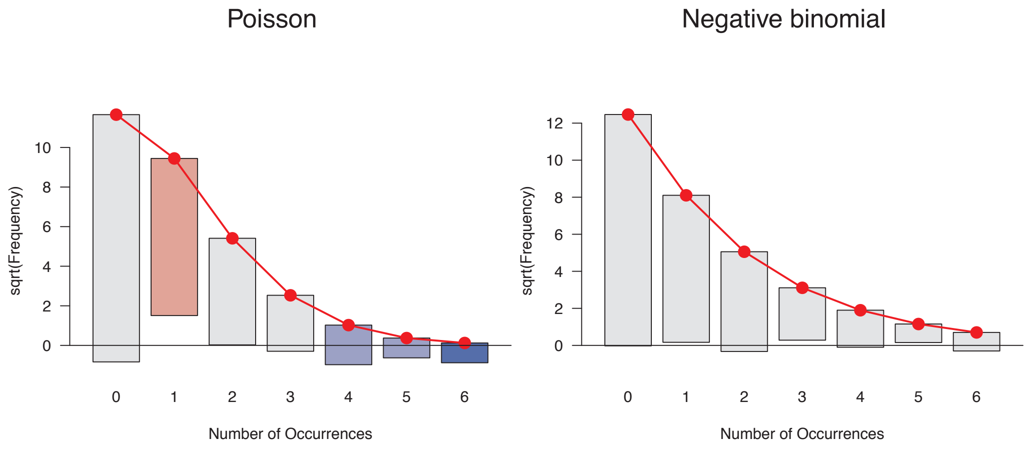


Figure 3.16: Hanging rootograms for the Federalist Papers data, comparing the Poisson and negative binomial models.

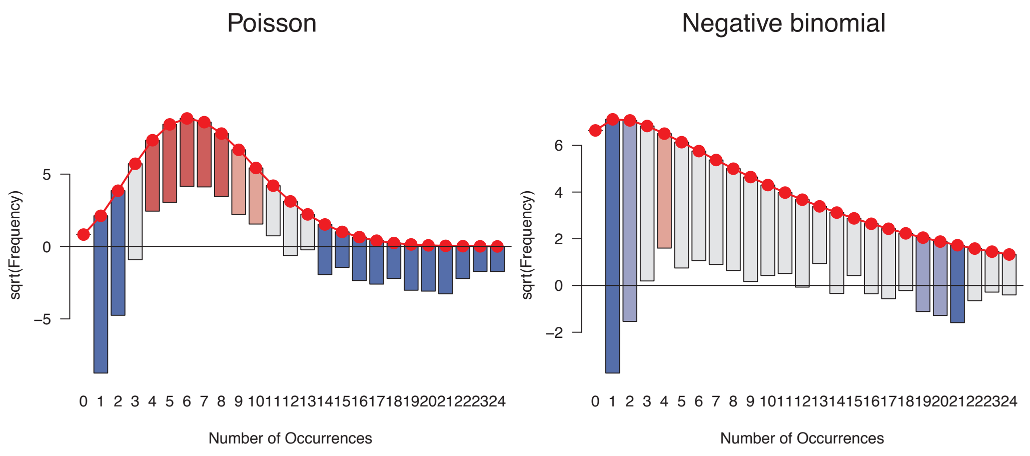


Figure 3.17: Hanging rootograms for the Butterfly data, comparing the Poisson and negative binomial models. The lack of fit for both is readily apparent.

The slope, b, in Eqn. (3.12) is zero for the Poisson, negative for the binomial, and positive for the negative binomial and logarithmic series distributions; the latter two are distinguished by their intercepts. In practical applications of this idea, the details are important: how to fit the line, and how to determine if the pattern of signs are sufficient to reasonably provide a diagnosis of the distribution type.

One difficulty in applying this technique is that the number of points (distinct values of k) in the Ord plot is often small, and the sampling variances of $k\, n_k / n_{k-1}$ can vary enormously. A little reflection indicates that points where n_k is small should be given less weight in determining the slope of the line (and hence determining the form of the distribution). In applications it has been found that using a weighted least squares fit of $k\, n_k / n_{k-1}$ on k, using weights of $w_k = \sqrt{n_k - 1}$ produces reasonably good automatic diagnosis of the form of a probability distribution. Moreover, to judge whether a coefficient is positive or negative, a small tolerance is used; if none of the distributions can be classified, no parameters are estimated. Caution is advised in accepting the conclusion, because it is based on these simple heuristics.

In the vcd package this method is implemented in the `Ord_plot()` function. The essential ideas are illustrated using the *Butterfly* data below, which produces Figure 3.18. Note that the function returns (invisibly) the values of the intercept and slope in the weighted least squares (WLS) regression.

```
> ord <- Ord_plot(Butterfly,
+                 main = "Butterfly species collected in Malaya",
+                 gp = gpar(cex = 1), pch = 16)
> ord

Intercept     Slope
 -0.70896   1.06082
```

In this plot, the black line shows the usual ordinary least squares (OLS) regression fit of frequency, n_k, on number of occurrences, k; the red line shows the weighted least squares fit, using weights of $\sqrt{n_k - 1}$. In this case, the two lines are fairly close together, in regards to their intercepts and slopes. The positive slope and negative intercept diagnoses this as a log-series distribution.

In other cases, the number of distinct points (values of k) is small, and the sampling variances of the ratios $k\, n_k / n_{k-1}$ can vary enormously. The following examples illustrate some other distributions and some of the details of the heuristics.

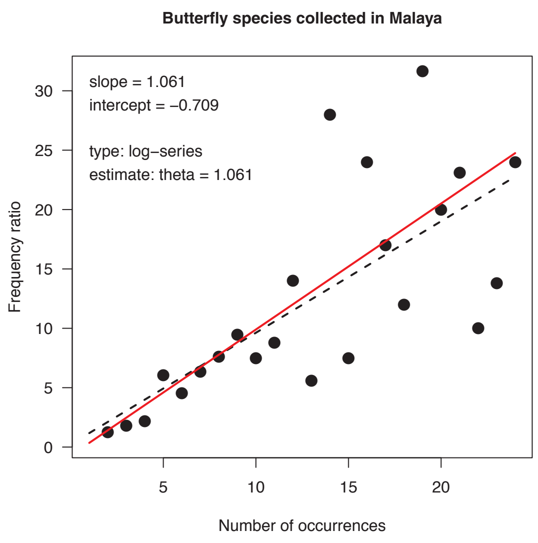


Figure 3.18: Ord plot for the Butterfly data. The slope and intercept in the plot correctly diagnoses the log-series distribution.

3.4.0.1 Ord plot examples

EXAMPLE 3.17: Death by horse kick

The results below show the calculations for the horse kicks data, with the frequency ratio $k\, n_k / n_{k-1}$ labeled `y`.

```
> data("HorseKicks", package = "vcd")
> nk <- as.vector(HorseKicks)
> k <- as.numeric(names(HorseKicks))
> nk1 <- c(NA, nk[-length(nk)])
> y <- k * nk / nk1
> weight <- sqrt(pmax(nk, 1) - 1)
> (ord_df <- data.frame(k, nk, nk1, y, weight))

  k  nk nk1       y  weight
1 0 109  NA      NA 10.3923
2 1  65 109 0.59633  8.0000
3 2  22  65 0.67692  4.5826
4 3   3  22 0.40909  1.4142
5 4   1   3 1.33333  0.0000

> coef(lm(y ~ k, weights = weight, data = ord_df))

(Intercept)           k
   0.656016   -0.034141
```

The weighted least squares line, with weights w_k, has a slope (-0.03) close to zero, indicating the Poisson distribution.[16] The estimate $\lambda = a = .656$ compares favorably with the maximum likelihood estimate (MLE), $\lambda = 0.610$ and the value from the Poissonness plot, shown in the following section. The call to `Ord_plot()` below produces Figure 3.19.

```
> Ord_plot(HorseKicks,
+          main = "Death by horse kicks", gp = gpar(cex = 1), pch = 16)
```

[16]The heuristic adopted in `Ord_plot()` uses a tolerance of 0.1 to decide if a coefficient is negative, zero, or positive.

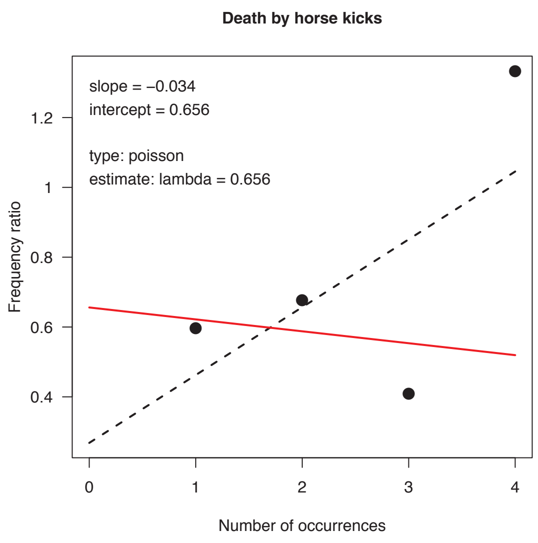


Figure 3.19: Ord plot for the HorseKicks data. The plot correctly diagnoses the Poisson distribution.

△

EXAMPLE 3.18: Federalist Papers

Figure 3.20 (left) shows the Ord plot for the *Federalist* data. The slope is positive, so either the negative binomial or log series are possible, according to Table 3.11. The intercept is essentially zero, which is ambiguous. However, the logarithmic series requires $b \approx -a$, so the negative binomial is a better choice. Mosteller and Wallace (1963, 1984) did in fact find a reasonably good fit to this distribution. Note that there is one apparent outlier, at $k = 6$, whose effect on the OLS line is to increase the slope and decrease the intercept. △

```
> Ord_plot(Federalist, main = "Instances of 'may' in Federalist Papers",
+          gp = gpar(cex = 1), pch = 16)
```

EXAMPLE 3.19: Women in queues

Jinkinson and Slater (1981) and Hoaglin and Tukey (1985) give the frequency distribution of the number of females observed in 100 queues of length 10 in a London Underground station, recorded in the data set *WomenQueue* in vcd.

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

nWomen
 0  1  2  3  4  5  6  7  8  9 10
 1  3  4 23 25 19 18  5  1  1  0
```

If it is assumed that people line up independently, and that men and women are equally likely to be found in a queue (not necessarily reasonable assumptions), then the number of women out of 10 would have a (symmetric) binomial distribution with parameters $n = 10$ and $p = \frac{1}{2}$. However, there is no real reason to expect that males and females are equally likely to be found in queues in the London underground, so we may be interested in estimating p from the data and determining if a binomial distribution fits.

```
> Ord_plot(WomenQueue, main = "Women in queues of length 10",
+          gp = gpar(cex = 1), pch = 16)
```

Figure 3.20 (right) shows the Ord plot for these data. The negative slope and positive intercept clearly diagnose this distribution as binomial. The rough estimate of $\hat{p} = b/(1-b) = 0.53$ indicates that women are slightly more prevalent than men in these data for the London underground. △

3.4.0.2 Limitations of Ord plots

Using a single simple diagnostic plot to determine one of four common discrete distributions is advantageous, but your enthusiasm should be dampened by several weaknesses:

- The Ord plot lacks resistance, because a single discrepant frequency affects the points n_k/n_{k-1} for both k and $k+1$.
- The sampling variance of $k\, n_k/n_{k-1}$ fluctuates widely (Hoaglin and Tukey, 1985, Jinkinson and Slater, 1981). The use of weights w_k helps, but is purely a heuristic device. The `Ord_plot()` function explicitly shows both the OLS line and the WLS line, which provides some indication of the effect of the points on the estimation of slope and intercept.

3.5 Poissonness plots and generalized distribution plots

The ***Poissonness plot*** (Hoaglin, 1980) is a robust plot to sensitively determine how well a one-way table of frequencies follows a Poisson distribution. It plots a quantity called a against k, designed so that the result will be points along a straight line when the data follow a Poisson distribution. When the data deviate from a Poisson, the points will be curved. Hoaglin and Tukey (1985) developed similar plots for other discrete distributions, including the binomial, negative binomial, and logarithmic series distributions. We first describe the features and construction of these plots for the Poisson distribution; then (Section 3.5.4) the extension to other distributions.

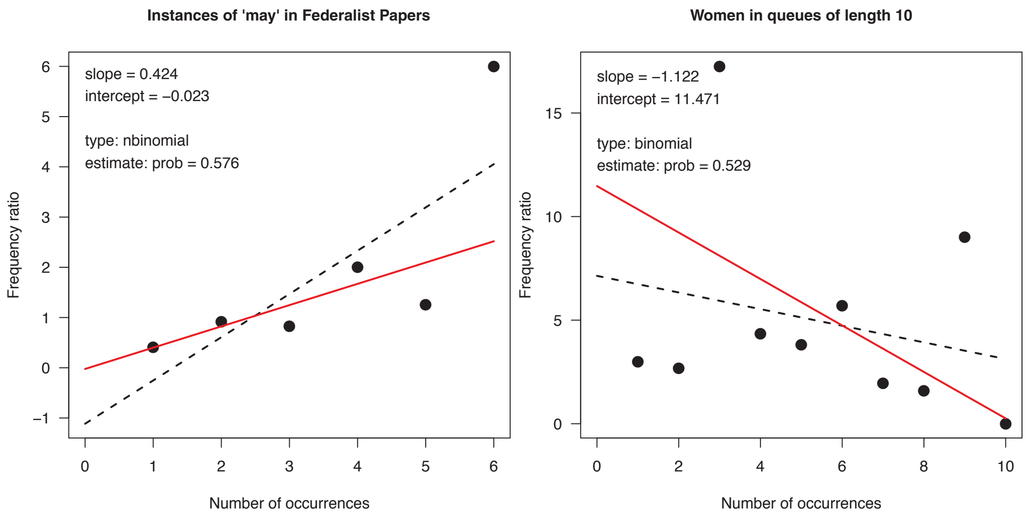


Figure 3.20: Ord plots for the Federalist (left) and WomenQueue (right) data sets.

3.5.1 Features of the Poissonness plot

The Poissonness plot has the following desirable features:

- ***Resistance***: a single discrepant value of n_k affects only the point at value k. (In the Ord plot it affects each of its neighbors.)

- ***Comparison standard***: An approximate confidence interval can be found for each point, indicating its inherent variability and helping to judge whether each point is discrepant.

- ***Influence***: Extensions of the method result in plots that show the effect of each point on the estimate of the main parameter of the distribution (λ in the Poisson).

3.5.2 Plot construction

Assume, for some fixed λ, each observed frequency, n_k equals the expected frequency, $m_k = Np_k$. Then, setting $n_k = Np_k = Ne^{-\lambda}\,\lambda^k/k!$, and taking logs of both sides gives

$$\log(n_k) = \log\, N - \lambda + k\, \log\, \lambda - \log\, k!\ .$$

This can be rearranged to a linear equation in k,

$$\phi\,(n_k) \equiv \log\left(\frac{k!\; n_k}{N}\right) = -\lambda + (\log\, \lambda)\, k\ . \tag{3.13}$$

The left side of Eqn. (3.13) is called the ***count metameter***, and denoted $\phi\,(n_k)$. Hence, plotting $\phi(n_k)$ against k should give a straight line of the form $\phi(n_k) = a + bk$ with

- slope = $\log\, \lambda$
- intercept = $-\lambda$

when the observed frequencies follow a Poisson distribution. If the points in this plot are close enough to a straight line, then an estimate of λ may be obtained from the slope b of the line, and $\hat{\lambda} = e^b$ should be reasonably close in value to the MLE of λ, $\hat{\lambda} = \bar{x}$. In this case, we might as well use the MLE as our estimate.

3.5.2.1 Leveled plot

If we have a preliminary estimate λ_0 of λ, we can use this to give a new plot where the reference line is horizontal, making comparison of the points with the line easier. In this leveled plot the vertical coordinate $\phi(n_k)$ is modified to

$$\phi'(n_k) = \phi(n_k) + \lambda_0 - k \log \lambda_0\ . \tag{3.14}$$

When the data follow a Poisson distribution with parameter λ, the modified plot will have

- slope = $\log \lambda - \log \lambda_0 = \log(\lambda/\lambda_0)$
- intercept = $\lambda_0 - \lambda$

In the ideal case, where our estimate of λ_0 is close to the true λ, the line will be approximately horizontal at $\phi(n_k)' = 0$. The modified plot is particularly useful in conjunction with the confidence intervals for individual points described below.

3.5.2.2 Confidence intervals

The goal of the Poissonness plot is to determine whether the points are “sufficiently linear” to conclude that the Poisson distribution is adequate for the data. Confidence intervals for the points can help you decide, and also show the relative precision of the points in these plots.

For example, when one or two points deviate from an otherwise nearly linear relation, it is helpful to determine whether the discrepancy is consistent with chance variation. As well, we must recognize that classes with small frequencies n_k are less precise than classes with large frequencies.

Hoaglin and Tukey (1985) develop approximate confidence intervals for $\log(m_k)$ for each point in the Poissonness plot. These are calculated as

$$\phi\left(n_k^*\right) \pm h_k \tag{3.15}$$

where the count metameter function is calculated using a modified frequency n_k^*, defined as

$$n_k^* = \begin{cases} n_k - .8n_k - .67 & n \geq 2 \\ 1/e & n = 1 \\ \text{undefined} & n = 0 \end{cases}$$

and h_k is the half-width of the 95% confidence interval,

$$h_k = 1.96 \frac{\sqrt{1-\widehat{p}_k}}{[n_k - (.25\widehat{p}_k + .47)\sqrt{n_k}]^{1/2}}$$

and $\hat{p}_k = n_k/N$. A more modern approach could use a bootstrap estimate.

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` = $\phi\left(n_k\right)$. The smaller filled points show the centers of the confidence intervals, `CI.center` = $\phi\left(n_k^*\right)$ (Eqn. (3.15)), and the dashed lines show the extent of the confidence intervals.

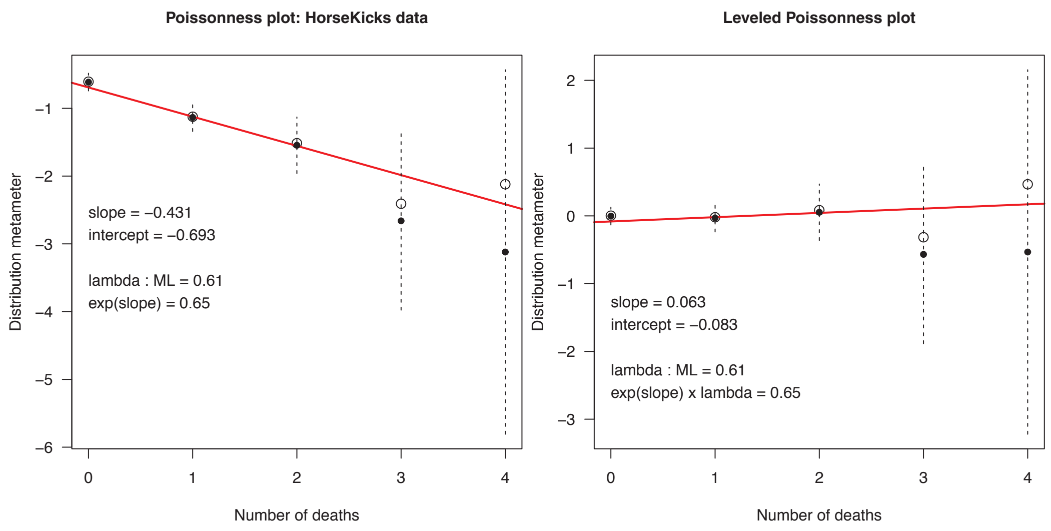


Figure 3.21: Poissonness plots for the HorseKick data. Left: standard plot; right: leveled plot.

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.

$\triangle$

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 $p = e^b/(1+e^b)$.

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.

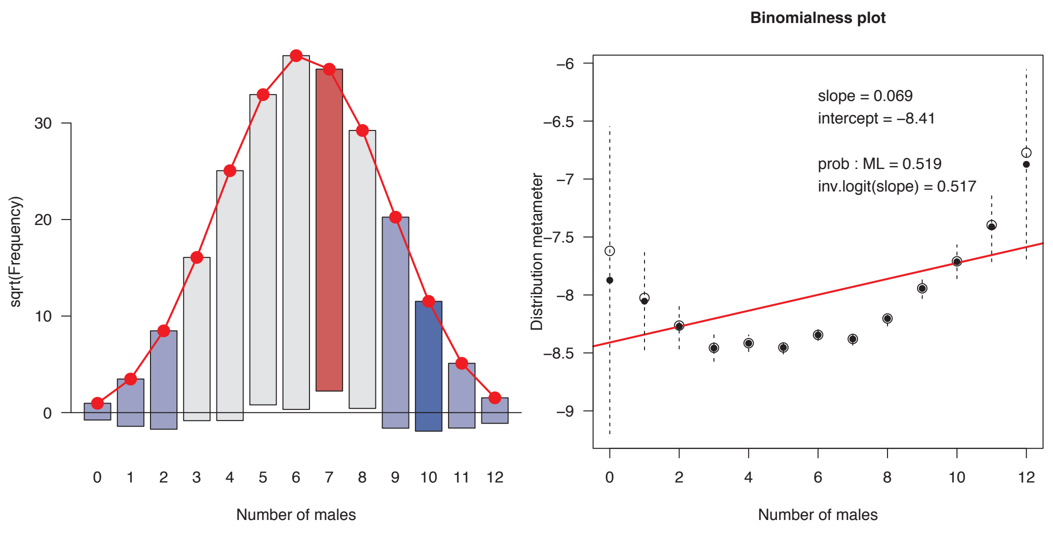


Figure 3.22: Diagnostic plots for males in Saxony families. Left: `goodfit()` plot; right: `distplot()` plot. Both plots show heavier tails than in a binomial distribution.

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."

```
> 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")
```

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	$e^{-\lambda}\lambda^k/k!$	$\log(k!n_k^*/N)$	$\log(\lambda)$	-λ
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.

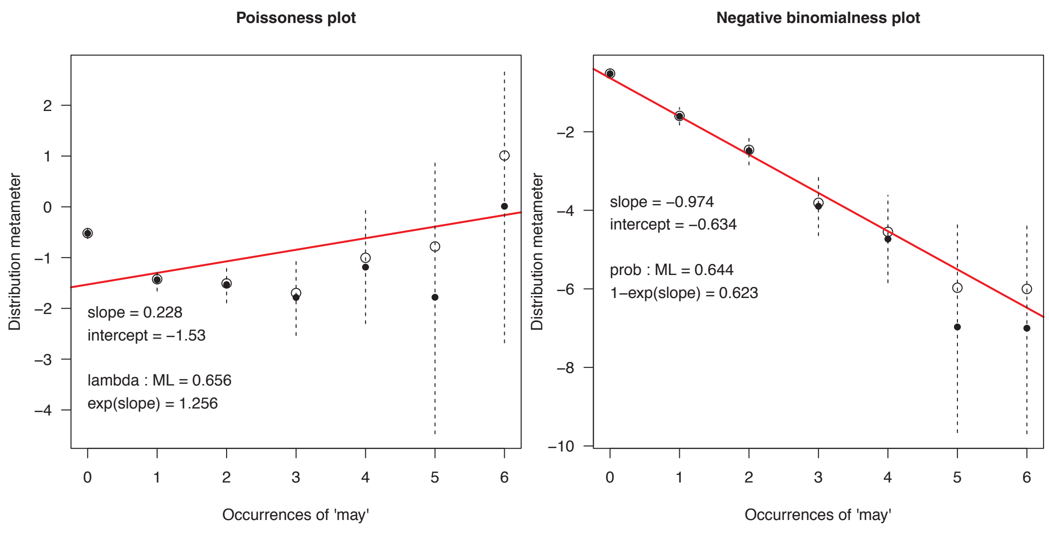


Figure 3.23: Diagnostic plots for the Federalist Papers data. Left: Poissonness plot; right: negative binomialness plot.

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'")
```

△

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()`.

[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.

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	$k, k\log(k) + (n-k)\log(n-k)$	$\log\binom{n}{k}$

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] 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.

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

```
> # 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()`.

```
> # double binomial, (12, p, psi)
> Sax.df$YlogitY <-
+   Males      * log(ifelse(Males == 0, 1, Males)) +
+         (12-Males) * log(ifelse(12-Males == 0, 1, 12-Males))
>
> Sax.dbin <- glm(Families ~ Males + YlogitY, offset = lchoose(12,0:12),
+         family = poisson, data = Sax.df)
> coef(Sax.dbin)

(Intercept)       Males     YlogitY
  -3.096918    0.065977    0.140205

> LRstats(Sax.bin, Sax.dbin)

Likelihood summary table:
         AIC BIC LR Chisq Df Pr(>Chisq)
Sax.bin  191 192     97.0 11      7e-16 ***
Sax.dbin 109 111     13.1 10       0.22
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

From the above, we can see that the double binomial model `Sax.dbin` with one more parameter is significantly better than the simple binomial and represents an adequate fit to the data. The table below displays the fitted values and standardized residuals for both models.

```
> results <- data.frame(Sax.df,
+           fit.bin = fitted(Sax.bin), res.bin = rstandard(Sax.bin),
+           fit.dbin = fitted(Sax.dbin), res.dbin = rstandard(Sax.dbin))
> print(results, digits = 2)

   Males Families YlogitY fit.bin res.bin fit.dbin res.dbin
1      0        3      30    0.93    1.70      3.0    0.026
2      1       24      26   12.09    3.05     23.4    0.136
3      2      104      24   71.80    3.71    104.3   -0.036
4      3      286      23  258.48    1.87    307.8   -1.492
5      4      670      22  628.06    1.94    652.9    0.778
```

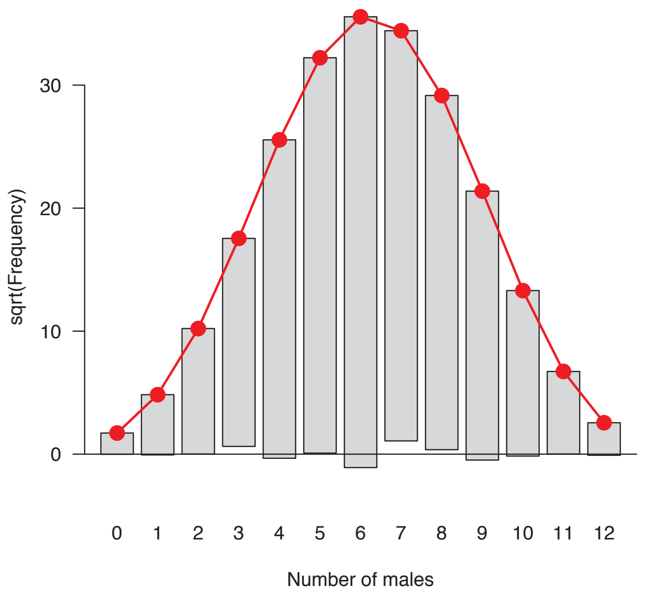


Figure 3.24: Rootogram for the double binomial model for the Saxony data. This now fits well in the tails of the distribution.

```
6       5     1033      22 1085.21   -1.87   1038.5   -0.202
7       6     1343      22 1367.28   -0.75   1264.2    2.635
8       7     1112      22 1265.63   -5.09   1185.0   -2.550
9       8      829      22  854.25   -1.03    850.1   -0.846
10      9      478      23  410.01    3.75    457.2    1.144
11     10      181      24  132.84    4.23    176.8    0.371
12     11       45      26   26.08    3.42     45.2   -0.039
13     12        7      30    2.35    2.45      6.5    0.192
```

Finally, Figure 3.24 shows the rootogram for the double binomial, which can be compared with the binomial model shown in Figure 3.22. We can see that the fit is now quite good, particularly in the tails. The positive coefficient for the term `YlogitY` gives additional weight in the tails.

```
> with(results, vcd::rootogram(Families, fit.dbin, Males,
+                              xlab = "Number of males"))
```

△

3.6.1 Covariates, overdispersion, and excess zeros

All of the examples in this chapter are somewhat special, in that in each case the data consist only of a one-way frequency distribution of a basic count variable. In more general and realistic settings, there may also be one or more explanatory variables or ***covariate***s that influence the frequency distributions of the counts. For example, in the `Saxony` data, the number of boys in families of size 12 was aggregated over the years 1876–1885, and it is possible that any deviation from a binomial distribution could be due to variation over time or unmeasured predictors (e.g., rural vs. urban, age of parents).

This is where the generalized linear model approach introduced here (treated in detail in Chapter 11), begins to shine—because it allows such covariates to be taken into account, and then questions regarding the *form* of the distribution pertain only to the variation of the frequencies not fitted

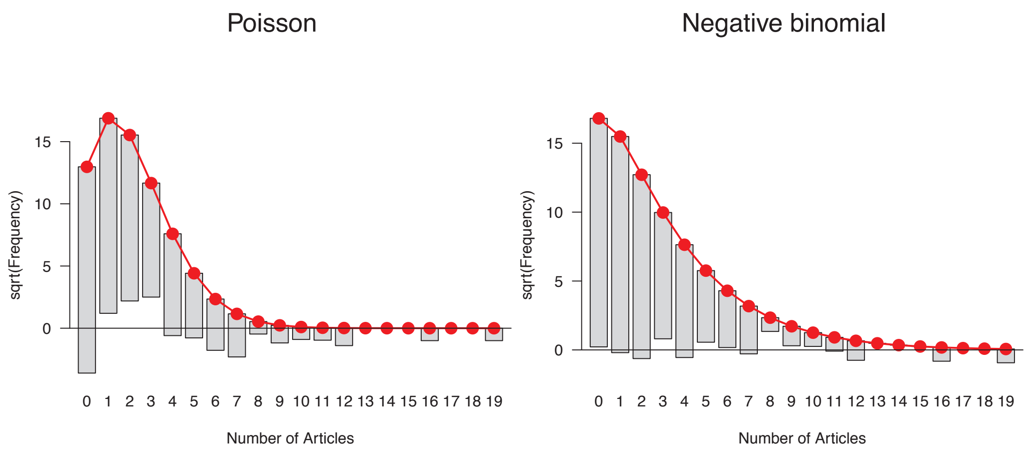


Figure 3.25: Hanging rootograms for publications by PhD candidates, comparing the Poisson and negative binomial models. The Poisson model clearly does not fit. The the negative binomial is better, but still has significant lack of fit.

by the model. The next example illustrates what can go wrong when important predictors are omitted from the analysis.

Example 3.24: Publications of PhD candidates

Long (1990, 1997) gave data on the number of publications by 915 doctoral candidates in biochemistry in the last three years of their PhD studies, contained in the data set *PhdPubs* in vcdExtra. The data set also includes information on gender, marital status, number of young children, prestige of the doctoral department, and number of publications by the student's mentor. The frequency distribution of number of publications by these students is shown below.

```
> data("PhdPubs", package = "vcdExtra")
> table(PhdPubs$articles)


  0   1   2   3   4   5   6   7   8   9  10  11  12  16  19
275 246 178  84  67  27  17  12   1   2   1   1   2   1   1
```

The naive approach, ignoring the potential predictors, is just to try fitting various probability models to this one-way distribution. Rootograms for the simpler Poisson distribution and the negative binomial that allows for overdispersion are shown in Figure 3.25.

```
> library(vcd)
> plot(goodfit(PhdPubs$articles), xlab = "Number of Articles",
+      main = "Poisson")
> plot(goodfit(PhdPubs$articles, type = "nbinomial"),
+      xlab = "Number of Articles", main = "Negative binomial")
```

From these plots it is clear that the Poisson distribution doesn't fit well at all, because there is a large excess of zero counts—candidates with no publications, and most of the counts of four or more publications are larger than the Poisson model predicts. The fit of the negative binomial model in the right panel of Figure 3.25 looks much better, except that for eight or more publications, there is a systematic tendency of overfitting for 8–10 and underfittting for the observed counts of 12 or more. This lack of fit is confirmed by the formal test.

```
> summary(goodfit(PhdPubs$articles, type = "nbinomial"))


 Goodness-of-fit test for nbinomial distribution

                      X^2 df  P(> X^2)
Likelihood Ratio 31.098 12 0.0019033
```

The difficulty with this simple analysis is not only that it ignores the possible predictors of publishing by these PhD candidates, but also, by doing so, it prevents a better, more nuanced explanation of the phenomenon under study. This example is re-visited in Chapter 11, Example 11.1, where we consider generalized linear models taking potential predictors into account, as well as extended *zero-inflated* models allowing special consideration of zero counts. △

3.7 Chapter summary

- Discrete distributions typically involve basic *counts* of occurrences of some event occurring with varying *frequency*. The ideas and methods for one-way tables described in this chapter are building blocks for the analysis of more complex data.
- The most commonly used discrete distributions include the binomial, Poisson, negative binomial, geometric, and logarithmic series distributions. Happily, these are all members of a family called the power series distributions. Methods of fitting an observed data set to any of these distributions are described, and implemented in the `goodfit()` function.
- After fitting an observed distribution it is useful to plot the observed and fitted frequencies. Several ways of making these plots are described, and implemented in the `rootogram()` function.
- A heuristic graphical method for identifying which discrete distribution is most appropriate for a given set of data involves plotting ratios kn_k/n_{k-1} against k. These plots are constructed by the function `Ord_plot()`.
- A more robust plot for a Poisson distribution involves plotting a count metameter, $\phi(n_k)$ against k, which gives a straight line (whose slope estimates the Poisson parameter) if the data follow a Poisson distribution. This plot provides robust confidence intervals for individual points and provides a means to assess the influence of individual points on the Poisson parameter. These plots are provided by the function `distplot()`.
- The ideas behind the Poissonness plot can be applied to the other discrete distributions.

3.8 Lab exercises

Exercise 3.1 The `Arbuthnot` data in HistData (Example 3.1) also contains the variable `Ratio`, giving the ratio of male to female births.

(a) Make a plot of `Ratio` over `Year`, similar to Figure 3.1. What features stand out? Which plot do you prefer to display the tendency for more male births?
(b) Plot the total number of christenings, `Males + Females` or `Total` (in 000s) over time. What unusual features do you see?

Exercise 3.2 Use the graphical methods illustrated in Section 3.2 to plot a collection of geometric distributions for $p = 0.2, 0.4, 0.6, 0.8$, over a range of values of $k = 0, 1, \dots 10$.

(a) With `xyplot()`, try the different plot formats using points connected with lines, as in Figure 3.9, or using points and lines down to the origin, as in the panels of Figure 3.10.
(b) Also with `xyplot()`, produce one version of a multi-line plot in a single panel that you think shows well how these distributions change with the probability p of success.
(c) Do the same in a multi-panel version, conditional on p.

Exercise 3.3 Use the data set *`WomenQueue`* to:

(a) Produce plots analogous to those shown in Section 3.1 (some sort of bar graph of frequencies).
(b) Check for goodness-of-fit to the binomial distribution using the `goodfit()` methods described in Section 3.3.2.
(c) Make a reasonable plot showing departure from the binomial distribution.
(d) Suggest some reasons why the number of women in queues of length 10 might depart from a binomial distribution, $\text{Bin}(n = 10, p = 1/2)$.

Exercise 3.4 Continue Example 3.13 on the distribution of male children in families in Saxony by fitting a binomial distribution, $\text{Bin}(n = 12, p = \frac{1}{2})$, specifying equal probability for boys and girls. [*Hint*: you need to specify both `size` and `prob` values for `goodfit()`.]

(a) Carry out the GOF test for this fixed binomial distribution. What is the ratio of χ^2/df? What do you conclude?
(b) Test the additional lack of fit for the model $\text{Bin}(n = 12, p = \frac{1}{2})$ compared to the model $\text{Bin}(n = 12, p = \hat{p})$ where $\hat{p}$ is estimated from the data.
(c) Use the `plot.gootfit()` method to visualize these two models.

Exercise 3.5 For the *`Federalist`* data, the examples in Section 3.3.1 and Section 3.3.2 showed the negative binomial to provide an acceptable fit. Compare this with the simpler special case of geometric distribution, corresponding to $n = 1$.

(a) Use `goodfit()` to fit the geometric distribution. [Hint: use `type="nbinomial"`, but specify `size=1` as a parameter.]
(b) Compare the negative binomial and the geometric models statistically, by a likelihood-ratio test of the difference between these two models.
(c) Compare the negative binomial and the geometric models visually by hanging rootograms or other methods.

Exercise 3.6 Mosteller and Wallace (1963, Table 2.4) give the frequencies, n_k, of counts $k = 0, 1, \dots$ of other selected marker words in 247 blocks of text known to have been written by Alexander Hamilton. The data below show the occurrences of the word *upon*, that Hamilton used much more than did James Madison.

```
> count <- 0 : 5
> Freq <- c(129, 83, 20, 9, 5, 1)
```

(a) Read these data into R and construct a one-way table of frequencies of counts or a matrix or data frame with frequencies in the first column and the corresponding counts in the second column, suitable for use with `goodfit()`.
(b) Fit and plot the Poisson model for these frequencies.
(c) Fit and plot the negative binomial model for these frequencies.
(d) What do you conclude?

Exercise 3.7 The data frame *Geissler* in the vcdExtra package contains the complete data from Geissler's (1889) tabulation of family sex composition in Saxony. The table below gives the number of boys in families of size 11.

boys	0	1	2	3	4	5	6	7	8	9	10	11
Freq	8	72	275	837	1,540	2,161	2,310	1,801	1,077	492	93	24

(a) Read these data into R.
(b) Following Example 3.13, use `goodfit()` to fit the binomial model and plot the results. Is there an indication that the binomial does not fit these data?
(c) Diagnose the form of the distribution using the methods described in Section 3.4.
(d) Try fitting the negative binomial distribution, and use `distplot()` to diagnose whether the negative binomial is a reasonable fit.

Exercise 3.8 The data frame *Bundesliga* gives a similar data set to that for UK soccer scores (*UKSoccer*) examined in Example 3.9, but over a wide range of years. The following lines calculate a two-way table, `BL1995`, of home-team and away-team goals for the 306 games in the year 1995.

```
> data("Bundesliga", package = "vcd")
> BL1995 <- xtabs(~ HomeGoals + AwayGoals, data = Bundesliga,
+                   subset = (Year == 1995))
> BL1995

         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` does not, and cannot, occur in this table, so the distribution is restricted to positive counts. Such distributions are called ***zero-truncated distribution***s. Such distributions are like the ordinary ones, but with the probability of zero being zero. Thus the other probabilities are scaled up (i.e., divided by $1 - \Pr(Y = 0)$) so they sum to 1.

years	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
inducted	46	10	8	7	8	4	2	4	6	3	3	1	4	1	2

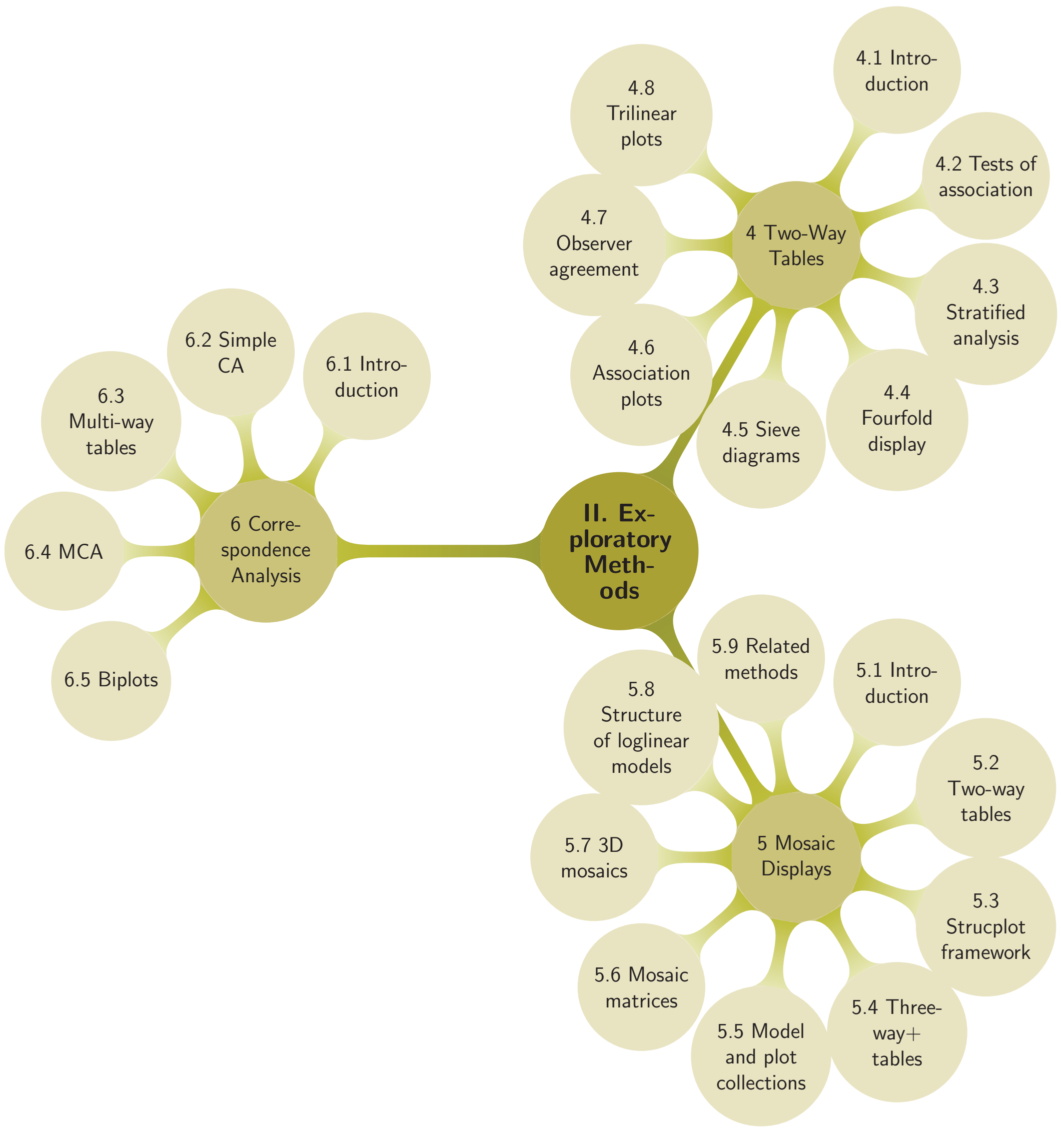
(a) For the Poisson distribution, show that the zero-truncated probability function can be expressed in the form

$$\Pr\{X = k \,|\, k > 0\}) = \frac{1}{1 - e^{-\lambda}} \times \frac{e^{-\lambda}\,\lambda^k}{k!} \qquad k = 1, 2, \ldots$$

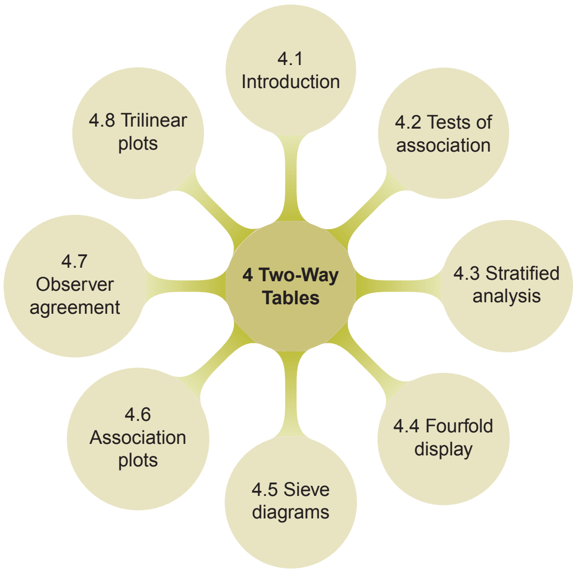
(b) Show that the mean is $\lambda/(1 - \exp(-\lambda))$.
(c) Enter these data into R as a one-way table, and use `goodfit()` to fit the standard Poisson distribution, as if you hadn't encountered the problem of zero truncation.

Part II

Exploratory and Hypothesis-Testing Methods



This page intentionally left blank



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 us 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.

4.1 Introduction

Tables are like cobwebs, like the sieve of Danaides; beautifully reticulated, orderly to look upon, but which will hold no conclusion. Tables are abstractions, and the object a most concrete one, so difficult to read the essence of.

From *Chartism* by Thomas Carlyle (1840), Chapter II, Statistics

Most methods of statistical analysis are concerned with understanding relationships or dependence among variables. With categorical variables, these relationships are often studied from data that has been summarized by a ***contingency table*** in table form or frequency form, giving the frequencies of observations cross-classified by two or more such variables. As Thomas Carlyle said, it is often difficult to appreciate the message conveyed in numerical tables.

This chapter is concerned with simple graphical methods for understanding the association between two categorical variables. Some examples are also presented that involve a third, ***stratifying variable***, where we wish to determine if the relationship between two primary variables is the same or different for all levels of the stratifying variable. More general methods for fitting models and displaying associations for three-way and larger tables are described in Chapter 5.

In Section 4.2, we describe briefly some numerical and statistical methods for testing whether an association exists between two variables, and measures for quantifying the strength of this association. In Section 4.3 we extend these ideas to situations where the relation between two variables is of primary interest, but there are one or more background variables to be controlled.

The main emphasis, however, is on graphical methods that help to describe the *pattern* of an association between variables. Section 4.4 presents the fourfold display, designed to portray the odds ratio in 2×2 tables or a set of k such tables. ***Sieve diagrams*** (Section 4.5) and ***association plot***s (Section 4.6) are more general methods for depicting the pattern of associations in any two-way table. When the row and column variables represent the classifications of different raters, specialized measures and visual displays for ***inter-rater agreement*** (Section 4.7) are particularly useful. Another specialized display, a ***trilinear plot*** or ***ternary plot***, described in Section 4.8, is designed for three-column frequency tables or compositional data. In order to make clear some of the distinctions that occur in contingency table analysis, we begin with several examples.

EXAMPLE 4.1: Berkeley admissions

Table 4.1 shows aggregate data on applicants to graduate school at Berkeley for the six largest departments in 1973 classified by admission and gender (Bickel et al., 1975). See *`UCBAdmissions`* (in package datasets) for the complete data set. For such data we might wish to study whether there is an association between admission and gender. Are male (or female) applicants more likely to be admitted? The presence of an association might be considered as evidence of sex bias in admission practices.

Table 4.1 is an example of the simplest kind of contingency table, a 2×2 classification of individuals according to two dichotomous (binary) variables. For such a table, the question of whether there is an association between admission and gender is equivalent to asking if the proportions of males and females who are admitted to graduate school are different, or whether the difference in proportions admitted is not zero. △

Table 4.1: Admissions to Berkeley graduate programs

Gender	Admitted	Rejected	Total	% Admit
Males	1198	1493	2691	44.52
Females	557	1278	1835	30.35
Total	1755	2771	4526	38.78

Although the methods for quantifying association in larger tables can be used for 2×2 tables, there are specialized measures (described in Section 4.2) and graphical methods for these simpler tables.

As we mentioned in Section 1.2.4 it is often useful to make a distinction between ***response***, or outcome variables, on the one hand, and possible ***explanatory*** or predictor variables on the other. In Table 4.1, it is natural to consider `admission` as the outcome, and `gender` as the explanatory variable. In other tables, no variable may be clearly identified as *the* outcome, or there may be several response variables, giving a multivariate problem.

EXAMPLE 4.2: Hair color and eye color

Table 4.2 shows data collected by Snee (1974) on the relation between hair color and eye color among 592 students in a statistics course (a two-way margin of *`HairEyeColor`*).

Neither hair color nor eye color is considered a response in relation to the other; our interest concerns whether an association exists between them. Hair color and eye color have both been classified into four categories. Although the categories used are among the most common, they are

Table 4.2: Hair-color eye-color data

Eye Color	Hair Color Black	Brown	Red	Blond	Total
Brown	68	119	26	7	220
Blue	20	84	17	94	215
Hazel	15	54	14	10	93
Green	5	29	14	16	64
Total	108	286	71	127	592

not the only categories possible.[1] A common, albeit deficient, representation of such a table is a ***grouped barchart***, as shown in the left of Figure 4.1:

```
> hec <- margin.table(HairEyeColor, 2:1)
> barplot(hec, beside = TRUE, legend = TRUE)
```

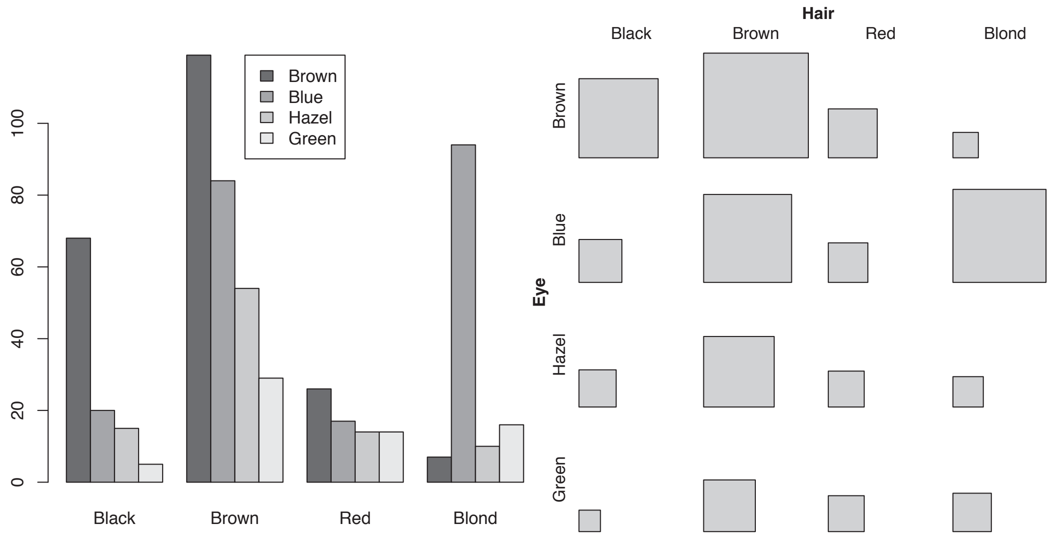


Figure 4.1: Two basic displays for the Hair-color Eye-color data. Left: grouped barchart; right: tile plot.

For each hair color, a group of bars represent the corresponding eye colors, the heights being proportional to the absolute frequencies. Bar graphs do not extend well to more than one dimension since

- the graphical representation does not match the tabular data structure, complicating comparisons with the raw data;
- it is harder to compare bars accross groups than within groups;
- by construction, the grouping suggests a conditional or causal relationship of the variables (here: "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);

[1]If students had been asked to write down their hair and eye colors, it is likely that many more than four categories of each would appear in a sample of nearly 600.

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

- 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 apparent that the "well" mental state decreases with social-economic status, while the "impaired" state increases. This pattern is more specific than overall association (as suspected for the hair-color eye-color data), and indeed, more powerful and focused tests are available when we treat these variables as *ordinal*, as we will see in Section 4.2.4. △

[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.

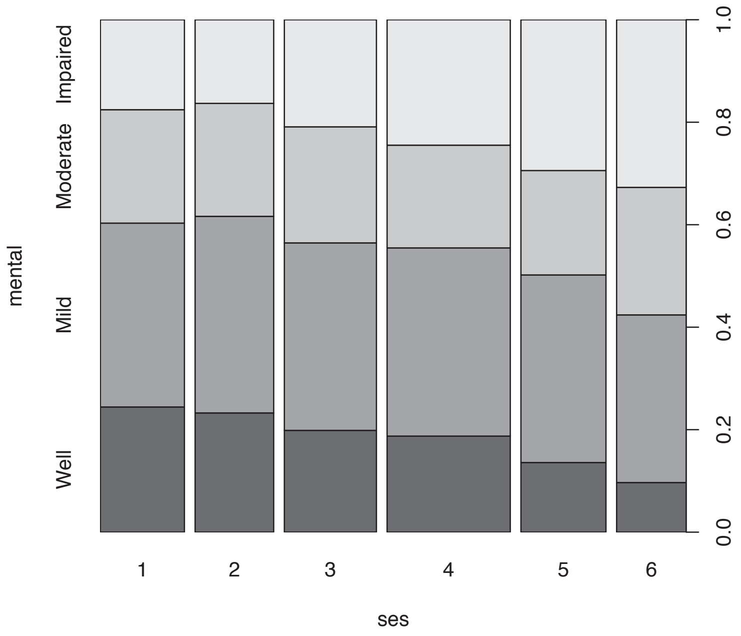


Figure 4.2: Spineplot of the Mental data.

EXAMPLE 4.4: Arthritis treatment

The data in Table 4.4 compares an active treatment for rheumatoid arthritis to a placebo (Koch and Edwards, 1988), used in examples in Chapter 2 (Example 2.2). The outcome reflects whether individuals showed no improvement, some improvement, or marked improvement. Here, the outcome variable is an ordinal one, and it is probably important to determine if the relation between treatment and outcome is the same for males and females. The data set is given in case form in *Arthritis* (in package vcd).

This is, of course, a three-way table, with factors `Treatment`, `Sex`, and `Improvement`. If the relation between treatment and outcome is the same for both genders, an analysis of the Treatment by Improvement table (collapsed over sex) could be carried out. Otherwise we could perform separate analyses for men and women, or treat the combinations of treatment and sex as four levels of a "population" variable, giving a 4×3 two-way table. These simplified approaches each ignore certain information available in an analysis of the full three-way table. $\triangle$

4.2 Tests of association for two-way tables

4.2.1 Notation and terminology

To establish notation, let $\boldsymbol{N} = \{n_{ij}\}$ be the observed frequency table of variables A and B with r rows and c columns, as shown in Table 4.5. In what follows, a subscript is replaced by a "+" when summed over the corresponding variable, so $n_{i+} = \sum_j n_{ij}$ gives the total frequency in row i, $n_{+j} = \sum_i n_{ij}$ gives the total frequency in column j, and $n_{++} = \sum_i \sum_j n_{ij}$ is the grand total; for convenience, n_{++} is also symbolized by n.

Table 4.4: Arthritis treatment data

		Improvement			
Treatment	Sex	None	Some	Marked	Total
Active	Female	6	5	16	27
	Male	7	2	5	14
Placebo	Female	19	7	6	32
	Male	10	0	1	11
Total		42	14	28	84

Table 4.5: The $R \times C$ contingency table

Row	Column category				
Category	1	2	$\cdots$	C	Total
1	n_{11}	n_{12}	$\cdots$	n_{1C}	n_{1+}
2	n_{21}	n_{22}	$\cdots$	n_{2C}	n_{2+}
$\vdots$	$\vdots$	$\vdots$	$\cdots$	$\vdots$	$\vdots$
R	n_{R1}	n_{R2}	$\cdots$	n_{RC}	n_{R+}
Total	n_{+1}	n_{+2}	$\cdots$	n_{+C}	n_{++}

When each observation is randomly sampled from some population and classified on two categorical variables, A and B, we refer to the ***joint distribution*** of these variables, and let $\pi_{ij} = \Pr(A = i, B = j)$ denote the population probability that an observation is classified in row i, column j (or cell (ij)) in the table. Corresponding to these population joint probabilities, the cell proportions, $p_{ij} = n_{ij}/n$, give the sample joint distribution.

The row totals n_{i+} and column totals n_{+j} are called ***marginal frequencies*** for variables A and B, respectively. These describe the distribution of each variable *ignoring* the other. For the population probabilities, the ***marginal distributions*** are defined analogously as the row and column totals of the joint probabilities, $\pi_{i+} = \sum_j \pi_{ij}$, and $\pi_{+j} = \sum_i \pi_{ij}$. The sample marginal proportions are, correspondingly, $p_{i+} = \sum_j p_{ij} = n_{i+}/n$, and $p_{+j} = \sum_i p_{ij} = n_{+j}/n$.

When one variable (the column variable, B, for example) is a response variable, and the other (A) is an explanatory variable, it is most often useful to examine the distribution of the response B for *each* level of A separately. These define the ***conditional distributions*** of B, given the level of A, and are defined for the population as $\pi_{j\,|\,i} = \pi_{ij}/\pi_{i+}$.

These definitions are illustrated for the Berkeley data (Table 4.1) below, using the function `CrossTable()`.

```
> Berkeley <- margin.table(UCBAdmissions, 2:1)
> library(gmodels)
> CrossTable(Berkeley, prop.chisq = FALSE, prop.c = FALSE,
+            format = "SPSS")


   Cell Contents
|-------------------------|
|                   Count |
|             Row Percent |
|           Total Percent |
|-------------------------|

Total Observations in Table:  4526

             | Admit
      Gender | Admitted   | Rejected   | Row Total |
-------------|------------|------------|-----------|
        Male |      1198  |      1493  |      2691 |
             |    44.519% |    55.481% |    59.456% |
             |    26.469% |    32.987% |           |
-------------|------------|------------|-----------|
      Female |       557  |      1278  |      1835 |
             |    30.354% |    69.646% |    40.544% |
             |    12.307% |    28.237% |           |
-------------|------------|------------|-----------|
Column Total |      1755  |      2771  |      4526 |
-------------|------------|------------|-----------|
```

The output shows the joint frequencies, n_{ij}, and joint sample percentages, $100 \times p_{ij}$, in the first row within each table cell. The second row in each cell ("Row percent") gives the conditional percentage of admission or rejection, $100 \times p_{j\,|\,i}$ for males and females separately. The row and column labelled "Total" give the marginal frequencies, n_{i+} and n_{+j}, and marginal percentages, p_{i+} and p_{+j}.

4.2.2 2 by 2 tables: Odds and odds ratios

The 2 by 2 contingency table of applicants to Berkeley graduate programs in Table 4.1 may be regarded as an example of a ***cross-sectional study***. The total of $n = 4,526$ applicants in 1973 has been classified by both gender and admission status. Here, we would probably consider the total n to be fixed, and the cell frequencies $n_{ij},\ i = 1,2; j = 1,2$ would then represent a single ***multinomial sample*** for the cross-classification by two binary variables, with probabilities cell $p_{ij},\ i = 1,2; j = 1,2$ such that

$$p_{11} + p_{12} + p_{21} + p_{22} = 1 .$$

The basic null hypothesis of interest for a multinomial sample is that of *independence*. Are admission and gender independent of each other?

Alternatively, if we consider admission the response variable, and gender an explanatory variable, we would treat the numbers of male and female applicants as fixed and consider the cell frequencies to represent two independent ***binomial samples*** for a binary response. In this case, the null hypothesis is described as that of *homogeneity* of the response proportions across the levels of the explanatory variable.

Measures of association are used to quantify the strength of association between variables. Among the many measures of association for contingency tables, the ***odds ratio*** is particularly useful for 2×2 tables, and is a fundamental parameter in several graphical displays and models described later. Other measures of strength of association for 2×2 tables are described in Stokes et al. (2000, Chapter 2) and Agresti (1996, Section 2.2).

For a binary response, where the probability of a "success" is π, the ***odds*** of a success is defined as

$$\text{odds} = \frac{\pi}{1-\pi} .$$

Hence, odds $= 1$ corresponds to $\pi = 0.5$, or success and failure equally likely. When success is more likely than failure $\pi > 0.5$, and the odds > 1; for instance, when $\pi = 0.75$, odds $= .75/.25 = 3$, so a success is three times as likely as a failure. When failure is more likely, $\pi < 0.5$, and the odds < 1; for instance, when $\pi = 0.25$, odds $= .25/.75 = \frac{1}{3}$.

The odds of success thus vary *multiplicatively* around 1. Taking logarithms gives an equivalent measure that varies *additively* around 0, called the ***log odds*** or ***logit***:

$$\text{logit}(\pi) \equiv \log(\text{odds}) = \log\left(\frac{\pi}{1-\pi}\right) . \tag{4.1}$$

The logit is symmetric about $\pi = 0.5$, in that $\text{logit}(\pi) = -\text{logit}(1-\pi)$. The following lines calculate the odds and log odds for a range of probabilities. As you will see in Chapter 7, the logit transformation of a probability is fundamental in logistic regression.

```
> p <- c(0.05, .1, .25, .50, .75, .9, .95)
> odds <- p / (1 - p)
> logodds <- log(odds)
> data.frame(p, odds, logodds)

     p      odds logodds
1 0.05  0.052632 -2.9444
2 0.10  0.111111 -2.1972
3 0.25  0.333333 -1.0986
4 0.50  1.000000  0.0000
5 0.75  3.000000  1.0986
6 0.90  9.000000  2.1972
7 0.95 19.000000  2.9444
```

A binary response for two groups gives a 2×2 table, with Group as the row variable, say. Let π_1 and π_2 be the success probabilities for Group 1 and Group 2. The ***odds ratio***, θ, is just the ratio of the odds for the two groups:

$$\text{odds ratio} \equiv \theta = \frac{\text{odds}_1}{\text{odds}_2} = \frac{\pi_1/(1-\pi_1)}{\pi_2/(1-\pi_2)} .$$

Like the odds itself, the odds ratio is always non-negative, between 0 and ∞. When $\theta = 1$, the distributions of success and failure are the same for both groups (so $\pi_1 = \pi_2$); there is no association between row and column variables, or the response is independent of group. When $\theta > 1$, Group 1 has a greater success probability; when $\theta < 1$, Group 2 has a greater success probability.

Similarly, the odds ratio may be transformed to a log scale, to give a measure that is symmetric about 0. The ***log odds ratio***, symbolized by ψ, is just the difference between the logits for Groups 1 and 2:

$$\text{log odds ratio} \equiv \psi = \log(\theta) = \log\left[\frac{\pi_1/(1-\pi_1)}{\pi_2/(1-\pi_2)}\right] = \text{logit}(\pi_1) - \text{logit}(\pi_2) .$$

Independence corresponds to $\psi = 0$, and reversing the rows or columns of the table merely changes the sign of ψ.

For sample data, the ***sample odds ratio*** is the ratio of the sample odds for the two groups:

$$\hat{\theta} = \frac{p_1/(1-p_1)}{p_2/(1-p_2)} = \frac{n_{11}/n_{12}}{n_{21}/n_{22}} = \frac{n_{11}n_{22}}{n_{12}n_{21}} . \tag{4.2}$$

The sample estimate $\hat{\theta}$ in Eqn. (4.2) is the maximum likelihood estimator of the true θ. The sampling distribution of $\hat{\theta}$ is asymptotically normal as $n \to \infty$, but may be highly skewed in small to moderate samples.

Consequently, inference for the odds ratio is more conveniently carried out in terms of the log odds ratio, whose sampling distribution is more closely normal, with mean $\psi = \log(\theta)$, and asymptotic standard error (ASE)

$$\text{ASE}_{\log(\theta)} \equiv \hat{s}(\hat{\psi}) = \sqrt{\frac{1}{n_{11}} + \frac{1}{n_{12}} + \frac{1}{n_{21}} + \frac{1}{n_{22}}} = \sqrt{\sum_{i,j} n_{ij}^{-1}} . \tag{4.3}$$

A large-sample $100(1-\alpha)\%$ confidence interval for $\log(\theta)$ may therefore be calculated as

$$\log(\theta) \pm z_{1-\alpha/2} \, \text{ASE}_{\log(\theta)} = \hat{\psi} \pm z_{1-\alpha/2} \, \hat{s}(\hat{\psi})$$

where $z_{1-\alpha/2}$ is the cumulative normal quantile with $1-\alpha/2$ in the lower tail. Confidence intervals for θ itself are obtained by exponentiating the end points of the interval for $\psi = \log(\theta)$,[4]

$$\exp\left(\hat{\psi} \pm z_{1-\alpha/2}\hat{s}(\hat{\psi})\right) .$$

EXAMPLE 4.5: Berkeley admissions

As an illustratation, we apply these formulae to the UCB Admissions data, using the `loddsratio()` function in vcd, which by default calculates log-odds:

```
> data("UCBAdmissions")
> UCB <- margin.table(UCBAdmissions, 1:2)
> (LOR <- loddsratio(UCB))

log odds ratios for Admit and Gender

[1] 0.61035

> (OR <- loddsratio(UCB, log = FALSE))

 odds ratios for Admit and Gender

[1] 1.8411
```

The function returns an object for which the `summary()` method computes the ASE and carries out the significance test (for the log odds):

```
> summary(LOR)


z test of coefficients:

                              Estimate Std. Error z value
Admitted:Rejected/Male:Female   0.6104     0.0639    9.55
                              Pr(>|z|)
Admitted:Rejected/Male:Female   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Clearly, the hypothesis of independence has to be rejected, suggesting the presence of gender bias. `confint()` computes confidence intervals for (log) odds ratios:

```
> 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
```

[4]Note that $\hat{\theta}$ is 0 or ∞ if any $n_{ij} = 0$. Haldane (1955) and Gart and Zweiful (1967) showed that improved estimators of θ and $\psi = \log(\theta)$ are obtained by replacing each n_{ij} by $[n_{ij} + \frac{1}{2}]$ in Eqn. (4.2) and Eqn. (4.3). This adjustment is preferred in small samples, and required if any zero cells occur. In large samples, the effect of adding 0.5 to each cell becomes negligible.

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
```

The row proportions show a clear difference in the outcome for the two groups: For those given the placebo, 67% reported no improvement; in the treated group, 51% reported marked improvement. χ^2 tests and measures of association are provided by `assocstats()` as shown below:

```
> assocstats(Art)

                    X^2 df  P(> X^2)
Likelihood Ratio 13.530  2 0.0011536
Pearson          13.055  2 0.0014626
```

```
Phi-Coefficient   : NA
Contingency Coeff.: 0.367
Cramer's V        : 0.394
```

△

The measures of association are normalized variants of the χ^2 statistic. Caution is needed for interpretation since the maximum values depend on the table dimensions.

4.2.4 Tests for ordinal variables

For $r \times c$ tables, more sensitive tests than the test for general association (independence) are available if either or both of the row and column variables are ordinal. Generalized ***Cochran–Mantel–Haenszel tests*** (Landis et al., 1978), which take the ordinal nature of a variable into account, are provided by the `CMHtest()` in vcdExtra. These tests are based on assigning numerical scores to the table categories; the default (table) scores treat the levels as equally spaced. They generally have higher power when the pattern of association is determined by the order of an ordinal variable.

EXAMPLE 4.7: Mental impairment and parents' SES
We illustrate these tests using the data on mental impairment and SES introduced in Example 4.3, where both variables can be considered ordinal.

```
> data("Mental", package = "vcdExtra")
> mental <- xtabs(Freq ~ ses + mental, data = Mental)
> assocstats(mental)    # standard chisq tests

                    X^2 df   P(> X^2)
Likelihood Ratio 47.418 15 3.1554e-05
Pearson          45.985 15 5.3458e-05

Phi-Coefficient   : NA
Contingency Coeff.: 0.164
Cramer's V        : 0.096

> CMHtest(mental)        # CMH tests

Cochran-Mantel-Haenszel Statistics for ses by mental

                 AltHypothesis Chisq Df     Prob
cor        Nonzero correlation  37.2  1 1.09e-09
rmeans  Row mean scores differ  40.3  5 1.30e-07
cmeans  Col mean scores differ  40.7  3 7.70e-09
general    General association  46.0 15 5.40e-05
```

In this data set, all four tests show a highly significant association. However, the `cor` test for nonzero correlation uses only one degree of freedom, whereas the test of general association requires 15 df. △

The four tests differ in the types of departure from independence they are sensitive to:

General Association When the row and column variables are both nominal (unordered), the only alternative hypothesis of interest is that there is *some* association between the row and column variables. The CMH test statistic is similar to the (Pearson) Chi-Square and Likelihood Ratio Chi-Square in the result from `assocstats()`; all have $(r-1)(c-1)$ df.

Row Mean Scores Differ If the column variable is ordinal, assigning scores to the column variable produces a mean for each row. The association between row and column variables can be expressed as a test of whether these means differ over the rows of the table, with $r-1$ df. This is analogous to the Kruskal-Wallis non-parametric test (ANOVA based on rank scores).

Column Mean Scores Differ Same as the above, assigning scores to the row variable.

Nonzero Correlation (Linear association) When *both* row and column variables are ordinal, we could assign scores to both variables and compute the correlation (r), giving Spearman's rank correlation coefficient. The CMH χ^2 is equal to $(N-1)r^2$, where N is the total sample size. The test is most sensitive to a pattern where the row mean score changes linearly over the rows.

4.2.5 Sample CMH profiles

Two contrived examples may make the differences among these tests more apparent. Visualizations of the patterns of association reinforces the aspects to which the tests are most sensitive, and introduces the sieve diagram described more fully in Section 4.5.

4.2.5.1 General association

The table below exhibits a general association between variables A and B, but no difference in row means or linear association. The row means for category j are calculated by assigning integer scores, $b_i = i$, to the column categories, and using the corresponding frequencies of row j as weights. The column means are obtained analogously. Figure 4.3 (left) shows the pattern of association in this table graphically, as a sieve diagram (described in Section 4.5).

	b1	b2	b3	b4	b5	Total	Mean
a1	0	15	25	15	0	55	3.0
a2	5	20	5	20	5	55	3.0
a3	20	5	5	5	20	55	3.0
Total	25	40	35	40	25	165	3.0
Mean	2.8	1.6	1.4	1.6	2.8	2.1	

This is reflected in the `CMHtest()` output shown below (`cmhdemo1` contains the data shown above).

```
> CMHtest(cmhdemo1)

Cochran-Mantel-Haenszel Statistics

                AltHypothesis Chisq Df      Prob
cor        Nonzero correlation   0.0  1 1.00e+00
rmeans  Row mean scores differ   0.0  2 1.00e+00
cmeans  Col mean scores differ  72.2  4 7.78e-15
general    General association  91.8  8 2.01e-16
```

The chi-square values for non-zero correlation and different row mean scores are exactly zero because the row means are all equal. Only the general association test shows that A and B are associated.

4.2.5.2 Linear association

The table below contains a weak, non-significant general association, but significant row mean differences and linear associations. The unstructured test of general association would therefore lead to the conclusion that no association exists, while the tests taking ordinal factors into account would conclude otherwise. Note that the largest frequencies shift towards lower levels of B as the level of variable A increases. See Figure 4.3 (right) for a visual representation of this pattern.

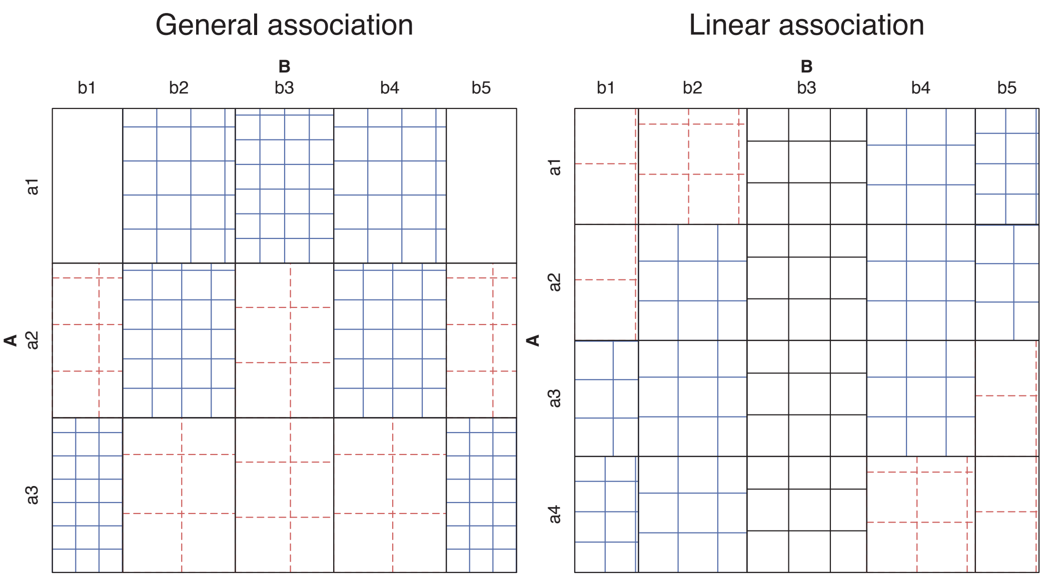


Figure 4.3: Sieve diagrams for two patterns of association: Left: general association; right: linear association.

	b1	b2	b3	b4	b5	Total	Mean
a1	2	5	8	8	8	31	3.48
a2	2	8	8	8	5	31	3.19
a3	5	8	8	8	2	31	2.81
a4	8	8	8	5	2	31	2.52
Total	17	29	32	29	17	124	3.00
Mean	3.1	2.7	2.5	2.3	1.9	2.5	

Note that the χ^2-values for the row-means and non-zero correlation tests from `CMHtest()` are very similar, but the correlation test is more highly significant since it is based on just one degree of freedom. In the following example, `cmhdemo2` corresponds to the table above:

```
> CMHtest(cmhdemo2)

Cochran-Mantel-Haenszel Statistics

                 AltHypothesis Chisq Df    Prob
cor        Nonzero correlation  10.6  1 0.00111
rmeans   Row mean scores differ  10.7  3 0.01361
cmeans   Col mean scores differ  11.4  4 0.02241
general    General association  13.4 12 0.34064
```

The difference in sensitivity and power among these tests for categorical data is analogous to the difference between general ANOVA tests and tests for linear trend (contrasts) in experimental designs with quantitative factors: The more specific test has greater power, but is sensitive to a narrower range of departures from the null hypothesis. The more focused tests for ordinal factors are a better bet when we believe that the association depends on the ordered nature of the factor levels.

4.3 Stratified analysis

An overall analysis ignores other variables (like sex), by collapsing over them. In the *Arthritis* data, it is possible that the treatment is effective only for one gender, or even that the treatment has opposite effects for men and women. If so, pooling over the ignored variable(s) can be seriously misleading.

4.3.1 Computing strata-wise statistics

A ***stratified analysis*** controls for the effects of one or more background variables. This is similar to the use of a blocking variable in an ANOVA design. Tests for association can be obtained by applying a function (`assocstats()`, `CMHtest()`) over the levels of the stratifying variables.

EXAMPLE 4.8: Arthritis treatment

The statements below request a stratified analysis of the arthritis treatment data with CMH tests, controlling for gender. Essentially, the analysis is carried out separately for males and females.

The table `Art2` is constructed as a three-way table, with `Sex` as the last dimension.

```
> Art2 <- xtabs(~ Treatment + Improved + Sex, data = Arthritis)
> Art2

, , Sex = Female

          Improved
Treatment None Some Marked
  Placebo   19    7      6
  Treated    6    5     16

, , Sex = Male

          Improved
Treatment None Some Marked
  Placebo   10    0      1
  Treated    7    2      5
```

Both `assocstats()` and `CMHtest()` are designed for stratified tables, and use all dimensions after the first two as strata.

```
> assocstats(Art2)

$`Sex:Female`
                    X^2 df  P(> X^2)
Likelihood Ratio 11.731  2 0.0028362
Pearson          11.296  2 0.0035242

Phi-Coefficient   : NA
Contingency Coeff.: 0.401
Cramer's V        : 0.438

$`Sex:Male`
                    X^2 df P(> X^2)
Likelihood Ratio 5.8549  2 0.053532
Pearson          4.9067  2 0.086003

Phi-Coefficient   : NA
Contingency Coeff.: 0.405
Cramer's V        : 0.443
```

Note that even though the strength of association (Cramer's V) is similar in the two groups, the χ^2 tests show significance for females, but not for males. This is true even using the more powerful CMH tests below, treating `Treatment` as ordinal. The reason is that there were more than twice as many females as males in this sample.

```
> CMHtest(Art2)

$`Sex:Female`
Cochran-Mantel-Haenszel Statistics for Treatment by Improved
in stratum Sex:Female

                 AltHypothesis Chisq Df     Prob
cor         Nonzero correlation  10.9  1 0.000944
rmeans   Row mean scores differ  10.9  1 0.000944
cmeans   Col mean scores differ  11.1  2 0.003878
general     General association  11.1  2 0.003878


$`Sex:Male`
Cochran-Mantel-Haenszel Statistics for Treatment by Improved
in stratum Sex:Male

                 AltHypothesis Chisq Df   Prob
cor         Nonzero correlation  3.71  1 0.0540
rmeans   Row mean scores differ  3.71  1 0.0540
cmeans   Col mean scores differ  4.71  2 0.0949
general     General association  4.71  2 0.0949

> apply(Art2, 3, sum)

Female   Male
    59     25
```

△

4.3.2 Assessing homogeneity of association

In a stratified analysis it is often crucial to know if the association between the primary table variables is the same over all strata. For $2 \times 2 \times k$ tables this question reduces to whether the odds ratio is the same in all k strata. The vcd package implements Woolf's test (Woolf, 1995) in `woolf_test()` for this purpose.

For larger n-way tables, this question is equivalent to testing whether the association between the primary variables, A and B, say, is the same for all levels of the stratifying variables, C, D, ...

EXAMPLE 4.9: Berkeley admissions

Here we illustrate the use of Woolf's test for the *UCBAdmissions* data. The test is significant, indicating that the odds ratios cannot be considered equal across departments. We will see why when we visualize the data by department in the next section.

```
> woolf_test(UCBAdmissions)


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

data:  UCBAdmissions
X-squared = 17.9, df = 5, p-value = 0.0031
```

△

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,

$$\texttt{[SexTreatment] [SexImproved] [TreatmentImproved]}. \tag{4.4}$$

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.

The main purpose of this display is to depict the sample odds ratio, $\hat{\theta} = (n_{11}/n_{12}) \div (n_{21}/n_{22})$. An association between the variables ($\theta \neq 1$) is shown by the tendency of diagonally opposite cells in one direction to differ in size from those in the opposite direction, and the display uses color or shading to show this direction. Confidence rings for the observed θ allow a visual test of the hypothesis of independence, $H_0 : \theta = 1$. They have the property that (in a standardized display) the rings for adjacent quadrants overlap *iff* the observed counts are consistent with the null hypothesis.

Example 4.11: Berkeley admissions

Figure 4.4 (left) shows the basic, unstandardized fourfold display for the Berkeley admissions data (Table 4.1). Here, the area of each quadrant is proportional to the cell frequency, shown numerically in each corner. The odds ratio is proportional to the product of the areas shaded dark, divided by the product of the areas shaded light. The sample odds ratio, Odds(Admit|Male) / Odds(Admit|Female) is 1.84 (see Example 4.9) indicating that males were nearly twice as likely to be admitted.

```
> fourfold(Berkeley, std = "ind.max")    # unstandardized
> fourfold(Berkeley, margin = 1)         # equating gender
```

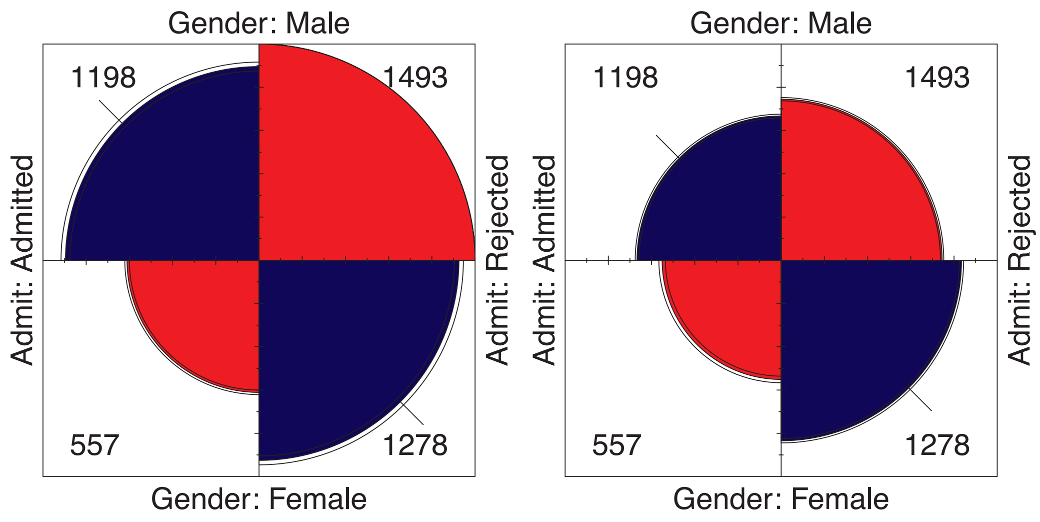


Figure 4.4: Fourfold displays for the Berkeley admission data. Left: unstandardized; right: equating the proportions of males and females.

However, it is difficult to make these visual comparisons because there are more men than women, and because the proportions admitted and rejected are unequal. In the unstandardized display the confidence bands have no interpretation as a test of $H_0 : \theta = 1$.

The data in a 2×2 table can be standardized to make these visual comparisons easier. Table 4.6 shows the Berkeley data with the addition of row percentages (which equate for the number of men and women applicants) indicating the proportion of each gender accepted and rejected. We see that 44.52% of males were admitted, while only 30.35% of females were admitted. Moreover, the row percentages have the same odds ratio as the raw data: $44.52 \times 69.65/30.35 \times 55.48 = 1.84$. Figure 4.4 (right) shows the fourfold display where the area of each quarter circle is proportional to these row percentages.

Table 4.6: Admissions to Berkeley graduate programs, frequencies and row percentages.

	Frequencies		Row Percents	
	Admitted	Rejected	Admitted	Rejected
Males	1198	1493	44.52	55.48
Females	557	1278	30.35	69.65

With this standardization, the confidence rings have the property that the confidence rings for each upper quadrant will overlap with those for the quadrant below it if the odds ratio does not differ from 1.0. (Details of the calculation of confidence rings are described in the next section.) No similar statement can be made about the corresponding left and right quadrants, however, because the overall rate of admission has not been standardized.

As a final step, we can standardize the data so that *both* table margins are equal, while preserving the odds ratio. Each quarter circle is then drawn to have an area proportional to this standardized cell frequency. This makes it easier to see the association between admission and sex without being influenced by the overall admission rate or the differential tendency of males and females to apply. With this standardization, the four quadrants will align (overlap) horizontally and vertically when the odds ratio is 1, regardless of the marginal frequencies. The fully standardized display, which is usually the most useful form, is shown in Figure 4.5.

```
> fourfold(Berkeley)   # standardize both margins
```

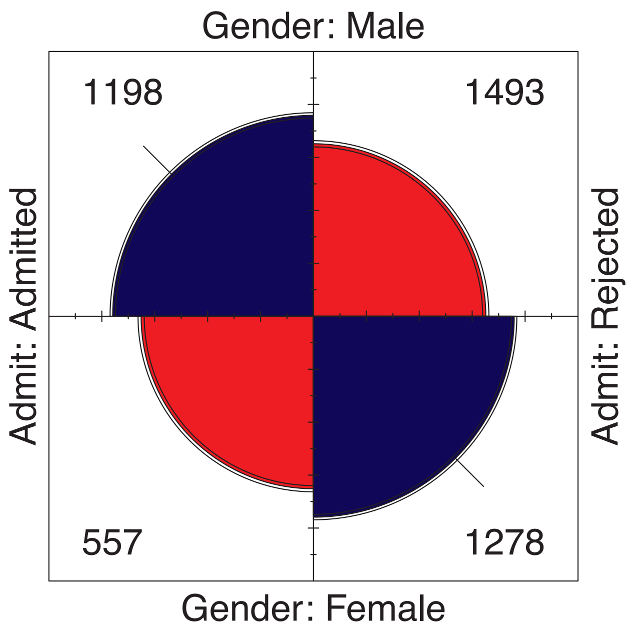


Figure 4.5: Fourfold display for Berkeley admission data with margins for gender and admission equated. The area of each quadrant shows the standardized frequency in each cell.

△

These displays also use color (blue) and diagonal tick marks to show the direction of positive association. The visual interpretation (also conveyed by area) is that males are more likely to be accepted, females more likely to be rejected.

The quadrants in Figure 4.5 do not align and the 95% confidence rings around each quadrant do not overlap, indicating that the odds ratio differs significantly from 1—putative evidence of gender bias. The very narrow width of the confidence rings gives a visual indication of the precision of the data—if we stopped here, we might feel quite confident of this conclusion.

4.4.1 Confidence rings for odds ratio

Confidence rings for the fourfold display are computed from a confidence interval for θ, whose endpoints can each be mapped into a 2×2 table. Each such table is then drawn in the same way as the data.

The interval for θ is most easily found by considering the distribution of $\hat{\psi} = \log \hat{\theta}$, whose standard error may be estimated by Eqn. (4.3). Then an approximate $1 - \alpha$ confidence interval for ψ is given by

$$\hat{\psi} \pm \hat{s}(\hat{\psi})\, z_{1-\alpha/2} = \{\hat{\psi}_l, \hat{\psi}_u\} ,$$

as described in Section 4.2.2. The corresponding limits for the odds ratio θ are $\{\exp(\hat{\psi}_l), \exp(\hat{\psi}_u)\}$. For the data shown in Figure 4.5, $\hat{\psi} = \log \hat{\theta} = .6104$, and $\hat{s}(\hat{\psi}) = 0.0639$, so the 95%, limits for θ are $\{1.624, 2.087\}$, as shown by the calculations below. The same result is returned by `confint()` for a "loddsratio" object.

```
> summary(loddsratio(Berkeley))


z test of coefficients:

                              Estimate Std. Error z value
Male:Female/Admitted:Rejected   0.6104     0.0639    9.55
                              Pr(>|z|)
Male:Female/Admitted:Rejected   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

> exp(.6103 + c(-1, 1) * qnorm(.975) * 0.06398)

[1] 1.6240 2.0869

> confint(loddsratio(Berkeley, log = FALSE))

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

Now consider how to find a 2×2 table whose frequencies correspond to the odds ratios at the limits of the confidence interval. A table standardized to equal row and column margins can be represented by the 2×2 matrix with entries

$$\begin{bmatrix} p & (1-p) \\ (1-p) & p \end{bmatrix} ,$$

whose odds ratio is $\theta = p^2/(1-p)^2$. Solving for p gives $p = \sqrt{\theta}/(1 + \sqrt{\theta})$. The corresponding frequencies can then be found by adjusting the standardized table to have the same row and column margins as the data. The results of these computations, which generate the confidence rings in Figure 4.5, are shown in Table 4.7.

4.4.2 Stratified analysis for $2 \times 2 \times k$ tables

In a $2 \times 2 \times k$ table, the last dimension often corresponds to "strata" or populations, and it is typically of interest to see if the association between the first two variables is homogeneous across strata. For such tables, simply make one fourfold panel for each stratum. The standardization of marginal frequencies is designed to allow easy visual comparison of the pattern of association when the marginal frequencies vary across two or more populations.

Table 4.7: Odds ratios and equivalent tables for 95% confidence rings for the Berkeley data.

	Odds Ratio	Standardized Table		Equivalent Frequencies	
Lower limit	1.624	0.560	0.440	1,167.1	587.9
		0.440	0.560	1,523.9	1,247.1
Data	1.841	0.576	0.424	1,198.0	557.0
		0.424	0.576	1,493.0	1,278.0
Upper limit	2.087	0.591	0.409	1,228.4	526.6
		0.409	0.591	1,462.6	1,308.4

4.4.2.1 Stratified displays

The admissions data shown in Figure 4.4 and Figure 4.5 were actually obtained from six departments—the six largest at Berkeley (Bickel et al., 1975). To determine the source of the apparent sex bias in favor of males, we make a new plot, Figure 4.6, stratified by department.

```
> # fourfold display
> UCB <- aperm(UCBAdmissions, c(2, 1, 3))
> fourfold(UCB, mfrow = c(2, 3))
```

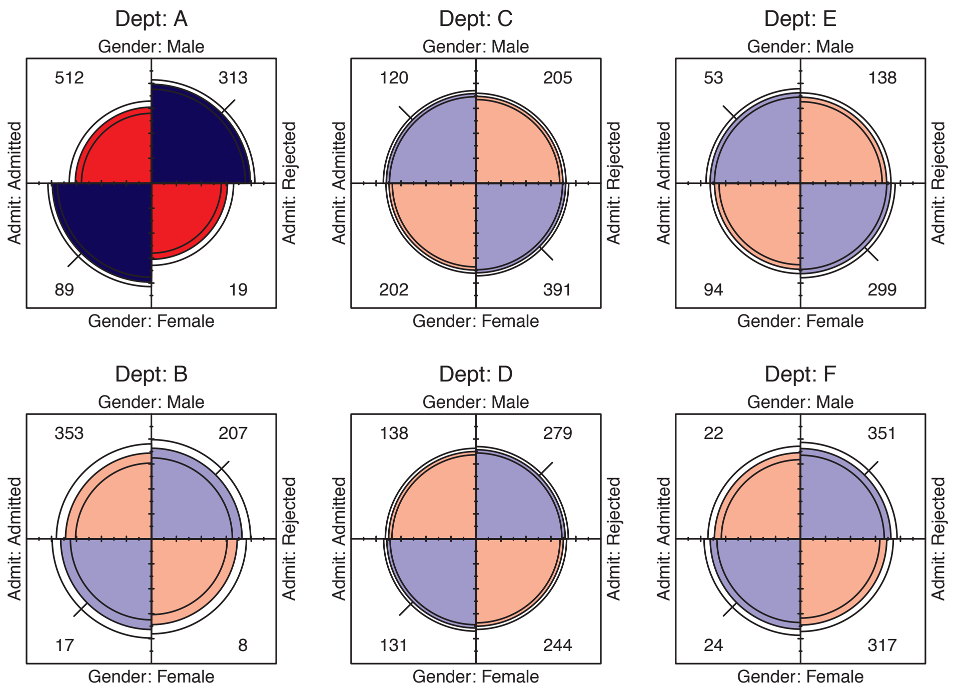


Figure 4.6: Fourfold displays for Berkeley admissions data, stratified by department. The more intense shading for Dept. A indicates a significant association.

Surprisingly, Figure 4.6 shows that, for five of the six departments, the odds of admission is approximately the same for both men and women applicants. Department A appears to differs from the others, with women approximately 2.86 ($= (313/19)/(512/89)$) times as likely to gain admission. This appearance is confirmed by the confidence rings, which in Figure 4.6 are joint[5] 95% intervals for θ_c, $c = 1, \ldots, k$.

This result, which contradicts the display for the aggregate data in Figure 4.4, is a nice example of ***Simpson's paradox***,[6] and illustrates clearly why an overall analysis of a three- (or higher-) way table can be misleading.

The resolution of this contradiction can be found in the large differences in admission rates among departments. Men and women apply to different departments differentially, and in these data women happen to apply in larger numbers to departments that have a low acceptance rate. The aggregate results are misleading because they falsely assume men and women are equally likely to apply in each field.[7]

4.4.2.2 Visualization principles for complex data

An important principle in the display of large, complex data sets is ***controlled comparison***—we want to make comparisons against a clear standard, with other things held constant. The fourfold display differs from a pie chart in that it holds the angles of the segments constant and varies the radius. An important consequence is that we can quite easily compare a series of fourfold displays for different strata, since corresponding cells of the table are always in the same position. As a result, an array of fourfold displays serve the goals of comparison and detection better than an array of pie charts.

Moreover, it allows the observed frequencies to be standardized by equating either the row or column totals, while preserving the design goal for this display—the odds ratio. In Figure 4.6, for example, the proportion of men and women, and the proportion of accepted applicants were equated visually in each department. This provides a clear standard that also greatly facilitates controlled comparison.

As mentioned in the introduction, another principle is ***visual impact***—we want the important features of the display to be easily distinguished from the less important (Tukey, 1993). Figure 4.6 distinguishes the one department for which the odds ratio differs significantly from 1 by shading intensity, even though the same information can be found by inspection of the confidence rings.

EXAMPLE 4.12: Breathlessness and wheeze in coal miners

The various ways of standardizing a collection of 2×2 tables allows visualizing relations with different factors (row percentages, column percentages, strata totals) controlled. However, different kinds of graphs can speak more eloquently to other questions by focusing more directly on the odds ratio.

Agresti (2002, Table 9.8) cites data from Ashford and Sowden (1970) on the association between two pulmonary conditions, breathlessness and wheeze, in a large sample of coal miners. The miners are classified into age groups, and the question treated by Agresti is whether the association between these two symptoms is homogeneous over age. These data are available in the `CoalMiners` data in vcd, a $2 \times 2 \times 9$ frequency table. The first group, aged 20–24 has been omitted from these analyses.

[5]For multiple-strata plots, `fourfold()` by default adjusts the significance level for multiple testing, using Holm's (1979) method provided by `p.adjust()`.

[6]Simpson's paradox (Simpson, 1951) occurs in a three-way table, $[A, B, C]$, when the marginal association between two variables, A, B collapsing over C, differs in *direction* from the partial association $A, B | C = c_k$ at the separate levels of C. Strictly speaking, Simpson's paradox would require that for all departments separately the odds ratio $\theta_k < 1$ (which occurs for Departments A, B, D, and F in Figure 4.6) while in the aggregate data $\theta > 1$.

[7]This explanation ignores the possibility of structural bias against women, e.g., lack of resources allocated to departments that attract women applicants.

```
> data("CoalMiners", package = "vcd")
> CM <- CoalMiners[, , 2 : 9]
> structable(. ~ Age, data = CM)

      Breathlessness    B        NoB
      Wheeze            W  NoW     W  NoW
Age
25-29                  23    9   105 1654
30-34                  54   19   177 1863
35-39                 121   48   257 2357
40-44                 169   54   273 1778
45-49                 269   88   324 1712
50-54                 404  117   245 1324
55-59                 406  152   225  967
60-64                 372  106   132  526
```

The question of interest can be addressed by displaying the odds ratio in the 2 × 2 tables with the margins of breathlessness and wheeze equated (i.e., with the default `std='margins'` option), which gives the graph shown in Figure 4.7. Although the panels for all age groups show an overwhelmingly positive association between these two symptoms, one can also (by looking carefully) see that the strength of this association declines with increasing age.

```
> fourfold(CM, mfcol = c(2, 4))
```

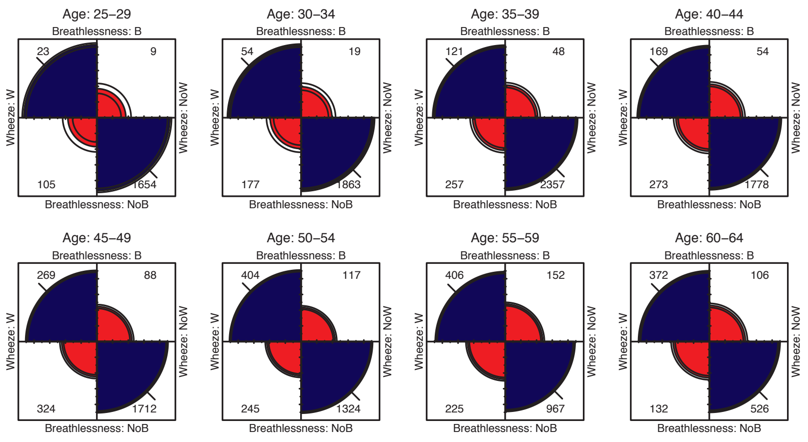


Figure 4.7: Fourfold display for CoalMiners data, both margins equated.

However, note that the pattern of change over age is somewhat subtle compared to the dominant positive association within each panel. When the goal is to display how the odds ratio varies with a quantitative factor such as age, it is often better to simply calculate and plot the odds ratio directly.

The `loddsratio()` function in vcd calculates odds ratios. By default, it returns the log odds. Use the option `log=FALSE` to get the odds ratios themselves. It is easy to see that the (log) odds ratios decline with age.

```
> loddsratio(CM)

log odds ratios for Breathlessness and Wheeze by Age

 25-29  30-34  35-39  40-44  45-49  50-54  55-59  60-64
3.6953 3.3983 3.1407 3.0147 2.7820 2.9264 2.4406 2.6380

> loddsratio(CM, log = FALSE)
```

```
 odds ratios for Breathlessness and Wheeze by Age

 25-29  30-34  35-39  40-44  45-49  50-54  55-59  60-64
40.256 29.914 23.119 20.383 16.152 18.660 11.480 13.985
```

When the analysis goal is to understand how the odds ratio varies with a stratifying factor (which could be a quantitative variable), it is often better to plot the odds ratio directly.

The lines below use the `plot()` method for "oddsratio" objects. This produces a line graph of the log odds ratio against the stratum variable, together with confidence interval error bars. In addition, because age is a quantitative variable, we can calculate and display the fitted relation for a linear model relating `lodds` to `age`. Here, we try using a quadratic model (`poly(age, 2)`) mainly to see if the trend is nonlinear.

```
> lor_CM <- loddsratio(CM)
> plot(lor_CM, bars=FALSE, baseline=FALSE, whiskers=.2)
>
> lor_CM_df <- as.data.frame(lor_CM)
> age <- seq(25, 60, by = 5) + 2
> lmod <- lm(LOR ~ poly(age, 2), weights = 1 / ASE^2, data = lor_CM_df)
> grid.lines(seq_along(age), fitted(lmod),
+            gp = gpar(col = "red", lwd = 2), default.units = "native")
```

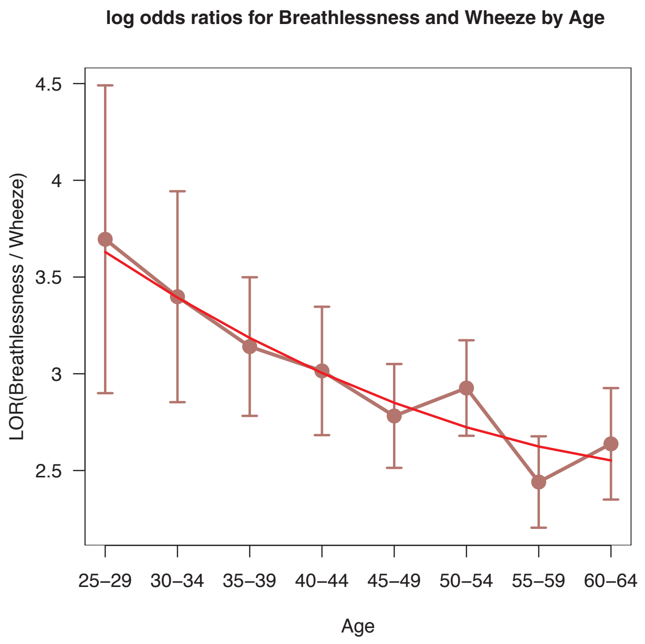


Figure 4.8: Log odds plot for the CoalMiners data. The smooth curve shows a quadratic fit to age.

In Figure 4.8, it appears that the decline in the log odds ratio levels off with increasing age. One virtue of fitting the model in this way is that we can test the additional contribution of the quadratic term, which turns out to be insignificant.

```
> summary(lmod)


Call:
lm(formula = LOR ~ poly(age, 2), data = lor_CM_df, weights = 1/ASE^2)
```

```
Weighted Residuals:
      1       2       3       4       5       6       7       8
 0.1617  0.0162 -0.2443  0.0627 -0.4971  1.6115 -1.5228  0.5851

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)     2.9953     0.0783   38.28  2.3e-07 ***
poly(age, 2)1  -0.9977     0.2513   -3.97    0.011 *
poly(age, 2)2   0.1768     0.2171    0.81    0.452
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.06 on 5 degrees of freedom
Multiple R-squared:  0.782,Adjusted R-squared:  0.694
F-statistic: 8.94 on 2 and 5 DF,  p-value: 0.0223
```

△

4.5 Sieve diagrams

The wise ones fashioned speech with their thought, sifting it as grain is sifted through a sieve.

Buddha

For two- (and higher-) way contingency tables, the design principles of perception, detection, and comparison (see Chapter 1) suggest that we should try to show the observed frequencies in relation to what we would expect those frequencies to be under a reasonable null model—for example, the hypothesis that the row and column variables are unassociated.

To this end, several schemes for representing contingency tables graphically are based on the fact that when the row and column variables are independent, the estimated expected frequencies, m_{ij}, are products of the row and column totals (divided by the grand total).

$$m_{ij} = \frac{n_{i+} n_{+j}}{n_{++}} \; .$$

Then, each cell can be represented by a rectangle whose area shows the observed cell frequency, n_{ij}, expected frequency, m_{ij}, or deviation (residual) from independence, $n_{ij} - m_{ij}$. Visual attributes (color, shading) of the rectangles can be used to highlight the pattern of association.

4.5.1 Two-way tables

For example, for any two-way table, the expected frequencies under independence can be represented by rectangles whose widths are proportional to the total frequency in each column, n_{+j}, and whose heights are proportional to the total frequency in each row, n_{i+}; the area of each rectangle is then proportional to m_{ij}. Figure 4.9 (left) shows the expected frequencies for the hair and eye color data (Table 4.2), calculated using `independence_table()` in vcd.

```
> haireye <- margin.table(HairEyeColor, 1:2)
> expected = independence_table(haireye)
> round(expected, 1)

       Eye
Hair    Brown  Blue Hazel Green
  Black  40.1  39.2  17.0  11.7
  Brown 106.3 103.9  44.9  30.9
  Red    26.4  25.8  11.2   7.7
  Blond  47.2  46.1  20.0  13.7
```

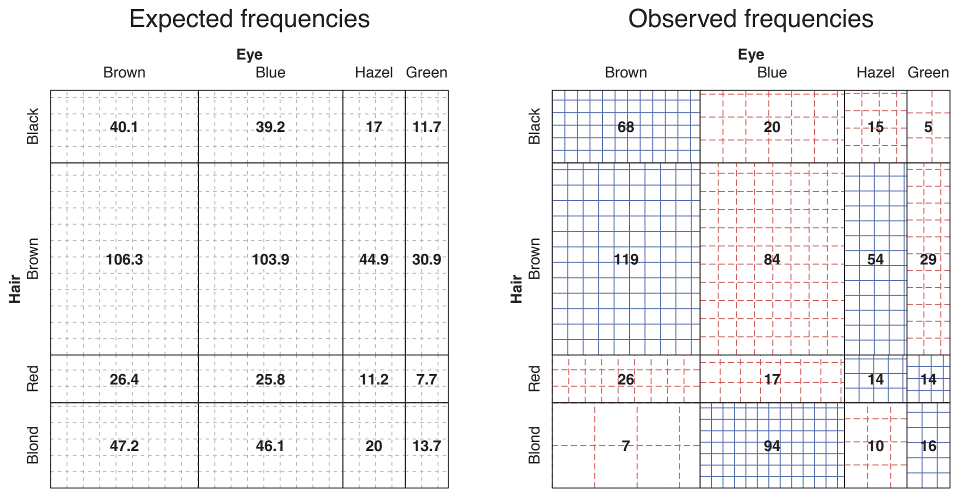


Figure 4.9: Sieve diagrams for the *HairEyeColor* data. Left: expected frequencies shown in cells as numbers and the number of boxes; right: observed frequencies shown in cells.

Figure 4.9 (left) simply represents the model—what the frequencies would be if hair color and eye color were independent—not the data. Note, however, that the rectangles are cross-ruled so that the number of boxes in each (counting up the fractional bits) equals the expected frequency with which the cell is labeled, and moreover, the rulings are equally spaced in all cells. Hence, cross-ruling the cells to show the observed frequency would give a data display that implicitly compares observed and expected frequencies as shown in Figure 4.9 (right).

Riedwyl and Schüpbach (1983, 1994) proposed a ***sieve diagram*** (later called a ***parquet diagram***) based on this principle. In this display the area of each rectangle is always proportional to expected frequency but observed frequency is shown by the number of squares in each rectangle, as in Figure 4.9 (right).

Hence, the difference between observed and expected frequency appears as variations in the density of shading. Cells whose observed frequency n_{ij} exceeds the expected m_{ij} appear denser than average. The pattern of positive and negative deviations from independence can be more easily seen by using color, say, red for negative deviations, and blue for positive.[8]

Example 4.13: Hair color and eye color

The sieve diagram for hair color and eye color shown in Figure 4.9 (right) can be interpreted as follows: The pattern of color and shading shows the high frequency of blue-eyed blonds and people with brown eyes and dark hair. People with hazel eyes are also more likely to have red or brown hair, and those with green eyes more likely to have red or blond hair, than would be observed under independence. △

Example 4.14: Visual acuity

In World War II, all workers in the UK Royal Ordnance factories were given test of visual acuity (unaided distance vision) of their left and right eyes on a 1 (high) to 4 (low) scale. The dataset *VisualAcuity* in vcd gives the results for 10,719 workers (3,242 men, 7,477 women) aged 30–39.

Figure 4.10 shows the sieve diagram for data from the larger sample of women (Kendall and

[8]Positive residuals are also shown by solid lines, negative residuals by broken lines, so that they may still be distinguished in monochrome versions.

Stuart (1961, Table 33.5) and Bishop et al. (1975, p. 284)). The *VisualAcuity* data is a frequency data frame and we first convert it to table form (VA), a $4 \times 4 \times 2$ table to re-label the variables and levels.

```
> # re-assign names/dimnames
> data("VisualAcuity", package = "vcd")
> VA <- xtabs(Freq ~ right + left + gender, data = VisualAcuity)
> dimnames(VA)[1:2] <- list(c("high", 2, 3, "low"))
> names(dimnames(VA))[1:2] <- paste(c("Right", "Left"), "eye grade")
> structable(aperm(VA))

                       Left eye grade high    2    3  low
gender Right eye grade
male   high                          821  112   85   35
       2                             116  494  145   27
       3                              72  151  583   87
       low                            43   34  106  331
female high                         1520  266  124   66
       2                             234 1512  432   78
       3                             117  362 1772  205
       low                            36   82  179  492
```

```
> sieve(VA[, , "female"], shade = TRUE)
```

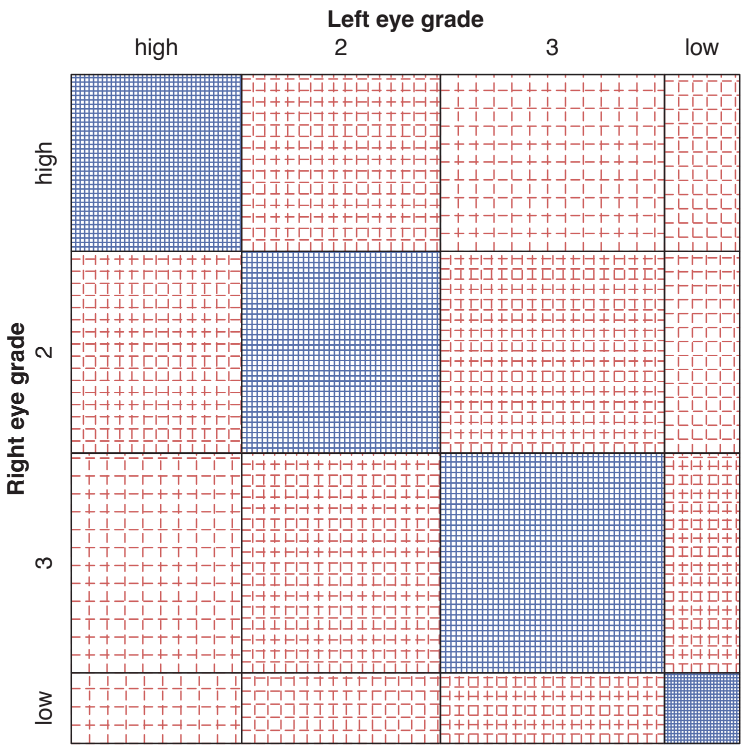


Figure 4.10: Vision classification for 7,477 women in Royal Ordnance factories. The high frequencies in the diagonal cells indicate the main association, but a subtler pattern also appears in the symmetric off-diagonal cells.

The diagonal cells show the obvious: people tend to have the same visual acuity in both eyes, and there is strong lack of independence. The off diagonal cells show a more subtle pattern that suggests symmetry—the cells below the diagonal are approximately equally dense as the corresponding cells above the diagonal. Moreover, the relatively consistent pattern on the diagonals $\pm1, \pm2, \ldots$ away from the main diagonals suggests that the association may be explained in terms of the *difference* in visual acuity between the two eyes.

These suggestions can be tested by fitting intermediate models between the null model of independence (which fits terribly) and the saturated model (which fits perfectly), as we shall see later in this book. A model of ***quasi-independence***, for example (see Example 10.5 in Chapter 9) ignores the diagonal cells and tests whether independence holds for the remainder of the table. The ***symmetry*** model for a square table allows association, but constrains the expected frequencies above and below the main diagonal to be equal. Such models provide a way of testing *specific* explanatory models that relate to substantive hypotheses and what we observe in our visualizations. These and other models for square tables are discussed further in Section 10.2. △

4.5.2 Larger tables: The strucplot framework

The implementation of sieve diagrams in vcd is far more general than illustrated in the examples above. For one thing, the `sieve` function has a formula method, which allows one to specify the variables in the display as a model formula. For example, for the *`VisualAcuity`* data, a plot of the (marginal) frequencies for left and right eye grades pooling over gender can be obtained with the call below (this plot is not shown).

```
> sieve(Freq ~ right + left, data = VisualAcuity, shade = TRUE)
```

More importantly, sieve diagrams are just one example of the ***strucplot framework***, a general system for visualizing n-way frequency tables in a hierarchical way. We describe this framework in more detail in Section 5.3 in the context of mosaic displays. For now, we just illustrate the extension of the formula method to provide for conditioning variables. In the call below, the formula `Freq ~ right + left | gender` means to produce a separate block in the plot for the levels of `gender`. The `set_varnames` argument relabels the variable names.

```
> sieve(Freq ~ right + left | gender, data = VisualAcuity,
+       shade = TRUE, set_varnames = c(right = "Right eye grade",
+                                      left = "Left eye grade"))
```

In Figure 4.11, the relative sizes of the blocks for the conditioning variable (`gender`) show the much larger number of women than men in this data. Within each block, color and density of the box rules shows the association of left and right acuity, and it appears that the pattern for men is similar to that observed for women.

An alternative way of visualizing stratified data is a ***coplot*** or ***conditioning plot***, which, for each stratum, shows an appropriate display for a subset of the data. Figure 4.12 visualizes separate sieve plots for men and women:

```
> cotabplot(VA, cond = "gender", panel = cotab_sieve, shade = TRUE)
```

The main difference to the extended sieve plots is that the distribution of the conditioning variable is not shown, which basically is a loss of information, but advantageous if the distribution of the conditioning variable(s) is highly skewed, since the partial displays of small strata will not be distorted.

The methods described in Section 4.3.2 can be used to test the hypothesis of homogeneity of association, and loglinear models described in Chapter 9 provide specific tests of hypotheses of ***symmetry***, ***quasi-independence***, and other models for structured associations.

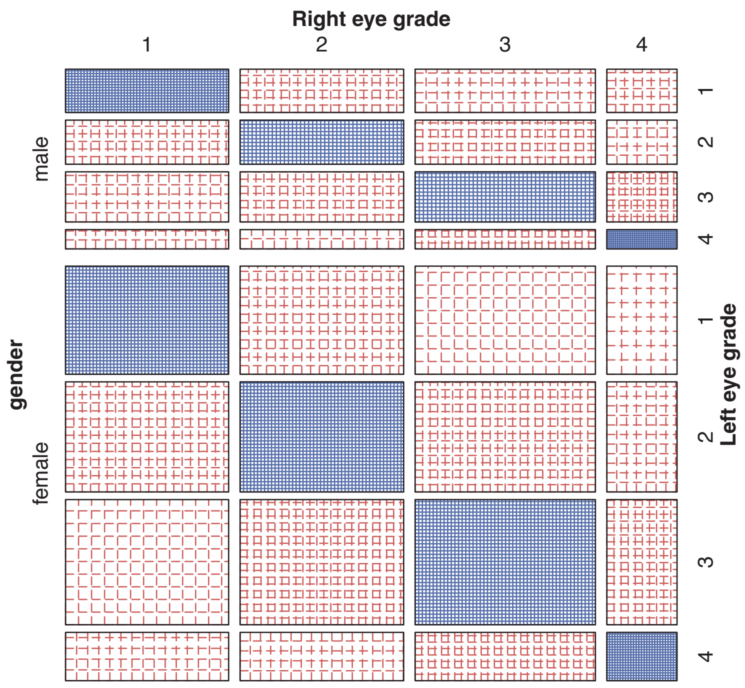


Figure 4.11: Sieve diagram for the three-way table of VisualAcuity, conditioned on gender.

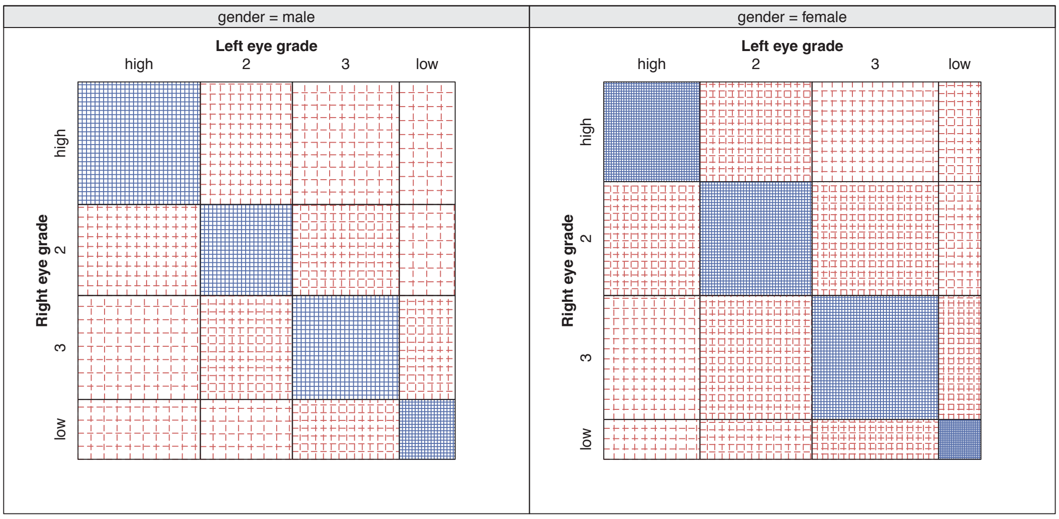


Figure 4.12: Conditional Sieve diagram for the three-way table of VisualAcuity, conditioned on gender.

Example 4.15: Berkeley admissions

This example illustrates some additional flexibility of sieve plots with the strucplot framework, using the Berkeley admissions data. The left panel of Figure 4.13 shows the sieve diagrams for the relation between department and admission, conditioned by gender. It can easily be seen that (a) overall, there were more male applicants than female; (b) there is a moderately similar pattern of observed $>$ expected (blue) for males and females.

```
> # conditioned on gender
> sieve(UCBAdmissions, shade = TRUE, condvar = 'Gender')
> # three-way table, Department first, with cell labels
> sieve(~ Dept + Admit + Gender, data = UCBAdmissions,
+         shade = TRUE, labeling = labeling_values,
+         gp_text = gpar(fontface = 2), abbreviate_labs = c(Gender = TRUE))
```

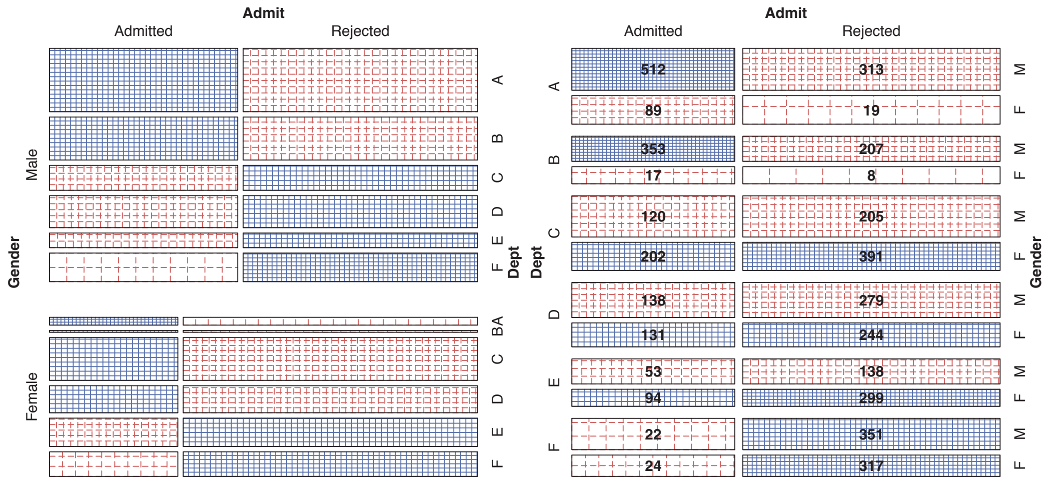


Figure 4.13: Sieve diagrams for the three-way table of the Berkeley admissions data. Left: Admit by Dept, conditioned on Gender; right: Dept re-ordered as the first splitting variable.

In the right panel of Figure 4.13, the three-way table was first permuted to make `Dept` the first splitting variable. Each 2 $\times$ 2 table of `Admit` by `Gender` then appears, giving a sieve diagram version of what we showed earlier in fourfold displays (Figure 4.6). The `labeling` argument is used here to write the cell frequency in each rectangle. `gp_text` renders them in bold font, and `abbreviate_labs` abbreviates the gender labels to avoid overplotting.

Alternatively, we can again use coplots to visualize conditioned sieve plots for this data. The following calls produce Figure 4.14 and Figure 4.15, with different conditioning and styles.

```
> cotabplot(UCBAdmissions, cond = "Gender", panel = cotab_sieve,
+              shade = TRUE)
```

```
> cotabplot(UCBAdmissions, cond = "Dept", panel = cotab_sieve,
+              shade = TRUE, labeling = labeling_values,
+              gp_text = gpar(fontface = "bold"))
```

Remark

Finally, for tables of more than two dimensions, there is a variety of different models for "independence" (discussed in Chapter 9 on log-linear models), and the strucplot framework allows these to

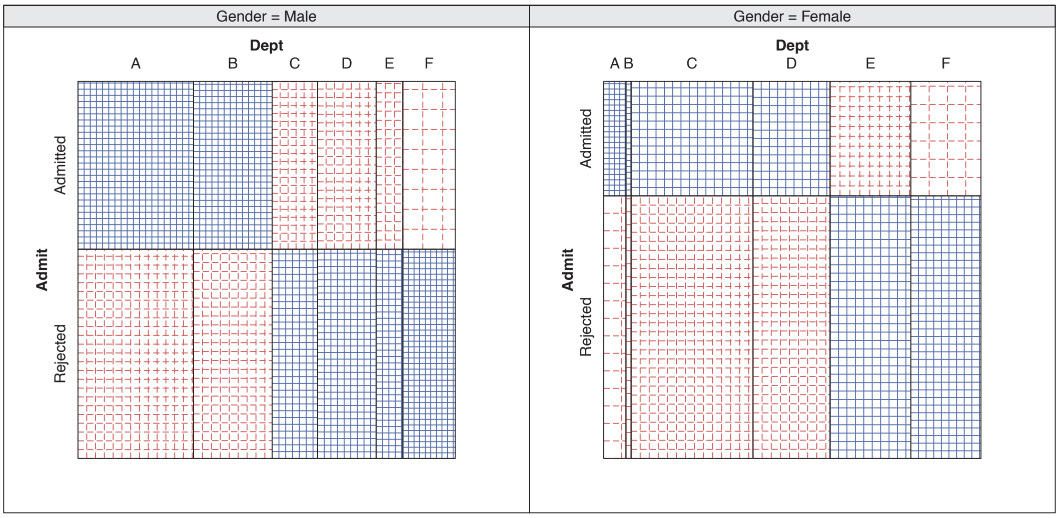


Figure 4.14: Conditional Sieve diagram for the three-way table of the Berkeley data, conditioned on gender.

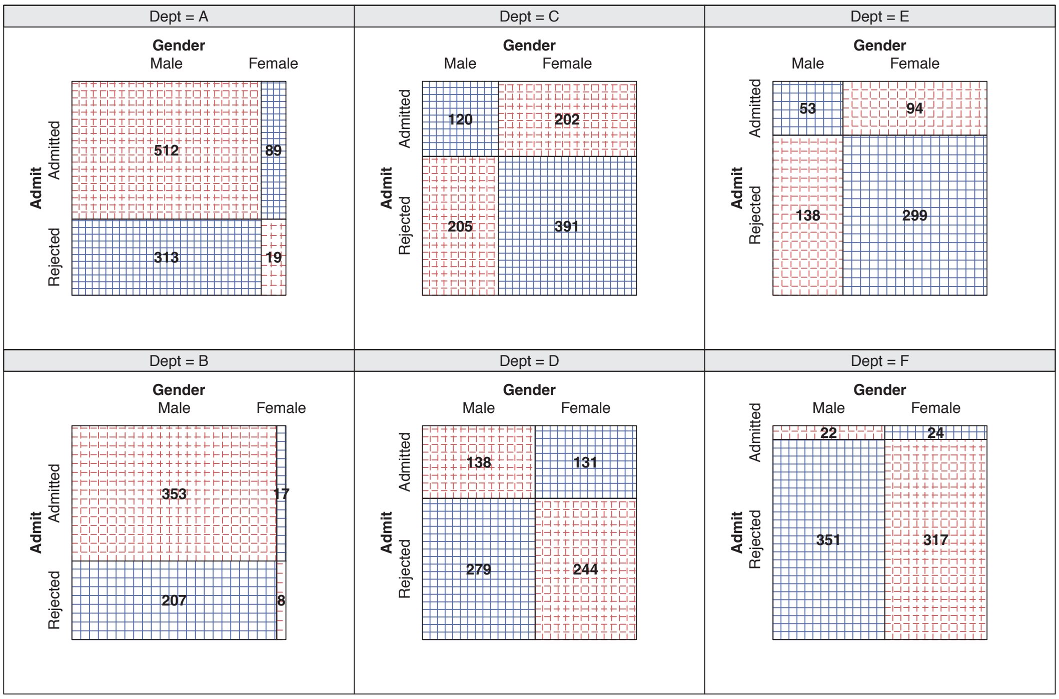


Figure 4.15: Conditional Sieve diagram for the three-way table of the Berkeley data, conditioned on department.

be specified with the `expected` argument, either as an array of numbers conforming to the `data` argument, or as a model formula for `loglm()`.

For example, a sieve diagram may be used to determine if the association between gender and department is the same across departments by fitting the model ~ `Admit * Gender + Dept`, which says that `Dept` is independent of the combinations of `Admit` and `Gender`. This is done as shown below, giving the plot in Figure 4.16.

```
> UCB2 <- aperm(UCBAdmissions, c(3, 2, 1))
> sieve(UCB2, shade = TRUE, expected = ~ Admit * Gender + Dept,
+       split_vertical = c(FALSE, TRUE, TRUE))
```

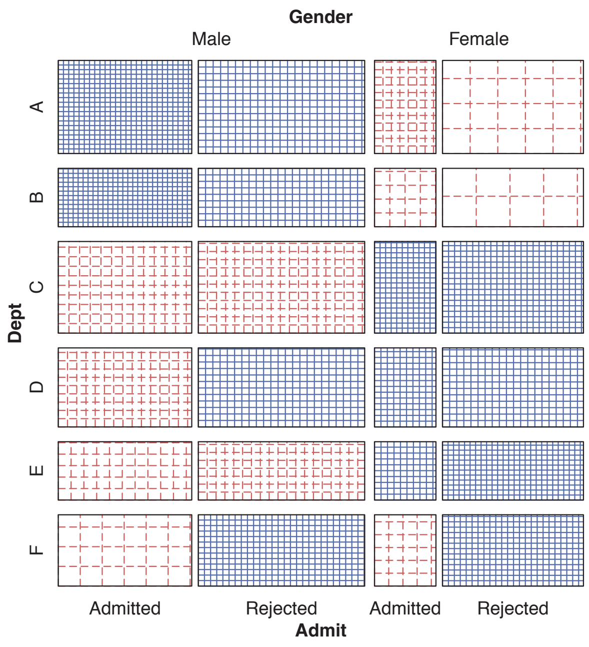


Figure 4.16: Sieve diagram for the Berkeley admissions data, fitting the model of joint independence, Admit * Gender + Dept.

In terms of the loglinear models discussed in Chapter 5, this is equivalent to fitting the model of ***joint independence***, [Admit Gender][Dept]. Figure 4.16 shows the greater numbers of male applicants in departments A and B (whose overall rate of admission is high) and greater numbers of female applicants in the remaining departments (where the admission rate is low).

△

4.6 Association plots

In the ***sieve diagram*** the foreground (rectangles) shows expected frequencies; deviations from independence are shown by color and density of shading. The ***association plot*** (Cohen, 1980, Friendly, 1991) puts deviations from independence in the foreground: the area of each box is made proportional to the (observed − expected) frequency.

For a two-way contingency table, the signed contribution to Pearson χ^2 for cell $i,\ j$ is

$$r_{ij} = \frac{n_{ij} - m_{ij}}{\sqrt{m_{ij}}} = \text{Pearson residual}, \qquad \chi^2 = \sum_{i,j} r_{ij}^2 \,. \tag{4.5}$$

In the association plot, each cell is shown by a rectangle, having:

- (signed) height $\sim r_{ij}$,
- width = $\sqrt{m_{ij}}$,

so, the area of each cell is proportional to the raw residual, $n_{ij} - m_{ij}$. The rectangles for each row in the table are positioned relative to a baseline representing independence ($r_{ij} = 0$) shown by a dotted line. Cells with observed $>$ expected frequency rise above the line (and are colored blue); cells that contain less than the expected frequency fall below it (and are shaded red).

```
> assoc(~ Hair + Eye, data = HairEyeColor, shade = TRUE)
> assoc(HairEyeColor, shade = TRUE)
```

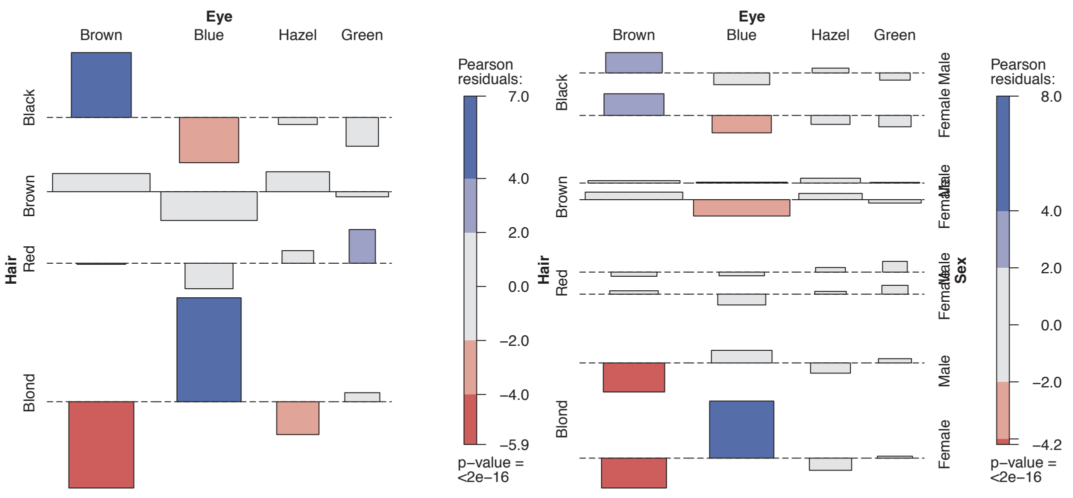


Figure 4.17: Association plot for the hair-color eye-color data. Left: marginal table, collapsed over gender; right: full table.

Figure 4.17 (left) shows the association plot for the data on hair color and eye color. In constructing this plot, each rectangle is shaded according to the value of the Pearson residual from Eqn. (4.5), using a simple scale shown in the legend, where residuals $|r_{ij}| > 2$ are shaded blue or red depending on their sign, and residuals $|r_{ij}| > 4$ are shaded with a more saturated color.

One virtue of the association plot is that it is quite simple to interpret in terms of the pattern of positive and negative r_{ij} values. Bertin (1981) uses similar graphics to display large complex contingency tables. Like the sieve diagram, however, patterns of association are most apparent when the rows and columns of the display are ordered in a sensible way.

We note here that the association plot also belongs to the strucplot framework and thus extends to higher-way tables. For example, the full *HairEyeColor* table is also classified by `Sex`. The plot for the three-way table is shown in Figure 4.17 (right). In this plot the third table variable (`Sex` here) is shown nested within the first two, allowing easy comparison of the profiles of hair and eye color for males and females.

4.7 Observer agreement

When the row and column variables represent different observers' rating the same subjects or objects, interest is focused on ***observer agreement*** rather than mere association. In this case, measures 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 $4 \times 4 \times 2$ 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
                Probable         3         11          4          0
                Possible         2         13          3          4
                Doubtful         1          2          4         14

> apply(MSPatients, 3, sum)        # show sample sizes

   Winnipeg New Orleans
        149          69
```

In this example, note that the distribution of degree of severity of MS may differ between the two patient samples. As well, for a given sample, the two neurologists may be more or less strict about the boundaries between the rating categories.

△

4.7.1 Measuring agreement

In assessing the strength of *agreement* we usually have a more stringent criterion than in measuring the strength of *association*, because observers ratings can be strongly associated without strong agreement. For example, one rater could use a more stringent criterion and thus consistently rate subjects one category lower (on an ordinal scale) than another rater.

More generally, measures of agreement must take account of the marginal frequencies with which two raters use the categories. If observers tend to use the categories with different frequency, this will affect measures of agreement.

Here we describe some simple indices that summarize agreement with a single score (and associated standard errors or confidence intervals). Von Eye and Mun (2006) treat this topic from the perspective of loglinear models.

4.7.1.1 Intraclass correlation

An analysis of variance framework leads to the **intraclass correlation** as a measure of inter-rater reliability, particularly when there are more than two raters. This approach is not covered here, but various applications are described by Shrout and Fleiss (1979), and implemented in R in `ICC()` in the psych (Revelle, 2015) package.

4.7.1.2 Cohen's Kappa

Cohen's kappa (κ) (Cohen, 1960, 1968) is a commonly used measure of agreement that compares the observed agreement to agreement expected by chance if the two observer's ratings were independent. If p_{ij} is the probability that a randomly selected subject is rated in category i by the first observer and in category j by the other, then the observed agreement is the sum of the diagonal entries, $P_o = \sum_i p_{ii}$. If the ratings were independent, this probability of agreement (by chance) would be $P_c = \sum_i p_{i+}\, p_{+i}$. Cohen's κ is then the ratio of the difference between actual agreement and chance agreement, $P_o - P_c$, to the maximum value this difference could obtain:

$$\kappa = \frac{P_o - P_c}{1 - P_c} \; . \tag{4.6}$$

When agreement is perfect, $\kappa = 1$; when agreement is no better than would be obtained from statistically independent ratings, $\kappa = 0$. κ could conceivably be negative, but this rarely occurs in practice. The minimum possible value depends on the marginal totals.

For large samples (n_{++}), κ has an approximate normal distribution when $H_0 : \kappa = 0$ is true and its standard error (Fleiss, 1973, Fleiss et al., 1969) is given by

$$\hat{\sigma}(\kappa) = \frac{P_c + P_c^2 - \sum_i p_{i+} p_{+i} (p_{i+} + p_{+i})}{n_{++}(1 - P_c)^2} \; .$$

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, $0 \le w_{ij} \le 1$ 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,

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

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")
```

4.7.2.2 Partial agreement

Partial agreement is allowed by including a weighted contribution from off-diagonal cells, b steps from the main diagonal. For a given cell frequency, n_{ij}, a pattern of weights, $w_1, w_2, \ldots, w_b$ is applied to the cell frequencies as shown schematically below:

$$\begin{array}{ccccc}
 & & n_{i-b,i} & & \\
 & & \vdots & & \\
n_{i,i-b} & \cdots & n_{i,i} & \cdots & n_{i,i+b} \\
 & & \vdots & & \\
 & & n_{i+b,i} & &
\end{array}
\quad \Leftarrow \quad
\begin{array}{ccccc}
 & & w_b & & \\
 & & \vdots & & \\
w_b & \cdots & 1 & \cdots & w_b \\
 & & \vdots & & \\
 & & w_b & &
\end{array}$$

These weights are incorporated in the agreement chart (right panel of Figure 4.18) by successively lighter shaded rectangles whose size is proportional to the sum of the cell frequencies, denoted

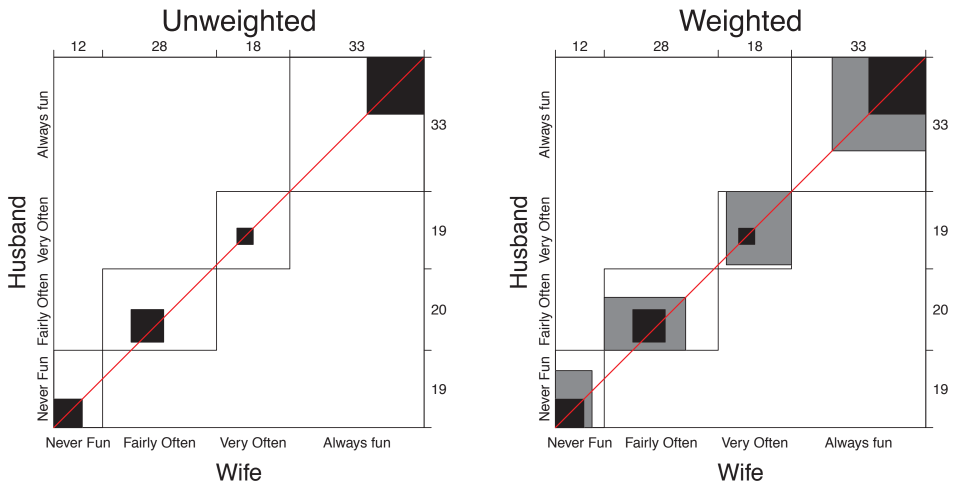


Figure 4.18: Agreement charts for husbands' and wives' sexual fun. Left: unweighted chart, showing only exact agreement; right: weighted chart, using weight $w_1 = 8/9$ for a one-step disagreement.

A_{bi}, shown above. A_{1i} allows 1-step disagreements, using weights 1 and w_1; A_{2i} includes 2-step disagreements, etc. From this, one can define a weighted measure of agreement, B^w, analogous to weighted κ:

$$B^w = \frac{\text{weighted sum of areas of agreement}}{\text{area of rectangles}} = 1 - \frac{\sum_i^k [n_{i+}n_{+i} - n_{ii}^2 - \sum_{b=1}^q w_b A_{bi}]}{\sum_i^k n_{i+}\, n_{+i}}$$

where w_b is the weight for A_{bi}, the shaded area b steps away from the main diagonal, and q is the furthest level of partial disagreement to be considered.

The function `agreementplot()` actually calculates both B and B^w and returns them invisibly as the result of the call. The results, $B = 0.146$, and $B^w = 0.498$, indicate a stronger degree of agreement when 1-step disagreements are included.

```
> B <- agreementplot(SexualFun)
> unlist(B)[1 : 2]

           Bangdiwala Bangdiwala_Weighted
              0.14646             0.49817
```

Example 4.18: Mammogram ratings

The *Mammograms* data in vcdExtra gives a 4×4 table of (probably contrived) ratings of 110 mammograms by two raters from Kundel and Polansky (2003), used to illustrate the calculation and interpretation of agreement measures in this context.[9]

```
> data("Mammograms", package = "vcdExtra")
> B <- agreementplot(Mammograms, main = "Mammogram ratings")
```

The agreement plot in Figure 4.19 shows substantial agreement among the two raters, particularly when one-step disagreements are taken into account. Careful study of this graph shows that

[9]In practice, of course, rater agreement on severity of diagnosis from radiology images varies with many factors. See Antonio and Crespi (2010) for a meta-analytic study concerning agreement in breast cancer diagnosis.

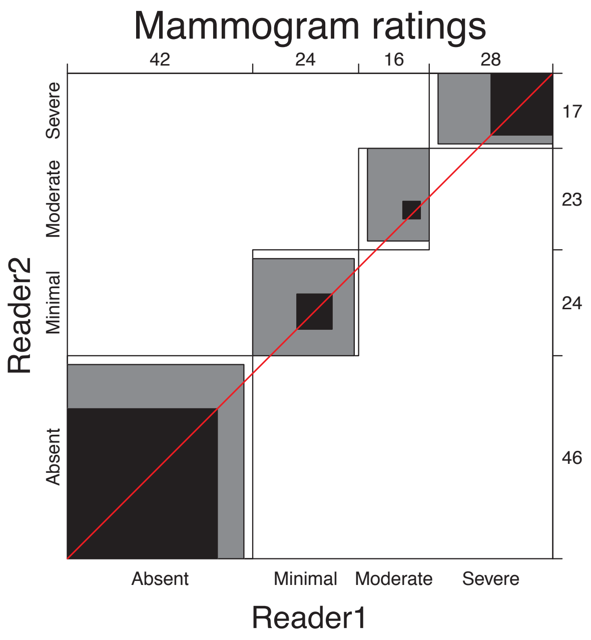


Figure 4.19: Agreement plot for the Mammograms data.

the two raters more often agree exactly for the extreme categories of "Absent" and "Severe." The amounts of unweighted and weighted agreement are shown numerically in the B and B^w statistics.

```
> unlist(B)[1 : 2]

          Bangdiwala Bangdiwala_Weighted
             0.42721             0.83665
```

△

4.7.3 Observer bias in agreement

With an ordered scale, it may happen that one observer consistently tends to classify the objects into higher or lower categories than the other, perhaps due to using stricter thresholds for the boundaries between adjacent categories. This bias produces differences in the marginal totals, (n_{i+} and n_{+i}), and decreases the maximum possible agreement. While special tests exist for ***marginal homogeneity***, the observer agreement chart shows this directly by the relation of the dark squares to the diagonal line: When the marginal totals are the same, the squares fall along the diagonal. The measures of agreement, κ and B, cannot determine whether lack of agreement is due to such bias, but the agreement chart can detect this.

EXAMPLE 4.19: Diagnosis of MS patients

Agreement charts for both patient samples in the *MSPatients* data are shown in Figure 4.20. The `agreementplot()` function only handles two-way tables, so we use `cotabplot()` with the `agreementplot` panel function to handle the `Patients` stratum:

```
> cotabplot(MSPatients, cond = "Patients", panel = cotab_agreementplot,
+             text_gp = gpar(fontsize = 18), xlab_rot=20)
```

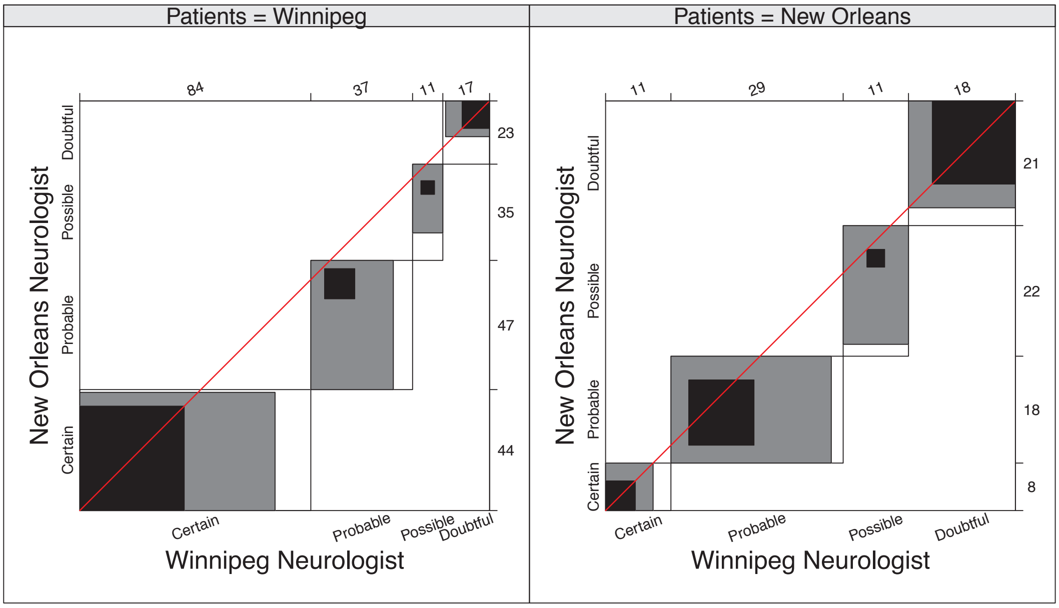


Figure 4.20: Weighted agreement charts for both patient samples in the MSPatients data. Departure of the middle rectangles from the diagonal indicates lack of marginal homogeneity.

It can be seen that, for both groups of patients, the rectangles for the two intermediate categories lie largely below the diagonal line (representing equality). This indicates that the Winnipeg neurologist tends to classify patients into more severe diagnostic categories. The departure from the diagonal is greater for the Winnipeg patients, for whom the Winnipeg neurologist uses the two most severe diagnostic categories very often, as can also be seen from the marginal totals printed in the plot margins.

Nevertheless, there is a reasonable amount of agreement if one-step disagreements are allowed, as can be seen in Figure 4.20 and quantified in the B^w statistics below. The agreement charts also serve to explain why the B measures for exact agreement are so much lower.

```
> agr1 <- agreementplot(MSPatients[, , "Winnipeg"])
> agr2 <- agreementplot(MSPatients[, , "New Orleans"])
> rbind(Winnipeg = unlist(agr1), NewOrleans = unlist(agr2))[, 1 : 2]

            Bangdiwala Bangdiwala_Weighted
Winnipeg       0.27210             0.73808
NewOrleans     0.28537             0.82231
```

△

4.8 Trilinear plots

The ***trilinear plot*** (also called a *ternary diagram* or *trinomial plot*) is a specialized display for a 3-column contingency table or for three variables whose relative proportions are to be displayed. Individuals may be assigned to one of three diagnostic categories, for example, or a chemical process may yield three constituents in varying proportions, or we may look at the division of votes among three parties in a parliamentary election. This display is useful, therefore, for both frequencies and proportions.

Trilinear plots are featured prominently in Aitchison (1986), who describes statistical models for this type of ***compositional data***. Upton (1976, 1994) uses them in detailed analyses of spatial and temporal changes in British general elections. Wainer (1996) reviews a variety of other uses of trilinear plots and applies them to aid in understanding the distributions of students' achievement in the National Assessment of Educational Progress, making some aesthetic improvements to the traditional form of these plots along the way.

A trilinear plot displays each observation as a point inside an equilateral triangle whose coordinates correspond to the relative proportions in each column. The three vertices represent the three extremes when 100% occurs in one of the three columns; a point in the exact center corresponds to equal proportions of $\frac{1}{3}$ in all three columns. In fact, each point represents the (weighted) barycenter of the triangle, the coordinates representing weights placed at the corresponding vertices. For instance, Figure 4.21 shows three points whose compositions of three variables, A, B, and C, are given in the data frame `DATA` below.

```
> library(ggtern)
> DATA <- data.frame(
+   A = c(40, 20, 10),
+   B = c(30, 60, 10),
+   C = c(30, 20, 80),
+   id = c("1", "2", "3"))
>
> aesthetic_mapping <- aes(x = C, y = A, z = B, colour = id)
> ggtern(data = DATA, mapping = aesthetic_mapping) +
+     geom_point(size = 4) +
+     theme_rgbw()
```

(The plot shown requires some more cosmetic parameters not shown for simplicity).

Note that each apex corresponds to 100% of the labeled variable, and the percentage of this variable decreases linearly along a line to the midpoint of the opposite baseline. The grid lines in the figure show the percentage value along each axis.

The construction of trilinear plots is described in detail in http://en.wikipedia.org/

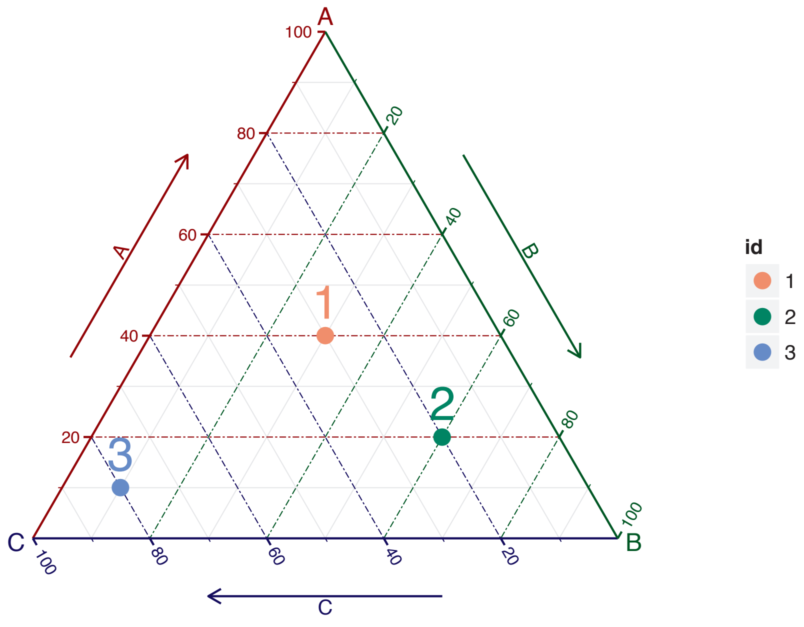


Figure 4.21: A trilinear plot showing three points, for variables A, B, C.

wiki/Ternary_plot. Briefly, let $P(a, b, c)$ represent the three components normalized so that $a + b + c = 1.0$. If the apex corresponding to Point A in Figure 4.21 is given (x, y) coordinates of $(x_A, y_A) = (0, 0)$, and those at apex B are $(x_B, y_B) = (100, 0)$, then the coordinates of apex C are $(x_C, y_C) = (50, 50\sqrt{3})$. The cartesian coordinates (x_P, y_P) of point P are then calculated as

$$\begin{aligned} y_P &= c\, y_C \\ x_P &= y_P \left(\frac{y_C - y_B}{x_C - x_B} \right) + \frac{\sqrt{3}}{2} y_C (1 - a) \,. \end{aligned}$$

In R, trilinear plots are implemented in the `triplot()` function in the TeachingDemos (Snow, 2013) package, and also in the ggtern (Hamilton, 2014) package, an extension of the ggplot2 framework. The latter is much more flexible, because it inherits all of the capabilities of ggplot2 for plot annotations, faceting, and layers. In essence, the function `ggtern()` is just a wrapper for `ggplot(...)`, which adds a change in the coordinate system from cartesian (x, y) coordinates to the ternary coordinate system with `coord_tern()`.

EXAMPLE 4.20: Lifeboats on the *Titanic*

We examine the question of who survived and why in the sinking of the *RMS Titanic* in Chapter 5 (Example 5.19, Example 5.21, Exercise 5.12), where we analyze a four-way table, *`Titanic`*, of the 2,201 people on board (1,316 passengers and 885 crew), classified by `Class`, `Sex`, `Age`, and `Survival`. A related data set, *`Lifeboats`* in vcd, tabulates the survivors according to the lifeboats on which they were loaded. This data sheds some additional light on the issue of survival and provides a nice illustration of trilinear plots.

A bit of background: after the disaster, the British Board of Trade launched several inquiries, the most comprehensive of which resulted in the *Report on the Loss of the "Titanic" (S.S.)* by Lord Mersey (Mersey, 1912).[10] The data frame *`Lifeboats`* in vcd contains the data listed on page 38 of that report.[11]

Of interest here is the composition of the boats by the three categories: men, women and children, and crew, and according to the launching of the boats from the port or starboard side. This can be shown in a trilinear display using the following statements. The plot, shown in Figure 4.22, has most of the points near the bottom left, corresponding to a high percentage of women and children. We create a variable, `id`, used to label those boats with more than 10% male passengers. In the `ggplot2` framework, plot aesthetics such as color and shape can be mapped to variables in the data set, and here we map these both to `side` of the boat.

```
> data("Lifeboats", package = "vcd")
> # label boats with more than 10% men
> Lifeboats$id <- ifelse(Lifeboats$men / Lifeboats$total > .1,
+                        as.character(Lifeboats$boat), "")
>
> AES <- aes(x = women, y = men, z = crew, colour = side, shape = side,
+            label = id)
> ggtern(data = Lifeboats, mapping = AES) +
+     geom_text() +
+     geom_point(size=2) +
+     geom_smooth_tern(method = "lm", alpha = 0.2)
```

[10]The *Titanic* was outfitted with 20 boats, half on each of the port and starboard sides, of which 14 were large lifeboats with a capacity of 65, two were emergency boats designed for 40 persons, and the remaining four were collapsible boats capable of holding 47, a total capacity of 1,178 (considered adequate at that time). Two of the collapsible boats, lashed to the roof of the officers' quarters, were ineffectively launched and utilized as rafts after the ship sunk. The report lists the time of launch and composition of the remaining 18 boats according to male passengers, women and children, and "men of crew," as reported by witnesses.

[11]The "data" lists a total of 854 in 18 boats, although only 712 were in fact saved. Mersey notes "it is obvious that these figures are quite unreliable."

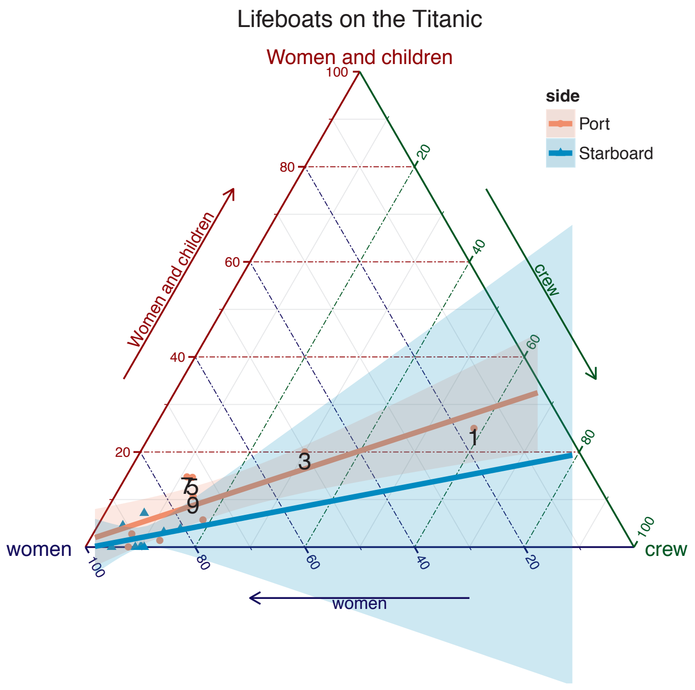


Figure 4.22: Lifeboats on the *Titanic*, showing the composition of each boat. Boats with more than 10% male passengers are labeled.

The resulting plot in Figure 4.22 (for which some more cosmetic parameters than shown in the code above have been used) makes it immediately apparent that many of the boats launched from the port side differ substantially from the starboard boats, whose passengers were almost entirely women and children. Boat 1 had only 20% (2 out of 10) women and children, while the percentage for boat 3 was only 50% (25 out of 50). We highlight the difference in composition of the boats launched from the two sides by adding seperate linear regression lines for the relation `men ~ women`.

The trilinear plot scales the numbers for each observation to sum to 1.0, so differences in the total number of people on each boat cannot be seen in Figure 4.22. The `total` number reported loaded is plotted against `launch` time in Figure 4.23, with a separate regression line and loess smooth fit to the data for the port and starboard sides (code again simplified for clarity):

```
> AES <- aes(x = launch, y = total, colour = side, label = boat)
> ggplot(data = Lifeboats, mapping = AES) +
+       geom_text() +
+       geom_smooth(method = "lm", aes(fill = side), size = 1.5) +
+       geom_smooth(method = "loess", aes(fill = side), se = FALSE,
+                   size = 1.2)
```

From the linear regression lines in Figure 4.23, it seems that the rescue effort began in panic on the port side, with relatively small numbers loaded, and (from Figure 4.22), small proportions of women and children. But the loading regime on that side improved steadily over time. The procedures began more efficiently on the starboard side but the numbers loaded increased only slightly. The smoothed loess curves indicate that over time, for each side, there was still a large variability from boat to boat.

△

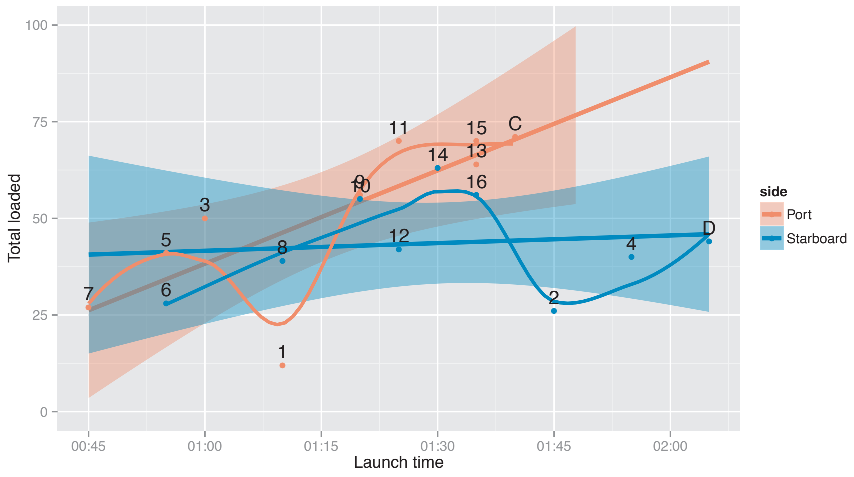


Figure 4.23: Number of people loaded on lifeboats on the Titanic vs. time of launch, by side of boat. The plot annotations show the linear regression and loess smooth.

4.9 Chapter summary

- A contingency table gives the frequencies of observations cross-classified by two or more categorical variables. With such data we are typically interested in testing whether associations exist, quantifying the strength of association, and understanding the nature of the association among these variables.

- For 2×2 tables, association is easily summarized in terms of the odds ratio or its logarithm. This measure can be extended to stratified $2 \times 2 \times k$ tables, where we can also assess whether the odds ratios are equal across strata or how they vary.

- For $R \times C$ tables, measures and tests of general association between two categorical variables are most typically carried out using the Pearson's chi-squared or likelihood-ratio tests provided by `assocstats()`. Stratified tests controlling for one or more background variables, and tests for ordinal categories, are provided by the Cochran–Mantel–Haenszel tests given by `CMHtest()`.

- For 2×2 tables, the fourfold display provides a visualization of the association between variables in terms of the odds ratio. Confidence rings provide a visual test of whether the odds ratio differs significantly from 1. Stratified plots for $2 \times 2 \times k$ tables are also provided by `fourfold()`.

- Sieve diagrams and association plots provide other useful displays of the pattern of association in $R \times C$ tables. These also extend to higher-way tables as part of the strucplot framework.

- When the row and column variables represent different observers rating the same subjects, interest is focused on agreement rather than mere association. Cohen's κ is one measure of strength of agreement. The observer agreement chart provides a visual display of how the observers agree and disagree.

- Another specialized display, the trilinear plot, is useful for three-column frequency tables or compositional data.

4.10 Lab exercises

Exercise 4.1 The data set `fat`, created below, gives a 2×2 table recording the level of cholesterol in diet and the presence of symptoms of heart disease for a sample of 23 people.

```
> fat <- matrix(c(6, 4, 2, 11), 2, 2)
> dimnames(fat) <- list(diet = c("LoChol", "HiChol"),
+                       disease = c("No", "Yes"))
```

(a) Use `chisq.test(fat)` to test for association between diet and disease. Is there any indication that this test may not be appropriate here?
(b) Use a fourfold display to test this association visually. Experiment with the different options for standardizing the margins, using the `margin` argument to `fourfold()`. What evidence is shown in different displays regarding whether the odds ratio differs significantly from 1?
(c) `oddsratio(fat, log = FALSE)` will give you a numerical answer. How does this compare to your visual impression from fourfold displays?
(d) With such a small sample, Fisher's exact test may be more reliable for statistical inference. Use `fisher.test(fat)`, and compare these results to what you have observed before.
(e) Write a one-paragraph summary of your findings and conclusions for this data set.

Exercise 4.2 The data set *`Abortion`* in vcdExtra gives a $2 \times 2 \times 2$ table of opinions regarding abortion in relation to sex and status of the respondent. This table has the following structure:

```
> data("Abortion", package = "vcdExtra")
> str(Abortion)

 table [1:2, 1:2, 1:2] 171 152 138 167 79 148 112 133
 - attr(*, "dimnames")=List of 3
  ..$ Sex             : chr [1:2] "Female" "Male"
  ..$ Status          : chr [1:2] "Lo" "Hi"
  ..$ Support_Abortion: chr [1:2] "Yes" "No"
```

(a) Taking support for abortion as the outcome variable, produce fourfold displays showing the association with sex, stratified by status.
(b) Do the same for the association of support for abortion with status, stratified by sex.
(c) For each of the problems above, use `oddsratio()` to calculate the numerical values of the odds ratio, as stratified in the question.
(d) Write a brief summary of how support for abortion depends on sex and status.

Exercise 4.3 The *`JobSat`* table on income and job satisfaction created in Example 2.5 is contained in the vcdExtra package.

(a) Carry out a standard χ^2 test for association between income and job satisfaction. Is there any indication that this test might not be appropriate? Repeat this test using `simulate.p.value = TRUE` to obtain a Monte Carlo test that does not depend on large sample size. Does this change your conclusion?
(b) Both variables are ordinal, so CMH tests may be more powerful here. Carry out that analysis. What do you conclude?

Exercise 4.4 The *`Hospital`* data in vcd gives a 3×3 table relating the length of stay (in years) of 132 long-term schizophrenic patients in two London mental hospitals with the frequency of visits by family and friends.

(a) Carry out a χ^2 test for association between the two variables.
(b) Use `assocstats()` to compute association statistics. How would you describe the strength of association here?
(c) Produce an association plot for these data, with visit frequency as the vertical variable. Describe the pattern of the relation you see here.
(d) Both variables can be considered ordinal, so `CMHtest()` may be useful here. Carry out that analysis. Do any of the tests lead to different conclusions?

Exercise 4.5 Continuing with the *Hospital* data:

(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
	Con	24	8	13	45
Siskel	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()`.

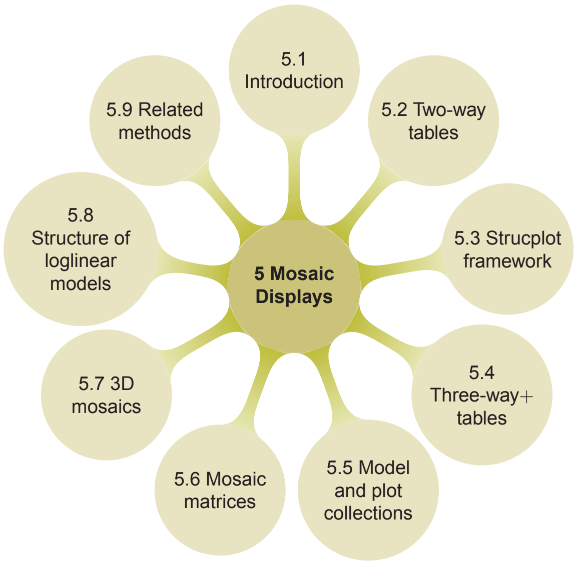
Exercise 4.8 For the *VisualAcuity* data set:

(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 display is similar to the sieve diagram (Section 4.5). However, mosaic plots and related methods described here:

- generalize more readily to n-way tables. One can usefully examine 3-way, 4-way, and even larger tables, subject to the limitations of resolution in any graph;
- are intimately connected to loglinear models, generalized linear models, and generalized non-linear models for frequency data;
- provide a method for fitting a series of sequential loglinear models to the various marginal totals of an n-way table; and
- can be used to illustrate the relations among variables that are fitted by various loglinear models.

The basic ideas behind these graphical methods are explained for two-way tables in Section 5.2; the ***strucplot framework*** on which these are based is described in Section 5.3. The graphical extension of mosaic plots to three-way and large tables (Section 5.4) is quite direct. However, the details require a brief introduction to loglinear models and some terminology for different types of "independence" in such tables, also described in this section. Mosaic methods are further extended to all-pairwise plots in Section 5.6 and 3D plots in Section 5.7.

5.2 Two-way tables

The mosaic display (Friendly, 1992, 1994b, 1997, Hartigan and Kleiner, 1981, 1984) is like a grouped barchart, where the heights (or widths) of the bars show the relative frequencies of one variable, and widths (heights) of the sections in each bar show the conditional frequencies of the second variable, given the first. This gives an area-proportional visualization of the frequencies composed of tiles corresponding to the cells created by successive vertical and horizontal splits of rectangle, representing the total frequency in the table. The construction of the mosaic display, and what it reveals, are most easily understood for two-way tables.

EXAMPLE 5.1: Hair color and eye color

Consider the data shown earlier in Table 4.2, showing the relation between hair color and eye color among students in a statistics course. The basic mosaic display for this 4×4 table is shown in Figure 5.1.

```
> data("HairEyeColor", package = "datasets")
> haireye <- margin.table(HairEyeColor, 1 : 2)
> mosaic(haireye, labeling = labeling_values)
```

For such a two-way table, the mosaic in Figure 5.1 is constructed by first dividing a unit square in proportion to the marginal totals of one variable, say, hair color.

For these data, the marginal frequencies and proportions of hair color are calculated below:

```
> (hair <- margin.table(haireye, 1))

Hair
Black Brown   Red Blond
  108   286    71   127

> prop.table(hair)

Hair
  Black   Brown     Red   Blond
0.18243 0.48311 0.11993 0.21453
```

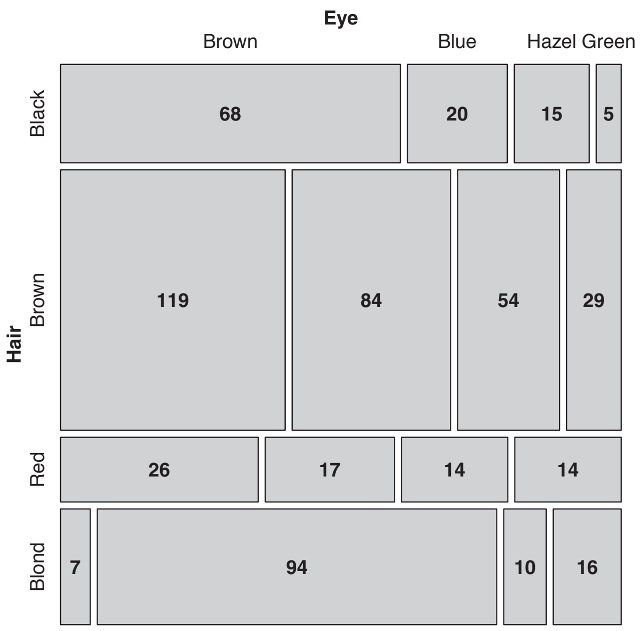


Figure 5.1: Basic mosaic display for hair color and eye color data. The area of each rectangle is proportional to the observed frequency in that cell, shown as numbers.

These frequencies can be shown as the mosaic for the first variable (hair color), with the unit square split according to the marginal proportions as in Figure 5.2 (left). The rectangular tiles are then shaded to show the residuals (deviations) from a particular model as shown in the right panel of Figure 5.2. The details of the calculations for shading are:

- The one-way table of marginal totals can be fit to a model, in this case, the (implausible) model that all hair colors are equally probable. This model has expected frequencies $m_i = 592/4 = 148$:

```
> expected <- rep(sum(hair) / 4, 4)
> names(expected) <- names(hair)
> expected

Black Brown   Red Blond
  148   148   148   148
```

- The Pearson residuals from this model, $r_i = (n_i - m_i)/\sqrt{m_i}$, are:

```
> (residuals <- (hair - expected) / sqrt(expected))

Hair
  Black   Brown     Red   Blond
-3.2880 11.3435 -6.3294 -1.7262
```

and these values are shown by color and shading as shown in the legend in Figure 5.3. The high positive value for brown hair indicates that people with brown hair are much more frequent in this sample than the equiprobability model would predict; the large negative residual for red hair shows that redheads are much less common. Further details of the schemes for shading are described below, but essentially we use increasing intensities of blue (red) for positive (negative) residuals.

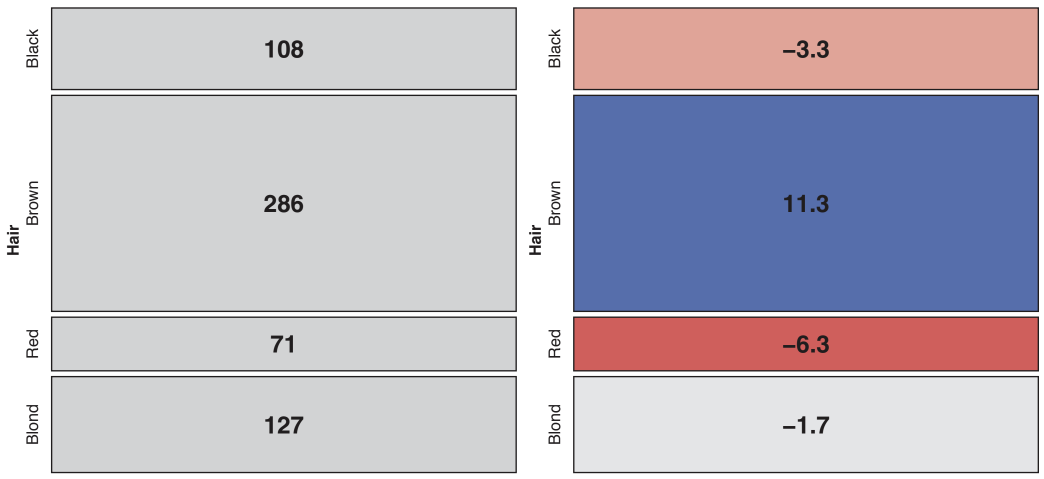


Figure 5.2: First step in constructing a mosaic display. Left: splitting the unit square according to frequencies of hair color; right: shading the tiles according to residuals from a model of equal marginal probabilities.

In the next step, the rectangle for each hair color is subdivided in proportion to the *relative* (conditional) frequencies of the second variable—eye color, giving the following conditional row proportions:

```
> round(addmargins(prop.table(haireye, 1), 2), 3)

       Eye
Hair     Brown  Blue Hazel Green   Sum
  Black 0.630 0.185 0.139 0.046 1.000
  Brown 0.416 0.294 0.189 0.101 1.000
  Red   0.366 0.239 0.197 0.197 1.000
  Blond 0.055 0.740 0.079 0.126 1.000
```

The proportions in each row determine the width of the tiles in the second mosaic display in Figure 5.3.

- Again, the cells are shaded in relation to standardized Pearson residuals, $r_{ij} = (n_{ij} - m_{ij})/\sqrt{m_{ij}}$, from a model. For a two-way table, the model is that hair color and eye color are independent in the population from which this sample was drawn. These residuals are calculated as shown below using `independence_table()` to calculate the expected values m_{ij} under this model ($m_{ij} = n_{++}\pi_{i+}\pi_{+j}$):

```
> exp <- independence_table(haireye)
> resids <- (haireye - exp) / sqrt(exp)
> round(resids, 2)

       Eye
Hair     Brown  Blue Hazel Green
  Black  4.40 -3.07 -0.48 -1.95
  Brown  1.23 -1.95  1.35 -0.35
  Red   -0.07 -1.73  0.85  2.28
  Blond -5.85  7.05 -2.23  0.61
```

- Thus, in Figure 5.3, the two tiles shaded deep blue correspond to the two cells, (Black, Brown) and (Blond, Blue), whose residuals are greater than $+4$, indicating much greater frequency in

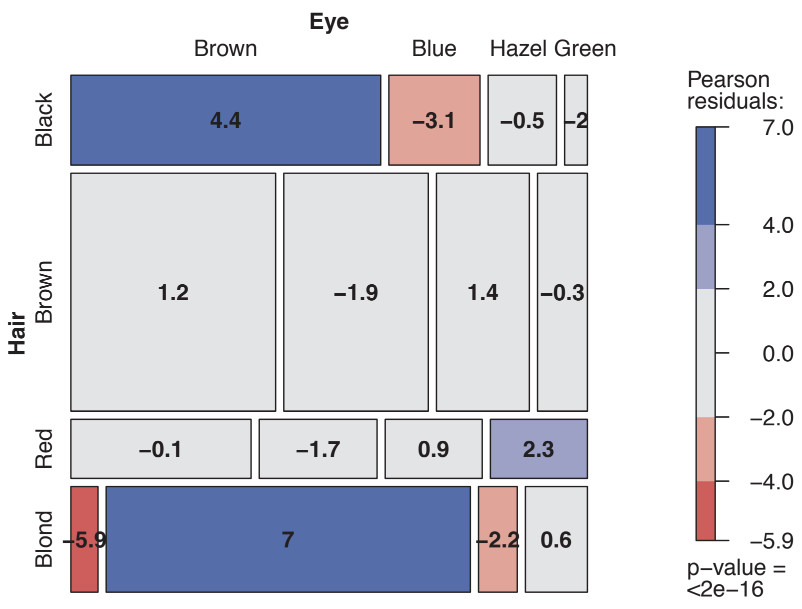


Figure 5.3: Second step in constructing the mosaic display. Each rectangle for hair color is subdivided in proportion to the relative frequencies of eye color, and the tiles are shaded in relation to residuals from the model of independence.

those cells than would be found if hair color and eye color were independent. The tile shaded deep red, (Blond, Brown), corresponds to the largest negative residual = -5.85, indicating this combination is extremely rare under the hypothesis of independence.

- The overall Pearson χ^2 statistic for the independence model is just the sum of squares of the residuals, with degrees of freedom $(r-1) \times (c-1)$.

```
> (chisq <- sum(resids ^ 2))

[1] 138.29

> (df <- prod(dim(haireye) - 1))

[1] 9

> pchisq(chisq, df, lower.tail = FALSE)

[1] 2.3253e-25
```

- These results are of course identical to what `chisq.test()` provides. Note that the latter can be used to retrieve the residuals:

```
> chisq.test(haireye)


Pearson's Chi-squared test

data:  haireye
X-squared = 138, df = 9, p-value <2e-16
```

```
> round(residuals(chisq.test(haireye)), 2)

       Eye
Hair     Brown  Blue Hazel Green
  Black  4.40 -3.07 -0.48 -1.95
  Brown  1.23 -1.95  1.35 -0.35
  Red   -0.07 -1.73  0.85  2.28
  Blond -5.85  7.05 -2.23  0.61
```

△

5.2.1 Shading levels

A variety of schemes for shading the tiles are available in the strucplot framework (Section 5.3), but the simplest (and default) shading patterns for the tiles are based on the sign and magnitude of the standardized Pearson residuals, using shades of blue for positive residuals and red for negative residuals, and two threshold values for their magnitudes, $|r_{ij}| > 2$ and $|r_{ij}| > 4$.

Because the standardized residuals are approximately unit-normal $N(0, 1)$ values, this corresponds to highlighting cells whose residuals are *individually* significant at approximately the .05 and .0001 level, respectively. Other shading schemes described later provide tests of significance, but the main purpose of highlighting cells is to draw attention to the *pattern* of departures of the data from the assumed model of independence.

5.2.2 Interpretation and reordering

To interpret the association between hair color and eye color, consider the pattern of positive (blue) and negative (red) tiles in the mosaic display. We interpret positive values as showing cells whose observed frequency is substantially greater than would be found under independence; negative values indicate cells that occur less often than under independence.

The interpretation can often be enhanced by reordering the rows or columns of the two-way table so that the residuals have an *opposite corner* pattern of signs. This usually helps us interpret any systematic patterns of association in terms of the ordering of the row and column categories.

In this example, a more direct interpretation can be achieved by reordering the eye colors as shown in Figure 5.4. Note that in this rearrangement both hair colors and eye colors are ordered from dark to light, suggesting an overall interpretation of the association between hair color and eye color.

```
> # re-order eye colors from dark to light
> haireye2 <- as.table(haireye[, c("Brown", "Hazel", "Green", "Blue")])
> mosaic(haireye2, shade = TRUE)
```

In general, the levels of a factor in mosaic displays are often best reordered by arranging them according to their scores on the first (largest) *correspondence analysis* dimension (Friendly, 1994b); see Chapter 6 for details. Friendly and Kwan (2003) use this as one example of ***effect ordering*** for data displays, illustrated in Chapter 1.

Thus, the mosaic in Figure 5.4 shows that the association between hair and eye color is essentially that:

- people with dark hair tend to have dark eyes,
- those with light hair tend to have light eyes,
- people with red hair and green eyes do not quite fit this pattern.

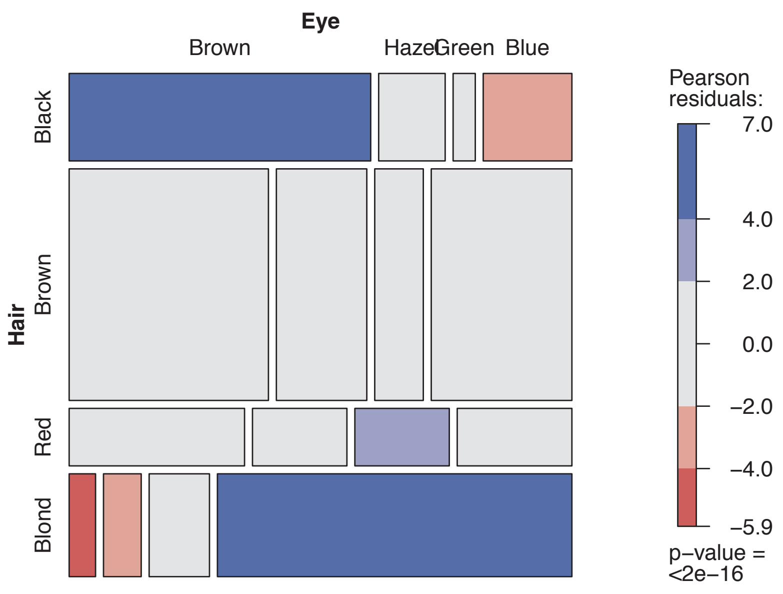


Figure 5.4: Two-way mosaic for hair color and eye color, reordered. The eye colors were reordered from dark to light, enhancing the interpretation.

5.3 The strucplot framework

Mosaic displays have much in common with sieve plots and association plots described in Chapter 4 and with related graphical methods such as ***doubledecker plots*** described later in this chapter. The main idea is to visualize a contingency table of frequencies by "tiles" corresponding to the table cells arranged in rectangular form. For multiway tables with more than two factors, the variables are nested into rows and columns using recursive conditional splits, given the table margins. The result is a "flat" representation that can be visualized in ways similar to a two-dimensional representation of a table. The `structable()` function described in Section 2.5 gives the tabular version of a strucplot. The description below follows Meyer et al. (2006), also included as a vignette (accessible from R as `vignette("strucplot", pkg = "vcd")`), in vcd.

Rather than implementing each of these methods separately, the ***strucplot framework*** in the vcd package provides a general class of methods of which these are all instances. This framework defines a class of conditional displays, which allows for granular control of graphical appearance aspects, including:

- the content of the tiles, e.g., observed or expected frequencies
- the split direction for each dimension, horizontal or vertical
- the graphical parameters of the tiles' content, e.g., color or other visual attributes
- the spacing between the tiles
- the labeling of the tiles

5.3.1 Components overview

The strucplot framework is highly modularized: Figure 5.5 shows the hierarchical relationship between the various components. For the most part, you will directly use the convenience and related

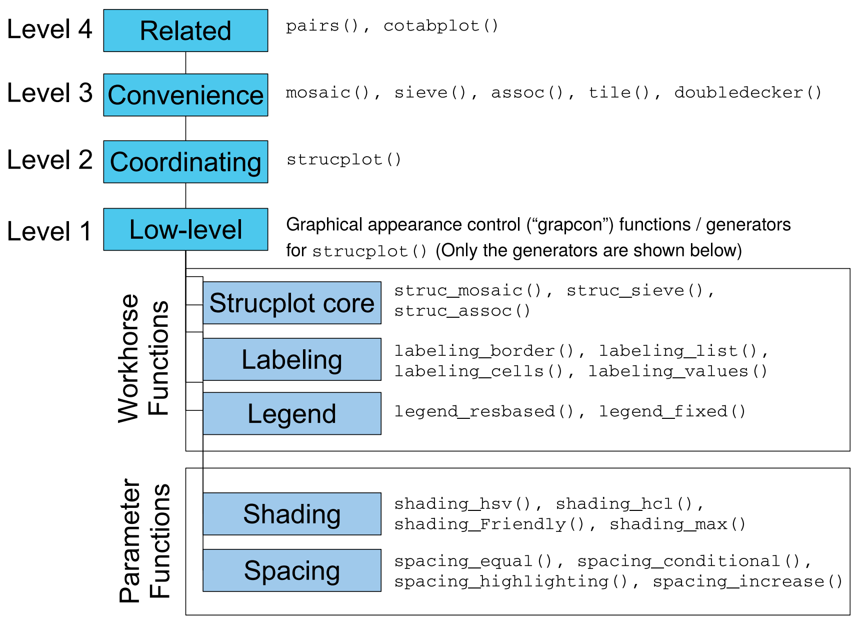

Level 4
Related
pairs(), cotabplot()
Level 3
Convenience
mosaic(), sieve(), assoc(), tile(), doubledecker()
Level 2
Coordinating
strucplot()
Level 1
Low-level
Graphical appearance control ("grapcon") functions / generators for strucplot() (Only the generators are shown below)
Workhorse Functions
Strucplot core
struc_mosaic(), struc_sieve(), struc_assoc()
Labeling
labeling_border(), labeling_list(), labeling_cells(), labeling_values()
Legend
legend_resbased(), legend_fixed()
Parameter Functions
Shading
shading_hsv(), shading_hcl(), shading_Friendly(), shading_max()
Spacing
spacing_equal(), spacing_conditional(), spacing_highlighting(), spacing_increase()

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

Table 5.1: Available graphical appearance control (grapcon) generators in the strucplot framework

Group	Grapcon generator	Description
strucplot core	`struc_assoc()`	core function for association plots
	`struc_mosaic()`	core function for mosaic plots (also used for tile plots)
	`struc_sieve()`	core function for sieve plots
labeling	`labeling_border()`	border labels
	`labeling_cboxed()`	centered labels with boxes, all labels clipped, and on top and left border
	`labeling_cells()`	cell labels
	`labeling_conditional()`	border labels for conditioning variables and cell labels for conditioned variables
	`labeling_doubledecker()`	draws labels for doubledecker plot
	`labeling_lboxed()`	left-aligned labels with boxes
	`labeling_left()`	left-aligned border labels
	`labeling_left2()`	left-aligned border labels, all labels on top and left border
	`labeling_list()`	draws a list of labels under the plot
	`labeling_residuals()`	show residuals in cells
	`labeling_value()`	show values (observed, expected) in cells
shading	`shading_binary()`	visualizes the sign of the residuals
	`shading_Friendly()`	implements Friendly shading (based on HSV colors)
	`shading_hcl()`	shading based on HCL colors
	`shading_hsv()`	shading based on HSV colors
	`shading_max()`	shading visualizing the maximum test statistic (based on HCL colors)
	`shading_sieve()`	implements Friendly shading customized for sieve plots (based on HCL colors)
spacing	`spacing_conditional()`	increasing spacing for conditioning variables, equal spacing for conditioned variables
	`spacing_dimequal()`	equal spacing for each dimension
	`spacing_equal()`	equal spacing for all dimensions
	`spacing_highlighting()`	increasing spacing, last dimension set to zero
	`spacing_increase()`	increasing spacing
legend	`legend_fixed()`	creates a fixed number of bins (similar to `mosaicplot()`)
	`legend_resbased()`	suitable for an arbitrary number of bins (also for continuous shadings)

`pairs()` methods for table objects) arranging collections of plots of the strucplot framework into more complex displays (e.g., by means of panel functions).

5.3.2 Shading schemes

Unlike other graphics functions in base R, the strucplot framework allows almost full control over the graphical parameters of all plot elements. In particular, in association plots, mosaic plots, and sieve plots, you can modify the graphical appearance of each tile individually.

Built on top of this functionality, the framework supplies a set of shading functions choosing colors appropriate for the visualization of loglinear models. The tiles' graphical parameters are set using the `gp` argument of the functions of the strucplot framework. This argument basically expects an object of class "gpar" whose components are arrays of the same shape (length and dimensionality) as the data table.

For added generality, however, you can also supply a `grapcon` function that computes such an object given a vector of residuals, or, alternatively, a *generating function* that takes certain arguments and returns such a grapcon function (see Table 5.1). vcd provides several shading functions, including support for both HSV (hue-saturation-value) and HCL (hue-chroma-luminance) colors, and visualization of significance tests.

5.3.2.1 Specifying graphical parameters for strucplot displays

Strucplot displays in vcd are built using the grid graphics package (Murrell, 2011). There are many graphical parameters that can be set using `gp = gpar(...)` in a call to a high-level strucplot function. Among these, the following are often most useful to control the drawing components:

`col`	Color for lines and borders.
`fill`	Color for filling rectangles, polygons, ...
`alpha`	Alpha channel for transparency of fill color.
`lty`	Line type for lines and borders.
`lwd`	Line width for lines and borders.

In addition, a number of parameters control the display of text labels in these displays:

`fontsize`	The size of text (in points)
`cex`	Multiplier applied to fontsize
`fontfamily`	The font family (serif, sans, `mono`, ...)
`fontface`	The font face (**bold**, *italic*, ...)

See `help(gpar)` for a complete list and `help(par)` for further details.

We illustrate this capability below using the hair color and eye color data as reordered in Figure 5.4. The following example produces a ***Marimekko chart***, or a "poor-man's mosaic display" as shown in the left panel of Figure 5.6. This is essentially a divided bar chart where the eye colors within each horizontal bar for the hair color group are all given the same color. In the example, the matrix `fill_colors` is constructed to conform to the `haireye2` table, using color values that approximate the eye colors.[1]

```
> # color by hair color
> fill_colors <- c("brown4", "#acba72", "green", "lightblue")
> (fill_colors_mat <- t(matrix(rep(fill_colors, 4), ncol = 4)))

     [,1]     [,2]      [,3]    [,4]       
[1,] "brown4" "#acba72" "green" "lightblue"
[2,] "brown4" "#acba72" "green" "lightblue"
[3,] "brown4" "#acba72" "green" "lightblue"
[4,] "brown4" "#acba72" "green" "lightblue"

> mosaic(haireye2, gp = gpar(fill = fill_colors_mat, col = 0))
```

Alternatively, we could have used the convenience function `shading_Marimekko()` in vcd:

```
> mosaic(haireye2, gp = shading_Marimekko(haireye2))
```

Note that because the hair colors and eye colors are both ordered, this shows the decreasing prevalence of light hair color amongst those with brown eyes and the increasing prevalence of light hair with blue eyes.

Alternatively, for some purposes,[2] we might like to use color to highlight the pattern of diagonal

[1]Actually, the `fill_colors` vector could be directly used since values are recycled as needed by `mosaic()`.

[2]For example, this would be appropriate for a square table, showing agreement between row and column categories, as in Section 4.7.

cells, and the off-diagonals 1, 2, 3 steps removed. The R function `toeplitz()` returns such a patterned matrix, and we can use this to calculate the `fill_colors` by indexing the result of the `rainbow_hcl()` palette function in colorspace (Ihaka et al., 2015) (generating better colors than `palette()`). The code below produces the right panel in Figure 5.6.

```
> # toeplitz designs
> library(colorspace)
> toeplitz(1 : 4)

     [,1] [,2] [,3] [,4]
[1,]    1    2    3    4
[2,]    2    1    2    3
[3,]    3    2    1    2
[4,]    4    3    2    1

> fill_colors <- rainbow_hcl(8)[1 + toeplitz(1 : 4)]
> mosaic(haireye2, gp = gpar(fill = fill_colors, col = 0))
```

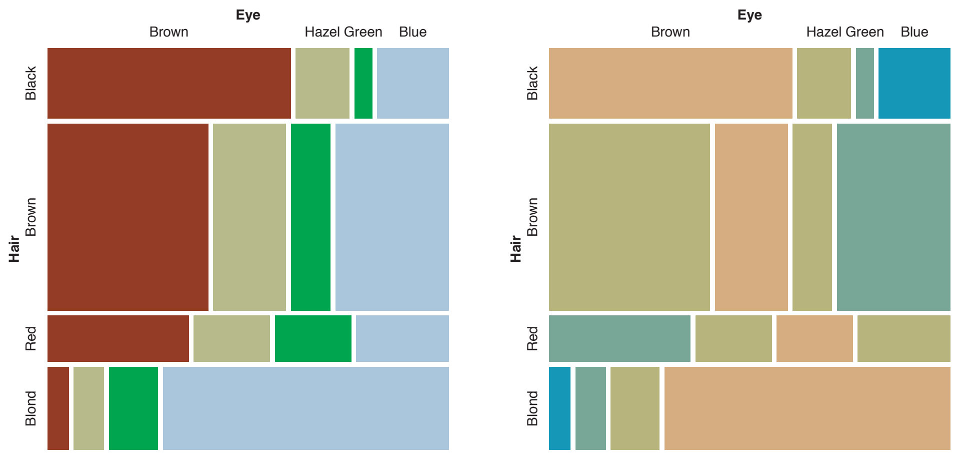


Figure 5.6: Mosaic displays for the `haireye2` data, using custom colors to fill the tiles. Left: Marimekko chart, using colors to reflect the eye colors; right: Toeplitz-based colors, reflecting the diagonal strips in a square table.

Again, vcd offers a convenience function for this purposes, `shading_diagonal()`:

```
> mosaic(haireye2, gp = shading_diagonal(haireye2))
```

More simply, to shade a mosaic according to the levels of one variable (typically a response variable), you can use the `highlighting` arguments of `mosaic()`. The first call below gives a result similar to the left panel of Figure 5.6. Alternatively, using the formula method for `mosaic()`, specify the response variable as the left-hand side.

```
> mosaic(haireye2, highlighting = "Eye", highlighting_fill = fill_colors)
> mosaic(Eye ~ Hair, data = haireye2, highlighting_fill = fill_colors)
```

5.3.2.2 Residual-based shading

The important idea that differentiates mosaic and other strucplot displays from the “poor-man’s,” Marimekko versions (Figure 5.6) often shown in other software, is that rather than just using shading color to *identify* the cells, we can use these attributes to show something more—*residuals* from some model, whose pattern helps to explain the association between the table variables.

As described above, the strucplot framework includes a variety of shading_ functions, and these can be customized with optional arguments. Zeileis et al. (2007) describe a general approach to residual-based shadings for area-proportional visualizations, used in the development of the strucplot framework in vcd.

Example 5.2: Interpolation options

One simple thing to do is to modify the interpolate option passed to the default shading_hcl function, as shown in Figure 5.7.

```
> # more shading levels
> mosaic(haireye2, shade = TRUE, gp_args = list(interpolate = 1 : 4))
>
> # continuous shading
> interp <- function(x) pmin(x / 6, 1)
> mosaic(haireye2, shade = TRUE, gp_args = list(interpolate = interp))
```

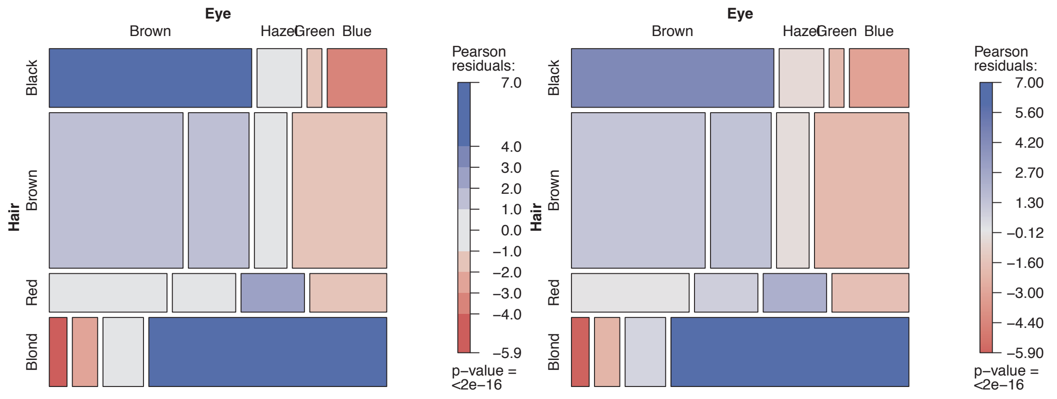


Figure 5.7: Interpolation options for shading levels in mosaic displays. Left: four shading levels; right: continuous shading.

For the left panel of Figure 5.7, a numeric vector is passed as interpolate=1:4, defining the boundaries of a step function mapping the absolute values of residuals to saturation levels in the HCL color scheme. For the right panel, a user-defined function, interp(), is created which maps the absolute residuals to saturation values in a continuous way (up to a maximum of 6).

Note that these two interpolation schemes produce quite similar results, differing mainly in the shading level of residuals within ± 1 and in the legend. In practice, the default discrete interpolation, using cutoffs of $\pm 2, \pm 4$ usually works quite well. △

Example 5.3: Shading functions

Alternatively, the names of shading functions can be passed as the gp argument, as shown below, producing Figure 5.8. Two shading function are illustrated here:

- The left panel of Figure 5.8 uses the classical Friendly (1994b) shading scheme, shading_Friendly with HSV colors[3] of blue and red and default cutoffs for absolute residuals, $\pm 2, \pm 4$, corresponding to `interpolate = c(2, 4)`. In this shading scheme, all tiles use an outline color (`col`) corresponding to the sign of the residual. As well, the border line type (`lty`) distinguishes positive and negative residuals, which is useful if a mosaic plot is printed in black and white.

- The right panel uses the `shading_max()` function, based on the ideas of Zeileis et al. (2007) on residual-based shadings for area-proportional visualizations. Instead of using the cutoffs 2 and 4, it employs the critical values, M_α, for the maximum absolute Pearson residual statistic,

$$M = \max_{i,j} |r_{ij}| \, ,$$

by default at $\alpha = 0.10$ and 0.01.[4] Only those residuals with $|r_{ij}| > M_\alpha$ are colored in the plot, using two levels for Value ("lightness") in HSV color space. Consequently, all color in the plot signals a significant departure from independence at 90% or 99% significance level, respectively.[5]

```
> mosaic(haireye2, gp = shading_Friendly, legend = legend_fixed)
> set.seed(1234)
> mosaic(haireye2, gp = shading_max)
```

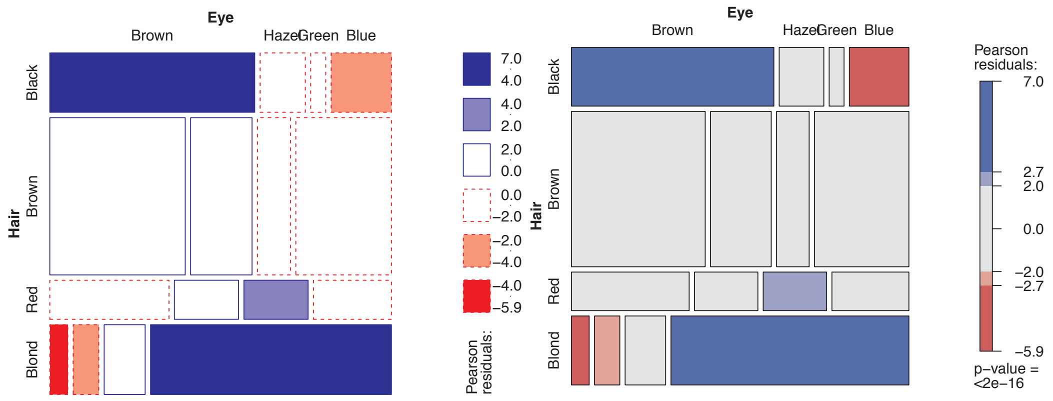


Figure 5.8: Shading functions for mosaic displays. Left: `shading_Friendly` using fixed cutoffs and the "Friendly" color scheme and an alternative legend style (`legend_fixed`); right: `shading_max`, using a permutation-based test to determine significance of residuals.

In this example, the difference between these two shading schemes is largely cosmetic, in that the pattern of association is similar in the two panels of Figure 5.8, and the interpretation would be the same. This is not always the case, as we will see in the next example. △

[3]`shading_Friendly2()` is a variant based on HCL colors.

[4]These default significance levels were chosen because this leads to displays where fully colored cells are clearly significant ($p < 0.01$), cells without color are clearly non-significant ($p > 0.1$), and cells in between can be considered to be weakly significant ($0.01 \le p \le 0.1$).

[5]This computation uses the vcd function `coindep_test()` to calculate generalized tests of (conditional) independence by simulation from the marginal distribution of the input table under (conditional) independence. In these examples using `shading_max`, the function `set.seed()` is used to initialize the random number generators to a given state for reproducibility.

Example 5.4: Arthritis treatment

This example uses the `Arthritis` data, illustrated earlier (Example 2.2), on the relation between treatment and outcome for rheumatoid arthritis. To confine this example to a two-way table, we use only the (larger) female patient group.

```
> art <- xtabs(~ Treatment + Improved, data = Arthritis,
+              subset = Sex == "Female")
> names(dimnames(art))[2] <- "Improvement"
```

The calls to `mosaic()` below compare `shading_Friendly` and `shading_max`, giving the plots shown in Figure 5.9.

```
> mosaic(art, gp = shading_Friendly, margin = c(right = 1),
+        labeling = labeling_residuals, suppress = 0, digits = 2)
> set.seed(1234)
> mosaic(art, gp = shading_max, margin = c(right = 1))
```

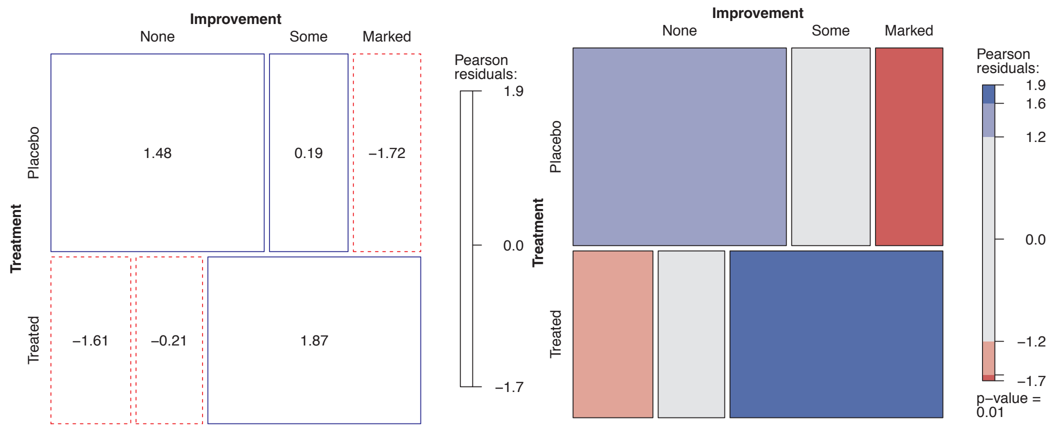


Figure 5.9: Mosaic plots for the female patients in the `Arthritis` data. Left: Fixed shading levels via `shading_Friendly`; right: shading levels determined by significant maximum residuals via `shading_max`.

This data set is somewhat paradoxical, in that the standard `chisq.test()` for association with these data gives a highly significant result, $\chi^2(2) = 11.3, p = 0.0035$, while the shading pattern using `shading_Friendly` in the left panel of Figure 5.9 shows all residuals within ± 2, and thus unshaded.

On the other hand, the `shading_max` shading in the right panel of Figure 5.9 shows that significant deviations from independence occur in the four corner cells, corresponding to more of the treated group showing marked improvement, and more of the placebo group showing no improvement.

Some details behind the `shading_max` method are shown below. The Pearson residuals for this table are calculated as:

```
> residuals(chisq.test(art))

          Improvement
Treatment      None     Some   Marked
  Placebo   1.47752  0.19267 -1.71734
  Treated  -1.60852 -0.20975  1.86960
```

The `shading_max()` function then calls `coindep_test(art)` to generate $n = 1000$ random tables with the same margins, and computes the maximum residual statistic for each. This gives a non-parametric p-value for the test of independence, $p = 0.011$ shown in the legend.

```
> set.seed(1243)
> art_max <- coindep_test(art)
> art_max


Permutation test for conditional independence

data:  art
f(x) = 1.87, p-value = 0.011
```

Finally, the 0.90 and 0.99 quantiles of the simulation distribution are used as shading levels, passed as the value of the `interpolate` argument.

```
> art_max$qdist(c(0.90, 0.99))

   90%    99%
1.2393 1.9167
```

△

The converse situation can also arise in practice. An overall test for association using Pearson's χ^2 may not be significant, but the maximum residual test may highlight one or more cells worthy of greater attention, as illustrated in the following example.

Example 5.5: UK soccer scores

In Example 3.9, we examined the distribution of goals scored by the home team and the away team in 380 games in the 1995/96 season by the 20 teams in the UK Football Association, Premier League. The analysis there focused on the distribution of the total goals scored, under the assumption that the number of goals scored by the home team and the away team were independent.

Here, the rows and columns of the table *UKSoccer* are both ordered, so it is convenient and compact to carry out all the CMH tests taking ordinality into account.

```
> data("UKSoccer", package = "vcd")
> CMHtest(UKSoccer)

Cochran-Mantel-Haenszel Statistics for Home by Away

                 AltHypothesis Chisq Df   Prob
cor         Nonzero correlation  1.01  1 0.315
rmeans   Row mean scores differ  5.63  4 0.229
cmeans   Col mean scores differ  7.42  4 0.115
general     General association 18.65 16 0.287
```

All of these are non-significant, so that might well be the end of the story, as far as independence of goals in home and away games is concerned. Yet, one residual, $r_{42} = 3.08$ stands out, corresponding to 4 or more goals by the home team and only 2 goals by the away team, which accounts for nearly half of the $\chi^2(16) = 18.7$ for general association. The mosaic plot is shown in Figure 5.10.

```
> set.seed(1234)
> mosaic(UKSoccer, gp = shading_max, labeling = labeling_residuals,
+        digits = 2)
```

This occurrence may or may not turn out to have some explanation, but at least the mosaic plot draws it to our attention, and is consistent with the (significant) result from `coindep_test()`.

△

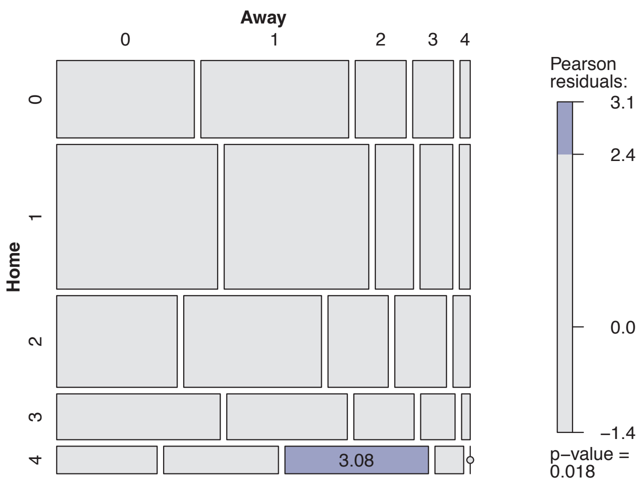


Figure 5.10: Mosaic display for UK soccer scores, highlighting one cell that stands out for further attention.

5.4 Three-way and larger tables

The mosaic displays and other graphical methods within the strucplot framework extend quite naturally to three-way and higher-way tables. The essential idea is that for the variables in a multiway table in a given order, each successive variable is used to subdivide the tile(s) in proportion to the relative (conditional) frequencies of that variable, given all previous variables. This process continues recursively until all table variables have been included.

For simplicity, we continue with the running example of hair color and eye color. Imagine that each cell of the two-way table for hair and eye color is further classified by one or more additional variables—sex and level of education, for example. Then each rectangle can be subdivided horizontally to show the proportion of males and females in that cell, and each of those horizontal portions can be subdivided vertically to show the proportions of people at each educational level in the hair–eye–sex group.

EXAMPLE 5.6: Hair color, eye color, and sex

Figure 5.11 shows the mosaic for the three-way table, with hair and eye color groups divided according to the proportions of males and females. As explained in the next section (Section 5.4.2) there are now different models for "independence" we could investigate, not just the (mutual) independence of all factors. Here, for example, we could examine whether the additional variable (`Sex`) is independent from the *joint* relationship between `Hair` and `Eye`.

```
> HEC <- HairEyeColor[, c("Brown", "Hazel", "Green", "Blue"),]
> mosaic(HEC, rot_labels = c(right = -45))
```

In Figure 5.11 it is easy to see that there is no systematic association between sex and the combinations of hair and eye color—the proportion of male/female students is roughly the same in almost all hair/eye color groups. Yet, among blue-eyed blonds, there seems to be an overabundance of females, and the proportion of blue-eyed males with brown hair also looks suspicious. △

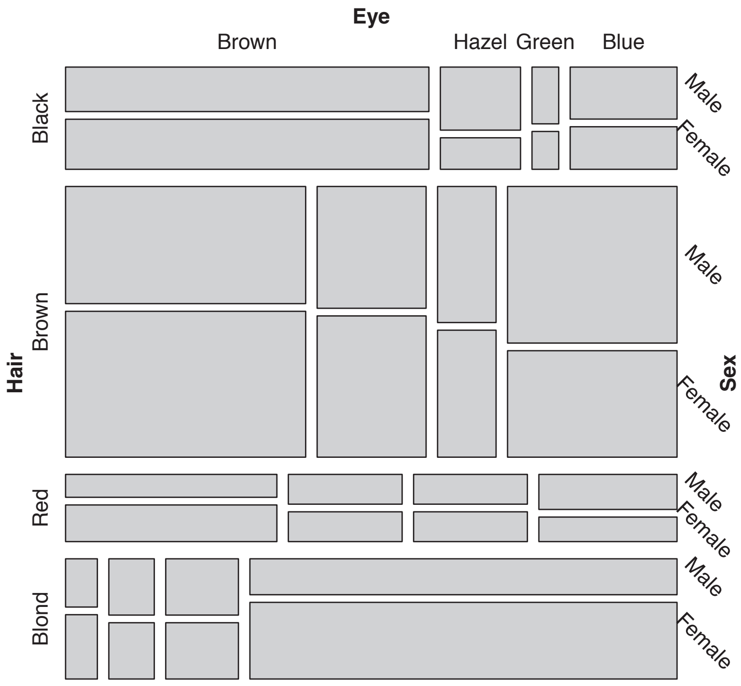


Figure 5.11: Three-way mosaic for hair color, eye color, and sex.

These and other hypotheses are best tested within the framework of ***loglinear model***s, allowing you to flexibly specify various independence models for any number of variables, and analyze them similarly to classical ANOVA models. This general topic is discussed in detail in Chapter 9. For the present purposes, we give a short introduction in the following section.

5.4.1 A primer on loglinear models

The essential idea behind loglinear models is that the multiplicative relationships among expected frequencies under independence (shown as areas in sieve diagrams and mosaic plots) become *additive* models when expressed as models for log frequency, and we briefly explain this connection here for two-way tables.

To see this, consider two discrete variables, A and B, with n_{ij} observations in each cell i, j of an $R \times C$ contingency table, and use $n_{i+} = \Sigma_j n_{ij}$ and $n_{+j} = \Sigma_i n_{ij}$ for the row and column marginal totals, respectively. The total frequency is $n_{++} = \Sigma_{ij} n_{ij}$. Analogously, we use m_{ij} for the expected frequency under any model and also use a subscript $+$ to represent summation over that dimension.

Then, the hypothesis of independence means that the expected frequencies, m_{ij}, obey

$$m_{ij} = \frac{m_{i+}\, m_{+j}}{m_{++}} \; .$$

This multiplicative model can be transformed to an additive (linear) model by taking logarithms of both sides:

$$\log(m_{ij}) = \log(m_{i+}) + \log(m_{+j}) - \log(m_{++}) \; .$$

This is usually re-expressed in an equivalent form in terms of model parameters μ, λ_i^A and λ_j^B

$$\log(m_{ij}) = \mu + \lambda_i^A + \lambda_j^B \equiv [A][B] \,. \tag{5.1}$$

Model Eqn. (5.1) asserts that the row and column variables are independent because there is no term that depends on both A and B.

In contrast, a model for a two-way table that allows an arbitrary association between the variables is the ***saturated model***, including an additional term, λ_{ij}^{AB}:

$$\log(m_{ij}) = \mu + \lambda_i^A + \lambda_j^B + \lambda_{ij}^{AB} \equiv [AB] \,. \tag{5.2}$$

Except for the difference in notation, model Eqn. (5.2) is formally the same as a two-factor ANOVA model with an interaction, typically expressed as $E(y_{ij}) = \mu+\alpha_i+\beta_j+(\alpha\beta)_{ij}$. Hence, associations between variables in loglinear models are analogous to interactions in ANOVA models.[6] In contrast to ANOVA, the "main effects," λ_i^A and λ_j^B, are rarely of interest—a typical log-linear analysis focuses only on the interaction (association) terms.

Models such as Eqn. (5.1) and Eqn. (5.2) are examples of ***hierarchical models***. This means that the model must contain all lower-order terms contained within any high-order term in the model.

Thus, the saturated model, Eqn. (5.2) contains λ_{ij}^{AB}, and therefore *must* contain λ_i^A and λ_j^B. As a result, hierarchical models may be identified by the shorthand notation that lists only the high-order terms: model Eqn. (5.2) is denoted $[AB]$, while model Eqn. (5.1) is $[A][B]$.

In R, the most basic function for fitting loglinear models is `loglin()` in the stats package. It is designed to work with the frequency data in table form, and a model specified in terms of the (high-order) table margins to be fitted. For example, the independence model Eqn. (5.1) is specified as

```
> loglin(mytable, margin = list(1, 2))
```

meaning that variables 1 and 2 are independent, whereas the saturated model Eqn. (5.2) would be specified as

```
> loglin(mytable, margin = list(c(1, 2)))
```

The function `loglm()` in MASS provides a more convenient front-end to `loglin()` to allow loglinear models to be specified using a model formula. With table variables `A` and `B`, the independence model can be fit using `loglm()` as

```
> loglm(~ A + B, data = mytable)
```

and the saturated model in either of the following equivalent forms:

```
> loglm(~ A + B + A : B, data = mytable)
> loglm(~ A * B, data = mytable)
```

In such model formulas, `A:B` indicates an interaction term λ_{ij}^{AB}, while `A*B` is expanded to also include the terms `A + B`.

Example 5.7: Hair color, eye color, and sex

Getting back to our running example of hair and eye color, we start casting the classical test of independence used in Section 5.2 as log-linear model analysis. Using the `haireye` two-way table, the independence of `Hair` and `Eye` is equivalent to the model [Hair][Eye] and formulated in R using `loglm()` as:

[6]The use of superscripted symbols, $\lambda_i^A, \lambda_j^B, \lambda_{ij}^{AB}$, rather than separate Greek letters is a convention in loglinear models, and useful mainly for multiway tables.

```
> 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`, giving Figure 5.12. 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.

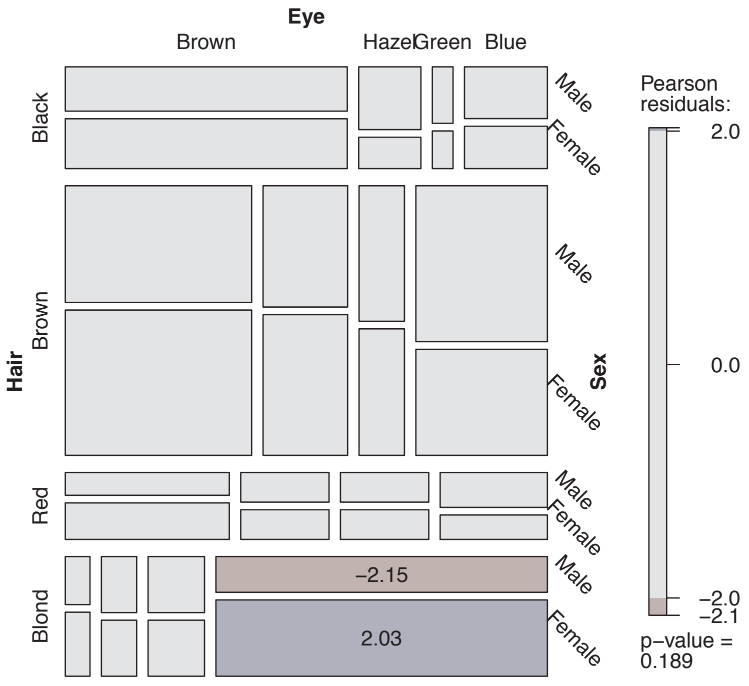


Figure 5.12: Three-way mosaic for hair color, eye color, and sex. Residuals from the model of joint independence, [HE][S] are shown by shading.

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 [•] 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 $m_{i+k} = n_{i+k}$.

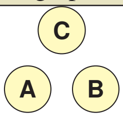
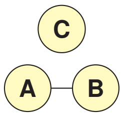
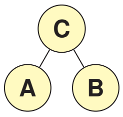
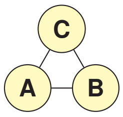
In this table, $A \perp B$ is read, "A is independent of B." The independence interpretation of the model Eqn. (5.3) is $B \perp C \mid A$, which can be read as "B is independent of C, given (conditional on) A." Table 5.2 also depicts the relations among variables as an ***association graph***, where 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")
```

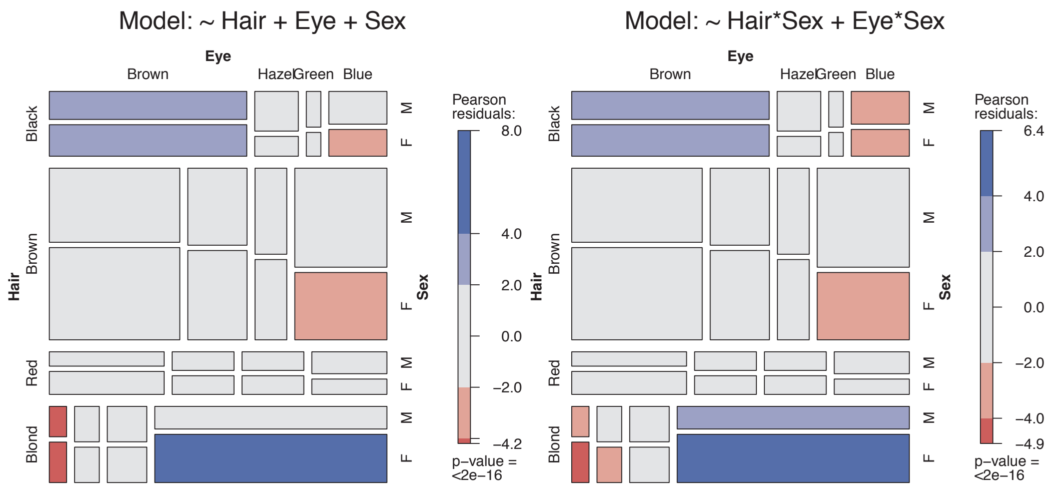


Figure 5.13: Mosaic displays for other models fit to the data on hair color, eye color, and sex. Left: Mutual independence model; right: Conditional independence of hair color and eye color given sex.

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.

We consider loglinear models in more detail in Chapter 9, but for now note that these models are fit using `loglm()` in the MASS package, with the model formula given in the `expected` argument. The details of these models can be seen by fitting these models explicitly, and the fit of several models can be summarized compactly using `LRstats()` in vcdExtra.

```
> library(MASS)
> # three types of independence:
> mod1 <- loglm(~ Hair + Eye + Sex, data = HEC)        # mutual
> mod2 <- loglm(~ Hair * Sex + Eye * Sex, data = HEC) # conditional
> mod3 <- loglm(~ Hair * Eye + Sex, data = HEC)        # joint
> LRstats(mod1, mod2, mod3)

Likelihood summary table:
     AIC BIC LR Chisq Df Pr(>Chisq)
mod1 321 333    166.3 24     <2e-16 ***
mod2 324 344    156.7 18     <2e-16 ***
mod3 193 218     19.9 15       0.18
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Alternatively, you can get the Pearson and likelihood ratio (LR) tests for a given model using `anova()`, or compare a set of models using LR tests on the *difference* in LR χ^2 from one model to the next, when a list of models is supplied to `anova()`.

```
> anova(mod1)

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

Statistics:
                    X^2 df P(> X^2)
Likelihood Ratio 166.30 24        0
Pearson          164.92 24        0

> anova(mod1, mod2, mod3, test = "chisq")

LR tests for hierarchical log-linear models

Model 1:
 ~Hair + Eye + Sex
Model 2:
 ~Hair * Sex + Eye * Sex
Model 3:
 ~Hair * Eye + Sex

          Deviance df Delta(Dev) Delta(df) P(> Delta(Dev)
Model 1    166.300 24
Model 2    156.678 18     9.6222         6        0.14149
Model 3     19.857 15   136.8213         3        0.00000
Saturated    0.000  0    19.8566        15        0.17750
```

△

5.5 Model and plot collections

This section describes a few special circumstances in which a collection of mosaic plots and related loglinear models can be used in a complementary fashion to understand the nature of associations in three-way and larger tables.

5.5.1 Sequential plots and models

As described in Section 5.2, we can think of the mosaic display for an n-way table as being constructed in stages, with the variables listed in a given order, and the unit tile decomposed recursively as each variable is entered in turn. This process turns out to have the useful property that it provides an additive (hierarchical) decomposition of the total association in a table, in a way analogous to sequential fitting with Type I sum of squares in regression models.

Typically, we just view the mosaic and fit models to the full n-way table, but it is useful to understand the connection with models for the marginal subtables, defined by summing over all variables not yet entered. For example, for a three-way table with variables A, B, C, the marginal subtables $\{A\}$ and $\{AB\}$ are calculated in the process of constructing the three-way mosaic. The $\{A\}$ marginal table can be fit to a model where the categories of variable A are equiprobable as shown in Figure 5.2 (or some other discrete distribution); the independence model can be fit to the $\{AB\}$ subtable as in Figure 5.2, and so forth.

This connection can be seen in the following formula that decomposes the joint cell probability in an n-way table with variables $v_1, v_2, \dots v_n$ as a sequential product of conditional probabilities,

$$p_{ijk\ell\cdots} = \underbrace{\overbrace{p_i \times p_{j|i}}^{\{v_1 v_2\}} \times p_{k|ij}}_{\{v_1 v_2 v_3\}} \times p_{\ell|ijk} \times \cdots \times p_{n|ijk\cdots} \tag{5.4}$$

In Eqn. (5.4), the first term corresponds to the one-way mosaic for v_1, the first two terms to the mosaic for v_1 and v_2, the first three terms to the mosaic for v_1, v_2, and v_2, and so forth.

It can be shown (Friendly, 1994b) that this sequential product of probabilities corresponds to a set of sequential models of *joint independence*, whose likelihood ratio G^2 statistics provide an additive decomposition of the total association, $G^2_{[v_1][v_2]\dots[v_n]}$ for the mutual independence model in the full table:

$$G^2_{[v_1][v_2]\dots[v_n]} = G^2_{[v_1][v_2]} + G^2_{[v_1 v_2][v_3]} + G^2_{[v_1 v_2 v_3][v_4]} + \cdots + G^2_{[v_1 \dots v_{n-1}][v_n]} \,. \tag{5.5}$$

For example, for the hair-eye data, the mosaic displays for the [Hair] [Eye] marginal table (Figure 5.4) and the [HairEye] [Sex] table (Figure 5.12) can be viewed as representing the partition of G^2 shown as a table below:

Model	Model symbol	df	G^2
Marginal	[Hair] [Eye]	9	146.44
Joint	[Hair, Eye] [Sex]	15	19.86
Mutual	[Hair] [Eye] [Sex]	24	166.30

The decomposition in this table reflecting Eqn. (5.5) is shown as a visual equation in Figure 5.14. You can see from the shading how the two sequential submodels contribute to overall association in the model of mutual independence.

Although sequential models of joint independence have the nice additive property illustrated above, other classes of sequential models are possible, and sometimes of substantive interest. The main types of these models are illustrated in Table 5.3 for 3-, 4-, and 5-way tables, with variables A, B, ..., E. In all cases, the natural model for the one-way margin is the equiprobability model, and that for the two-way margin is [A][B].

The vcdExtra package provides a collection of convenience functions that generate the loglinear model formulae symbolically, as indicated in the **function** column. The functions `mutual()`, `joint()`, `conditional()`, `markov()` and so forth simply generate a list of terms suitable for a model formula for `loglin()`. See `help(loglin-utilities)` for further details.

Wrapper functions `loglin2string()` and `loglin2formula()` convert these to character strings or model formulae, respectively, for use with `loglm()` and `mosaic()`-related functions in vcdExtra. Some examples are shown below.

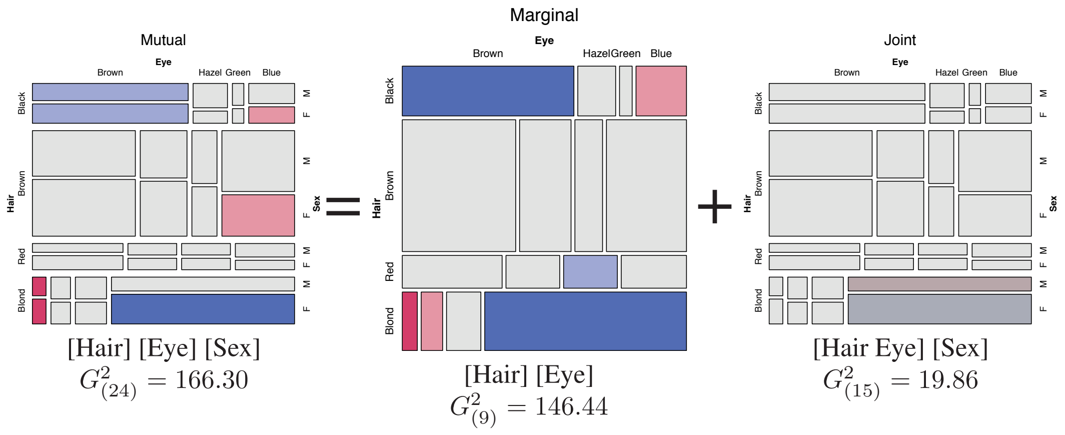


Figure 5.14: Visual representation of the decomposition of the G^2 for mutual independence (total) as the sum of marginal and joint independence.

Table 5.3: Classes of sequential models for n-way tables

function	3-way	4-way	5-way
`mutual`	[A] [B] [C]	[A] [B] [C] [D]	[A] [B] [C] [D] [E]
`joint`	[AB] [C]	[ABC] [D]	[ABCE] [E]
`joint (with=1)`	[A] [BC]	[A] [BCD]	[A] [BCDE]
`conditional`	[AC] [BC]	[AD] [BD] [CD]	[AE] [BE] [CE] [DE]
`conditional (with=1)`	[AB] [AC]	[AB] [AC] [AD]	[AB] [AC] [AD] [AE]
`markov (order=1)`	[AB] [BC]	[AB] [BC] [CD]	[AB] [BC] [CD] [DE]
`markov (order=2)`	[A] [B] [C]	[ABC] [BCD]	[ABC] [BCD] [CDE]
`saturated`	[ABC]	[ABCD]	[ABCDE]

```
> for(nf in 2 : 5) {
+   print(loglin2string(joint(nf, factors = LETTERS[1:5])))
+ }

[1] "[A] [B]"
[1] "[A,B] [C]"
[1] "[A,B,C] [D]"
[1] "[A,B,C,D] [E]"

> for(nf in 2 : 5) {
+   print(loglin2string(conditional(nf, factors = LETTERS[1:5]),
+                       sep = ""))
+ }

[1] "[A] [B]"
[1] "[AC] [BC]"
[1] "[AD] [BD] [CD]"
[1] "[AE] [BE] [CE] [DE]"

> for(nf in 2 : 5) {
+   print(loglin2formula(conditional(nf, factors = LETTERS[1:5])))
+ }

~A + B
~A:C + B:C
~A:D + B:D + C:D
~A:E + B:E + C:E + D:E
```

Applied to data, these functions take a `table` argument, and deliver the string or formula representation of a type of model for that table:

```
> loglin2formula(joint(3, table = HEC))

~Hair:Eye + Sex

> loglin2string(joint(3, table = HEC))

[1] "[Hair,Eye] [Sex]"
```

Their main use, however, is within higher-level functions, such as `seq_loglm()`, which fit the collection of sequential models of a given type.

```
> HEC.mods <- seq_loglm(HEC, type = "joint")
> LRstats(HEC.mods)

Likelihood summary table:
        AIC BIC LR Chisq Df Pr(>Chisq)
model.1 194 194    165.6  3     <2e-16 ***
model.2 241 246    146.4  9     <2e-16 ***
model.3 193 218     19.9 15       0.18
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

In this section we have described a variety of models that can be fit to higher-way tables, some relations among those models, and the aspects of lack of fit that are revealed in the mosaic displays. The following discussion illustrates the process of model fitting, using the mosaic as an interpretive guide to the nature of associations among the variables. In general, we start with a minimal baseline model.[9] The pattern of residuals in the mosaic will suggest associations to be added to an adequate explanatory model. As the model achieves better fit to the data, the degree of shading decreases, so we may think of the process of model fitting as "cleaning the mosaic."

5.5.2 Causal models

The sequence of models of joint independence has another interpretation when the ordering of the variables is based on a set of ordered hypotheses involving causal relationships among variables (Goodman, 1973, Fienberg, 1980, Section 7.2). Suppose, for example, that the causal ordering of four variables is $A \rightarrow B \rightarrow C \rightarrow D$, where the arrow means "is antecedent to." Goodman suggests that the conditional joint probabilities of B, C, and D given A can be characterized by a set of recursive logit models that treat (a) B as a response to A, (b) C as a response to A and B jointly, and (c) D as a response to A, B and C.

These are equivalent to the loglinear models that we fit as the sequential baseline models of joint independence, namely [A][B], [AB][C], and [ABC][D]. The combination of these models with the marginal probabilities of A gives a characterization of the joint probabilities of all four variables, as in Eqn. (5.4). In application, residuals from each submodel show the associations that remain unexplained.

Example 5.9: Marital status and pre- and extramarital sex

A study of divorce patterns in relation to premarital and extramarital sex by Thornes and Collard (1979) reported the 2^4 table shown below, and included in vcd as *PreSex*.

[9]When one variable, R, is a response, this normally is the model of joint independence, $[E_1 E_2 \ldots] [R]$, where $E_1, E_2, \ldots$ are the explanatory variables. Better-fitting models will often include associations of the form $[E_i\, R]$, $[E_i\, E_j\, R] \ldots$.

```
> data("PreSex", package = "vcd")
> structable(Gender + PremaritalSex + ExtramaritalSex ~ MaritalStatus,
+            data = PreSex)

                 Gender          Women              Men
                 PremaritalSex     Yes       No       Yes       No
                 ExtramaritalSex   Yes   No Yes   No Yes   No Yes   No
MaritalStatus
Divorced                            17   54  36  214  28   60  17   68
Married                              4   25   4  322  11   42   4  130
```

These data were analyzed by Agresti (2013, Section 6.1.7) and by Friendly (1994b, 2000), from which this account draws. A sample of about 500 people who had petitioned for divorce, and a similar number of married people, were asked two questions regarding their pre- and extramarital sexual experience: (1) "Before you married your (former) husband/wife, had you ever made love with anyone else?," (2) "During your (former) marriage (did you) have you had any affairs or brief sexual encounters with another man/woman?" The table variables are thus gender (G), reported premarital (P) and extramarital (E) sex, and current marital status (M).

In this analysis we consider the variables in the order G, P, E, and M, and first reorder the table variables for convenience.

```
> PreSex <- aperm(PreSex, 4 : 1)   # order variables G, P, E, M
```

That is, the first stage treats P as a response to G and examines the [Gender][Pre] mosaic to assess whether gender has an effect on premarital sex. The second stage treats E as a response to G and P jointly; the mosaic for [Gender, Pre] [Extra] shows whether extramarital sex is related to either gender or premarital sex. These are shown in Figure 5.15.

```
> # (Gender Pre)
> mosaic(margin.table(PreSex, 1 : 2), shade = TRUE,
+                     main = "Gender and Premarital Sex")
>
> ## (Gender Pre)(Extra)
> mosaic(margin.table(PreSex, 1 : 3),
+        expected = ~ Gender * PremaritalSex + ExtramaritalSex,
+        main = "Gender*Pre + ExtramaritalSex")
```

Finally, the mosaic for [Gender, Pre, Extra] [Marital] is examined for evidence of the dependence of marital status on the three previous variables jointly. As noted above, these models are equivalent to the recursive logit models whose path diagram is $G \rightarrow P \rightarrow E \rightarrow M$.[10] The G^2 values for these models shown below provide a decomposition of the G^2 for the model of complete independence fit to the full table.

Model	df	G^2
[G] [P]	1	75.259
[GP] [E]	3	48.929
[GPE] [M]	7	107.956
[G] [P] [E] [M]	11	232.142

The [Gender] [Pre] mosaic in the left panel of Figure 5.15 shows that men are much more likely to report premarital sex than are women; the sample odds ratio is 3.7. We also see that women are about twice as prevalent as men in this sample. The mosaic for the model of joint independence, [Gender Pre] [Extra] in the right panel of Figure 5.15, shows that extramarital sex depends on gender

[10] Agresti (2013, Figure 6.1) considers a slightly more complex, but more realistic model in which premarital sex affects both the propensity to have extramarital sex and subsequent marital status.

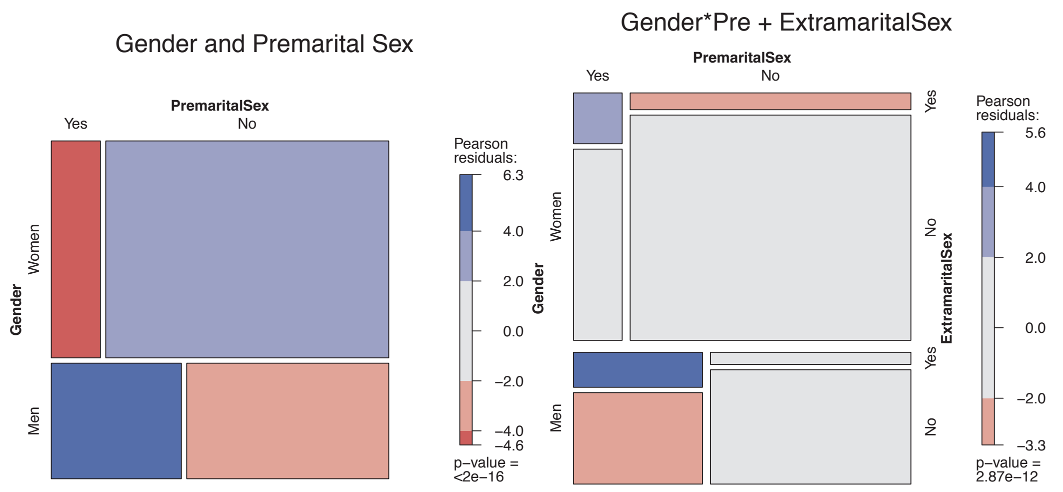


Figure 5.15: Mosaic displays for the first two marginal tables in the PreSex data. Left: Gender and premarital sex; right: fitting the model of joint independence with extramarital sex, [GP][E].

and premarital sex jointly. From the pattern of residuals in Figure 5.15 we see that men and women who have reported premarital sex are far more likely to report extramarital sex than those who have not. In this three-way marginal table, the conditional odds ratio of extramarital sex given premarital sex is nearly the same for both genders (3.61 for men and 3.56 for women). Thus, extramarital sex depends on premarital sex, but not on gender.

```
> loddsratio(margin.table(PreSex, 1 : 3), stratum = 1, log = FALSE)

 odds ratios for Gender and PremaritalSex by ExtramaritalSex

    Yes      No
0.28269 0.28611
```

Four-way mosaic plots for two models are shown in Figure 5.16.

```
> ## (Gender Pre Extra)(Marital)
> mosaic(PreSex,
+        expected = ~ Gender * PremaritalSex * ExtramaritalSex
+                   + MaritalStatus,
+        main = "Gender*Pre*Extra + MaritalStatus")
> ## (GPE)(PEM)
> mosaic(PreSex,
+        expected = ~ Gender * PremaritalSex * ExtramaritalSex
+                   + MaritalStatus * PremaritalSex * ExtramaritalSex,
+        main = "G*P*E + P*E*M")
```

△

5.5.3 Partial association

In a three-way (or larger) table it may be that two variables, say A and B, are associated at some levels of the third variable, C, but not at other levels of C. More generally, we may wish to explore whether and how the association among two (or more) variables in a contingency table varies over

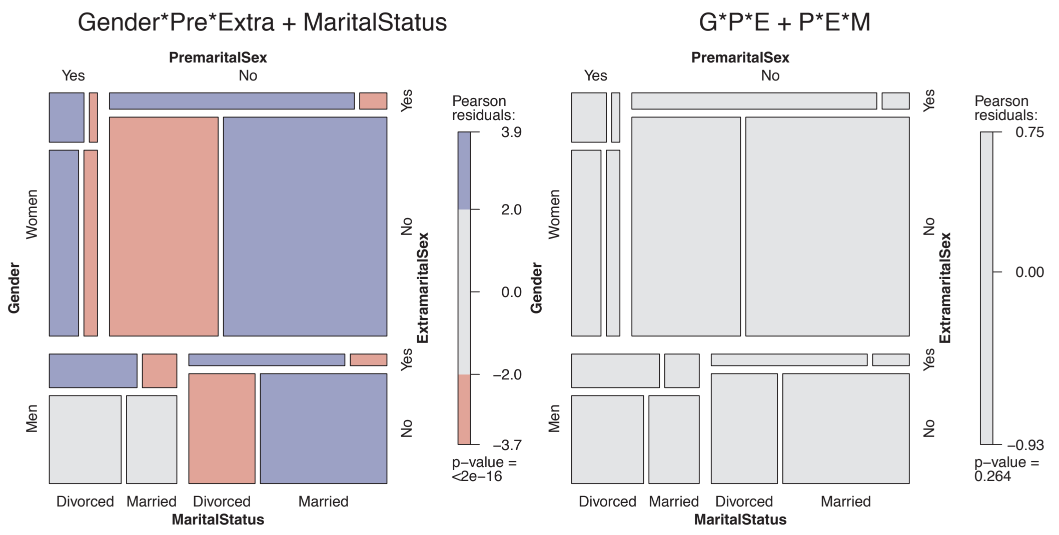


Figure 5.16: Four-way mosaics for the PreSex data. The left panel fits the model [GPE][M]. The pattern of residuals suggests other associations with marital status. The right panel fits the model [GPE][PEM].

the levels of the remaining variables. The term ***partial association*** refers to the association among some variables within the levels of the other variables.

Partial association represents a useful "divide and conquer" statistical strategy: it allows you to refine the question you want to answer for complex relations by breaking it down to smaller, easier questions.[11] It is a statistically happy fact that an answer to the larger, more complex question can be expressed as an algebraic sum of the answers to the smaller questions, just as was the case with sequential models of joint independence.

For concreteness, consider the case where you want to understand the relationship between *attitude* toward corporal punishment of children by parents or teachers (Never, Moderate use OK) and *memory* that the respondent had experienced corporal punishment as a child (Yes, No). But you also have measured other variables on the respondents, including their level of *education* and *age* category. In this case, the question of association among all the table variables may be complex, but we can answer a highly relevant, specialized question precisely, "Is there an association between attitude and memory, *controlling for education and age*?" The answer to this question can be thought of as the sum of the answers to the simpler question of association between attitude and memory across all combinations of the education and age categories.

A simpler version of this idea is considered first below (Example 5.10): among workers who were laid off due to either the closure of a plant or business vs. replacement by another worker, the (conditional) relationship of employment status (new job vs. still unemployed) and duration of unemployment can be studied as a sum of the associations between these focal variables over the separate tables for cause of layoff.

To make this idea precise, consider for example the model of conditional independence, $A \perp B \,|\, C$ for a three-way table. This model asserts that A and B are independent within *each* level of C. Denote the hypothesis that A and B are independent at level $C(k)$ by $A \perp B \,|\, C(k), k = 1, \ldots K$. Then one can show (Andersen, 1991) that

[11]This is an analog, for categorical data, of the ANOVA strategy for "probing interactions" by testing ***simple effects*** at the levels of one or more of the factors involved in a two- or higher-way interaction.

$$G^2_{A \perp B \mid C} = \sum_k^K G^2_{A \perp B \mid C(k)} \,. \tag{5.6}$$

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: $\chi^2_{A \perp B \mid C} = \sum_k^K \chi^2_{A \perp B \mid C(k)}$.

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
```

[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.

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, regardless of length of time they were employed. Workers who were replaced, however, apparently are more likely to be employed, particularly if they were employed for 3 months or more.

```
> # baseline model [A][BC]
> mosaic(Employment, shade = TRUE,
+        expected = ~ EmploymentStatus + EmploymentLength * LayoffCause,
+        main = "EmploymentStatus + Length * Cause")
```

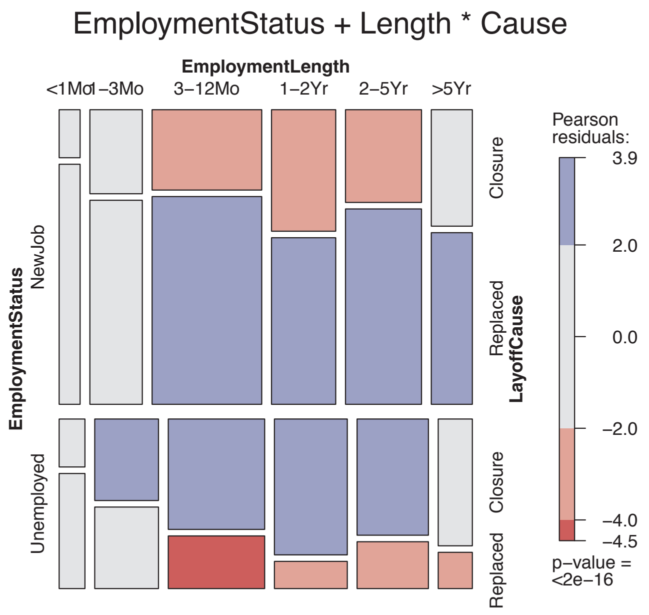


Figure 5.17: Mosaic display for the employment status data, fitting the baseline model of joint independence.

Beyond this baseline model, it is substantively more meaningful to consider the conditional independence model, $A \perp B \mid C$, (or [AC][BC] in shorthand notation), which asserts that employment status is independent of length of employment, given the cause of layoff. We fit this model as shown below:

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

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

Statistics:
                    X^2 df  P(> X^2)
Likelihood Ratio 24.630 10 0.0060927
Pearson          26.072 10 0.0036445
```

This model fits far better ($G^2(10) = 24.63$), but the lack of fit is still significant. The residuals, shown in Figure 5.18, still suggest that the pattern of association between employment and length is different for replaced workers and those laid off due to closure of their workplace.

```
> mosaic(Employment, shade = TRUE, gp_args = list(interpolate = 1 : 4),
+        expected = ~ EmploymentStatus * LayoffCause +
+                     EmploymentLength * LayoffCause,
+        main = "EmploymentStatus * Cause + Length * Cause")
```

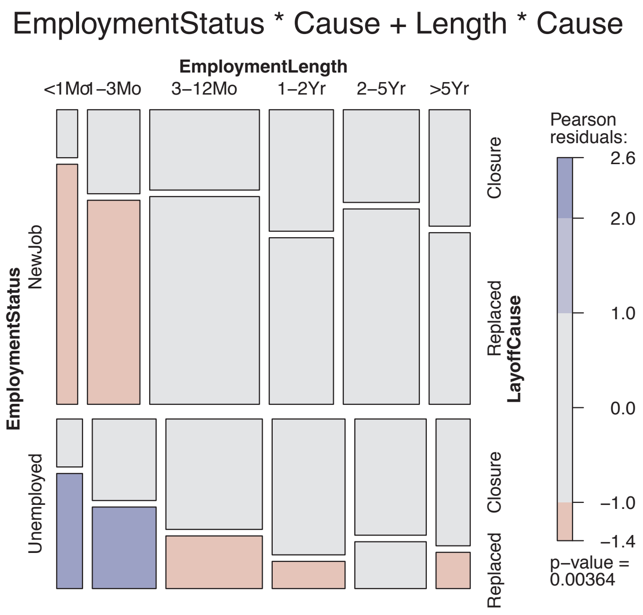


Figure 5.18: Mosaic display for the employment status data, fitting the model of conditional independence, [AC][BC].

To explain this result better, we can fit separate models for the *partial* relationship between EmploymentStatus and EmploymentLength for the two levels of LayoffCause. In R, with the *Employment* data as in table form, this is easily done using apply() over the LayoffCause margin, giving a list containing the two loglm() models.

```
> mods.list <-
+   apply(Employment, "LayoffCause",
+         function(x) loglm(~ EmploymentStatus + EmploymentLength,
+                           data = x))
> mods.list

$Closure
Call:
loglm(formula = ~EmploymentStatus + EmploymentLength, data = x)

Statistics:
                    X^2 df P(> X^2)
Likelihood Ratio 1.4786  5  0.91553
Pearson          1.4835  5  0.91497

$Replaced
Call:
loglm(formula = ~EmploymentStatus + EmploymentLength, data = x)

Statistics:
                    X^2 df   P(> X^2)
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")
```

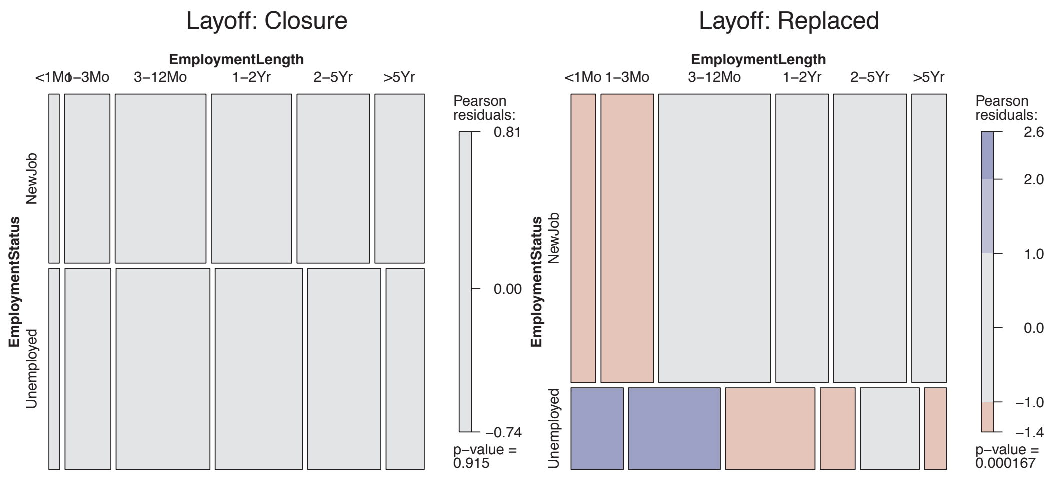


Figure 5.19: Mosaic displays for the employment status data, with separate panels for cause of layoff.

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 $A \perp B \,|\, (C, D)$, or [ACD][BCD] in shorthand notation.

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

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

data:  pun
f(x) = 34.6, p-value <2e-16
```

These tests all show substantial association between attitude and memory of corporal punishment. How can we understand and explain this?

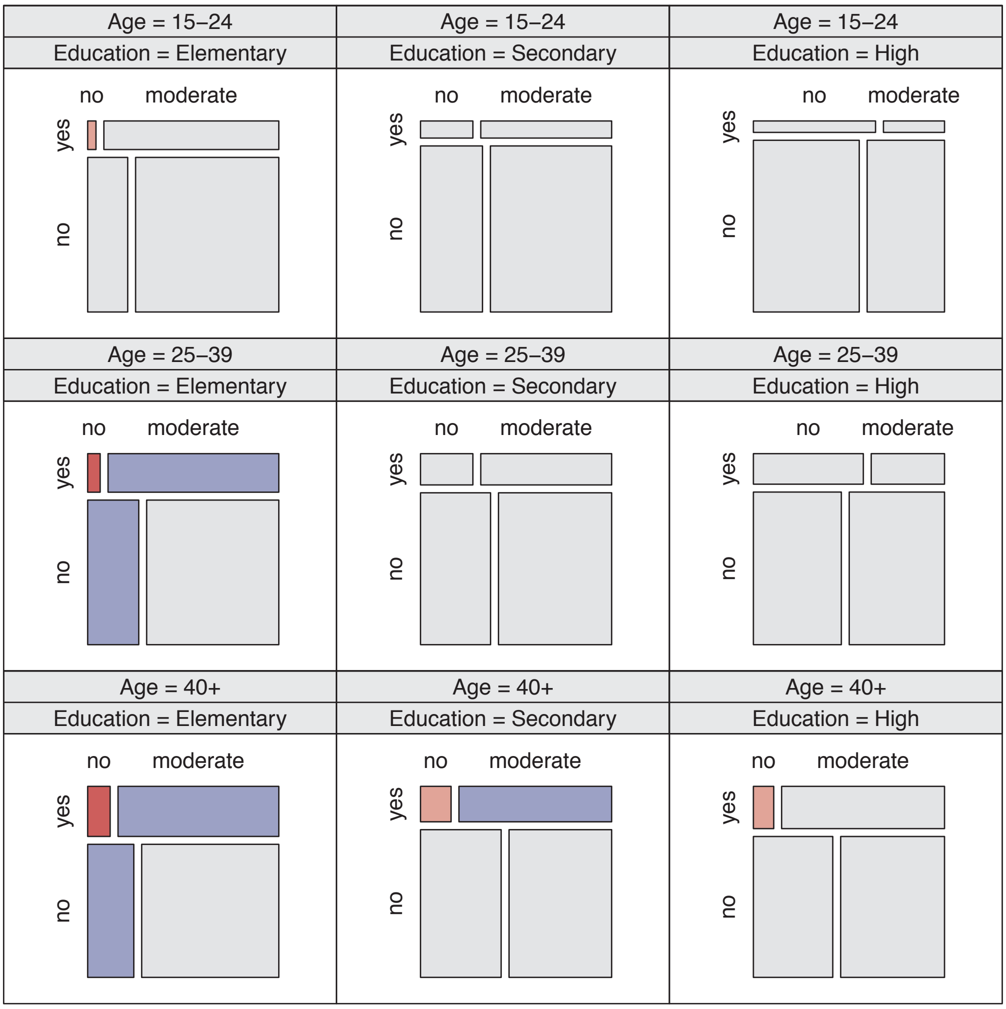


Figure 5.20: Conditional mosaic plot of the Punishment data for the model of conditional independence of attitude and memory, given age and education. Shading of tiles is based on the sum of squares statistic.

As in Example 5.10, we can partition the overall G^2 or χ^2 to show the contributions to this association from the combinations of age and education. The call to `apply()` below fits an independence model for `Memory` and `Attitude` for each stratum defined by the combinations of `Age` and `Education`, and extracts the Pearson χ^2 statistics. The result is returned as a 3×3 matrix.

```
> mods.list <- apply(pun, c("Age", "Education"),
+           function(x) loglm(~ Memory + Attitude, data = x)$pearson)
```

One visual analog of this table of χ^2 statistics is a `cotabplot()` of the (conditional) association of attitude and memory over the age and education cells, shown in Figure 5.20. `cotabplot()` is very general, allowing a variety of functions of the residuals to be used for shading (Zeileis et al., 2007). Here we use the (Pearson) sum of squares statistic, $\sum_{k,\ell} \chi^2_{k,\ell}$.

```
> set.seed(1071)
> pun_cotab <- cotab_coindep(pun, condvars = 3 : 4, type = "mosaic",
+   varnames = FALSE, margins = c(2, 1, 1, 2),
+   test = "sumchisq", interpolate = 1 : 2)
> cotabplot(~ Memory + Attitude | Age + Education,
+             data = pun, panel = pun_cotab)
```

Alternatively, the pattern of conditional association can be shown somewhat more directly in a conditional mosaic plot (Figure 5.21), using the same model formula to condition on age and education. This simply organizes the display to split on the conditioning variables first, with larger spacings.

```
> mosaic(~ Memory + Attitude | Age + Education, data = pun,
+         shade = TRUE, gp_args = list(interpolate = 1 : 4))
```

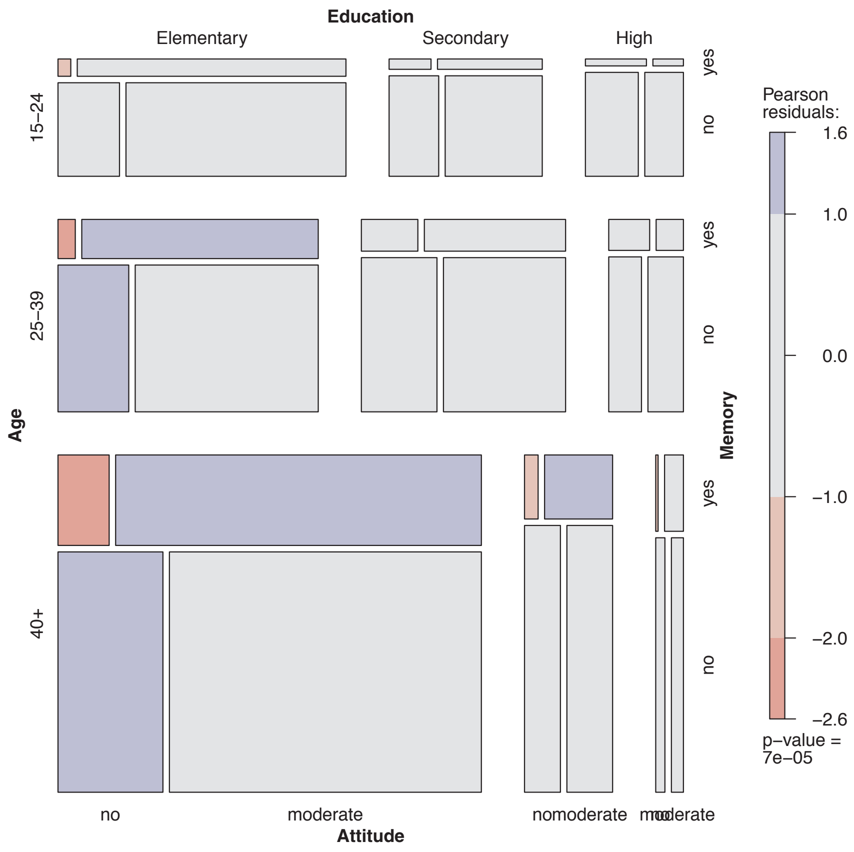


Figure 5.21: Conditional mosaic plot of the Punishment data for the model of conditional independence of attitude and memory, given age and education. This plot explicitly shows the total frequencies in the cells of age and education by the areas of the main blocks for these variables.

Both Figure 5.20 and Figure 5.21 reveal that the association between attitude and memory becomes stronger with increasing age among those with the lowest education (first column). Among those in the highest age group (bottom row), the strength of association *decreases* with increasing education. These two displays differ in that in the `cotabplot()` of Figure 5.20, the marginal frequencies of age and education are not shown, whereas in the `mosaic()` of Figure 5.21 they determine the relative sizes of the tiles for the combinations of age and education.

The divide-and-conquer strategy of partial association using statistical tests and visual displays now provides a simple, coherent explanation for this table: memory of experienced violence as a child tends to engender a more favorable attitude toward corporal punishment as an adult, but this association varies directly with both age and education. △

5.6 Mosaic matrices for categorical data

One reason for the wide usefulness of graphs of quantitative data has been the development of effective, general techniques for dealing with high-dimensional data sets. The ***scatterplot matrix*** shows all pairwise (marginal) views of a set of variables in a coherent display, whose design goal is to show the interdependence among the collection of variables as a whole. It combines multiple views of the data into a single display, which allows detection of patterns that could not readily be discerned from a series of separate graphs. In effect, a multivariate data set in p dimensions (variables) is shown as a collection of $p(p-1)$ two-dimensional scatterplots, each of which is the projection of the cloud of points on two of the variable axes. These ideas can be readily extended to categorical data.

A multiway contingency table of p categorical variables, $A, B, C, \ldots$, contains the interdependence among the collection of variables as a whole. The saturated loglinear model, $[ABC\ldots]$ fits this interdependence perfectly, but is often too complex to describe or understand.

By summing the table over all variables except two, A and B, say, we obtain a two-variable (marginal) table, showing the bivariate relationship between A and B, which is also a projection of the p-variable relation into the space of two (categorical) variables. If we do this for all $p(p-1)$ unordered pairs of categorical variables and display each two-variable table as a mosaic, we have a categorical analog of the scatterplot matrix, called a ***mosaic matrix***. Like the scatterplot matrix, the mosaic matrix can accommodate any number of variables in principle, but in practice is limited by the resolution of our display to three or four variables.

In R, the main implementation of this idea is in the generic function `pairs()`. The vcd package extends this to mosaic matrices with methods for "table" and "structable" objects. The gpairs package provides a ***generalized pairs plot***, with appropriate graphics for a mixture of quantitative and categorical variables.

5.6.1 Mosaic matrices for pairwise associations

EXAMPLE 5.12: Bartlett data on plum-root cuttings

The simplest example of what you can see in a mosaic matrix is provided by the $2 \times 2 \times 2$ table used by Bartlett (1935) to illustrate a method for testing for no three-way interaction in a contingency table (hypothesis H_4 in Table 5.2).

The data set *`Bartlett`* in vcdExtra gives the result of an agricultural experiment to investigate the survival of plum-root cuttings (`Alive`) in relation to two factors: `Time` of planting and the `Length` of the cutting. In this experiment, 240 cuttings were planted for each of the 2×2 combinations of these factors, and their survival (Alive, Dead) was later recorded.

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

The mosaic matrix for these data, showing all two-way marginal relations, is shown in Figure 5.22. It can immediately be seen that `Time` and `Length` are independent by the design of the experiment; we use `gp=shading_Friendly` here to emphasize this.

The top row and left column show the relation of survival to each of time of planting and cutting length. It is easily seen that greater survival is associated with cuttings taken now (vs. spring) and

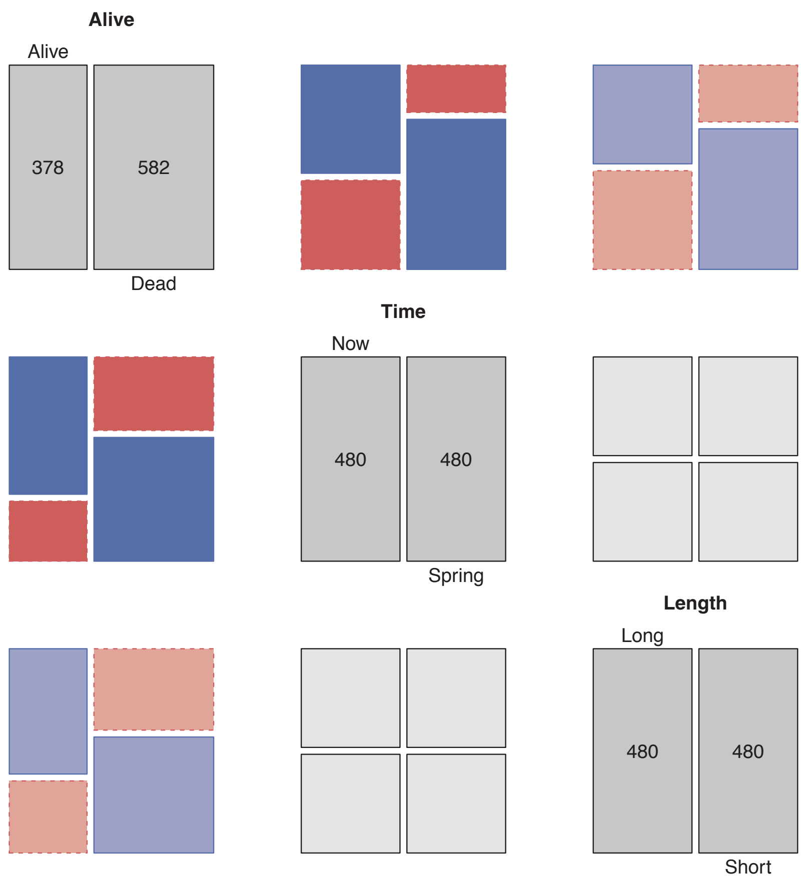


Figure 5.22: Mosaic pairs plot for the Bartlett data. Each panel shows the bivariate marginal relation between the row and column variables.

those cut long (vs. short), and the degree of association is stronger for planting time than for cutting length. △

Example 5.13: Marital status and pre- and extramarital sex

In Example 5.9 we examined a series of models relating marital status to reported premarital and extramarital sexual activity and gender in the *PreSex* data. Figure 5.23 shows the mosaic matrix for these data. The diagonal panels show the labels for the category levels as well as the one-way marginal totals.

```
> data("PreSex", package = "vcd")
> pairs(PreSex, gp = shading_Friendly2, space = 0.25,
+        gp_args = list(interpolate = 1 : 4),
+        diag_panel_args = list(offset_varnames = -0.5))
```

If we view gender, premarital sex, and extramarital sex as explanatory, and marital status (Divorced vs. still Married) as the response, then the mosaics in row 1 (and in column 1)[13] show how marital status depends on each predictor marginally. The remaining panels show the relations within the set of explanatory variables.

[13]Rows and columns in a mosaic matrix are identified as in a table or numerical matrix, with row 1, column 1 in the upper left corner.

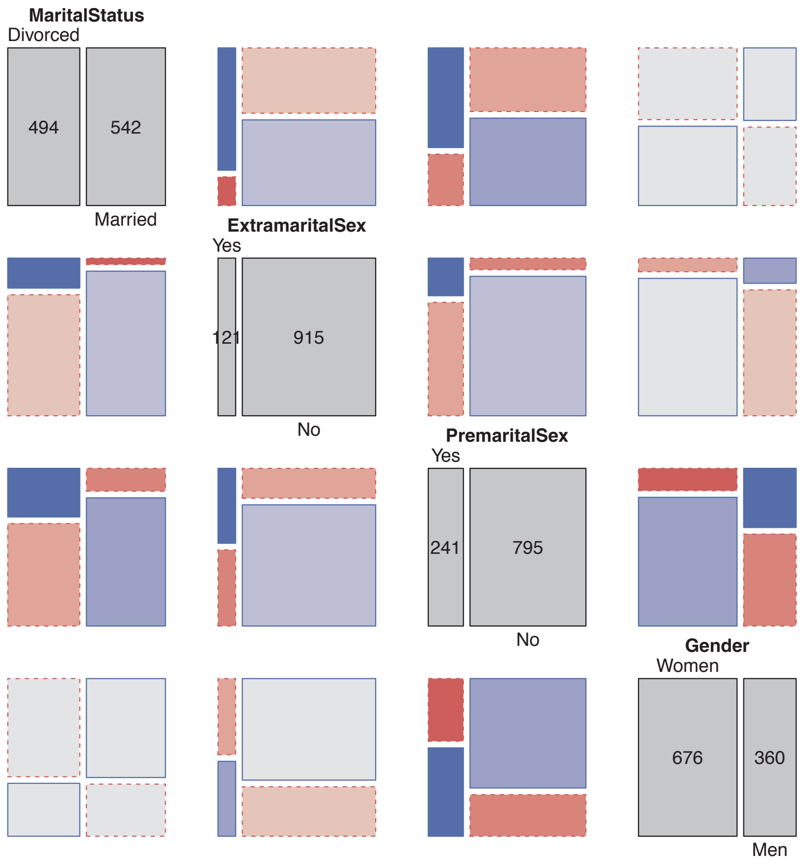


Figure 5.23: Mosaic pairs plot for the PreSex data. Each panel shows the bivariate marginal relation between the row and column variables.

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)
```

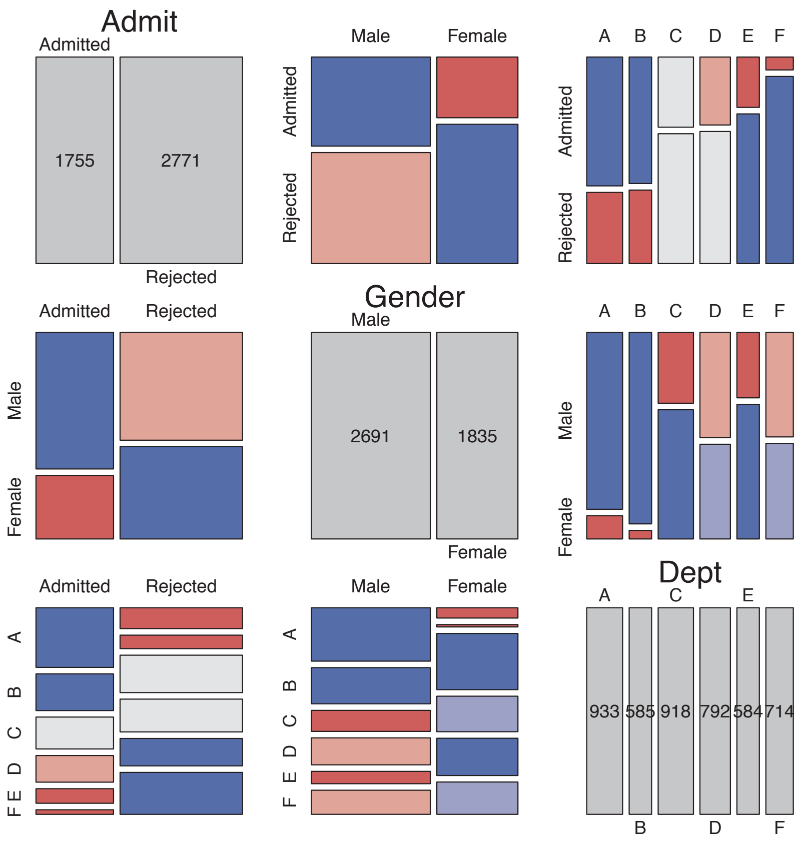


Figure 5.24: Mosaic matrix of the UCBAdmissions data showing bivariate marginal relations.

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 (see Figure 4.6) that, within all but department A, the likelihood of admission is equal for men and women, yet, overall, there appears to be a bias in favor of admitting men (see Figure 4.5). The (1,

[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.

3) and (3, 1) panels show the marginal relation between Admission and Department; that is, how admission rate varies across departments. Departments A and B have the greatest overall admission rate, departments E and F the least. The (2, 3) and (3, 2) panels show how men and women apply differentially to the various departments. It can be seen that men apply in much greater numbers to departments A and B, with higher admission rates, while women apply in greater numbers to the departments C–F, with the lowest overall rate of admission.

△

5.6.2 Generalized mosaic matrices and pairs plots

We need not show only the marginal relation between each pair of variables in a mosaic matrix. Friendly (1999b) describes the extension of this idea to conditional, partial, and other views of a contingency table.

In `pairs.table()`, different ***panel functions*** can be used to specify what is displayed in the upper, lower, and diagonal panels. For the off-diagonal panels, a `type` argument can be used to plot mosaics showing various kinds of independence relations:

`type = "pairwise"` – Shows bivariate marginal relations, collapsed over all other variables.
`type = "total"` – Shows mosaic plots for mutual independence.
`type = "conditional"` – Shows mosaic plots for conditional independence given all other variables.
`type = "joint"` – Shows mosaic plots for joint independence of all pairs of variables from the others.

EXAMPLE 5.15: Berkeley admissions

Figure 5.25 shows the generalized mosaic matrix for the *UCBAdmissions* data, using 3-way mosaics for all the off-diagonal cells. The observed frequencies, of course, are the same in all these cells. However, in the lower panels, the tiles are shaded according to models of joint independence, while in the upper panels, they are shaded according to models of mutual independence.

```
> pairs(UCBAdmissions, space = 0.2,
+        lower_panel = pairs_mosaic(type = "joint"),
+        upper_panel = pairs_mosaic(type = "total"))
```

In this example, it is more useful to fit and display the models of conditional independence for each pair of row, column variables given the remaining one, as shown in Figure 5.26.

```
> pairs(UCBAdmissions, type = "conditional", space = 0.2)
```

Thus, the shading in the (1, 2) and (2, 1) panels shows the fit of the model [Admit, Dept] [Gender, Dept], which asserts that Admission and Gender are independent, given (controlling for) Department. Except for Department A, this model fits quite well, again indicating lack of gender bias. The (1, 3) and (3, 1) panels show the relation between admission and department controlling for gender, highlighting the differential admission rates across departments.

△

Beyond this, the framework of pairs plots can be further generalized to *mixtures* of quantitative and categorical variables, as first described in Friendly (2003) and then in a wider context by Emerson et al. (2013) and Friendly (2013). The essential idea is to consider the combination of two variables, each of which can be either categorical (**C**) or quantitative (**Q**), and various ways to *render* that combination in a graphical display:

CC: mosaic display, sieve diagram, doubledecker plot, faceted or divided bar chart;

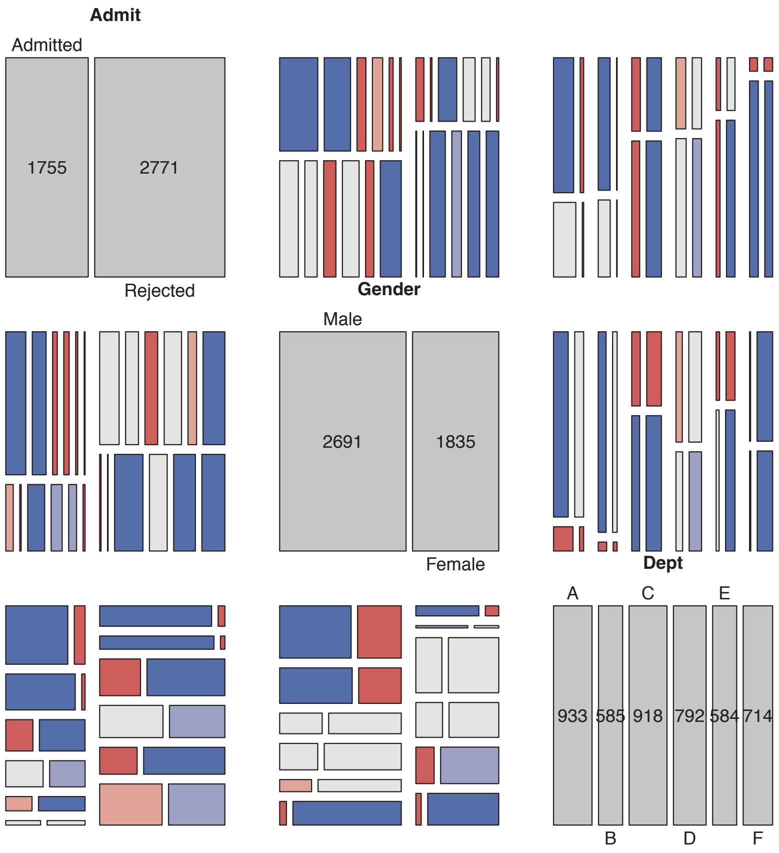


Figure 5.25: Generalized mosaic matrix of the UCBAdmissions data. The above-diagonal plots fit models of joint independence; below-diagonal plots fit models of mutual independence.

CQ: side-by-side boxplots, stripplots, faceted histograms, aligned density plots;
QQ: scatterplot, corrgram, data ellipses, etc.

In R some of these possibilities are provided in the gpairs package (using grid graphics and the vcd strucplot framework), and the GGally (Schloerke et al., 2014) package (an extension to ggplot2).

Example 5.16: Arthritis treatment

We illustrate these ideas with the *Arthritis* data using the gpairs package in Figure 5.27. In this data, the variables `Treatment`, `Sex`, and `Improved` are categorical, and `Age` is quantitative. The call to `gpairs()` below reorders the variables to put the response variable `Improved` in row 1, column 1. Various options can be passed to `mosaic()` using the `mosaic.pars` argument.

```
> library(gpairs)
> data("Arthritis", package = "vcd")
> gpairs(Arthritis[,c(5, 2, 3, 4)],
+        diag.pars = list(fontsize = 20),
+        mosaic.pars = list(gp = shading_Friendly,
+                           gp_args = list(interpolate = 1 : 4)))
```

`gpairs()` provides a variety of options for the **CQ** and **QQ** combinations, as well as the diagonal cells, but only the defaults are used here. The bottom row, corresponding to `Age`, uses boxplots to show the distributions of age for each of the categorical variables. The last column

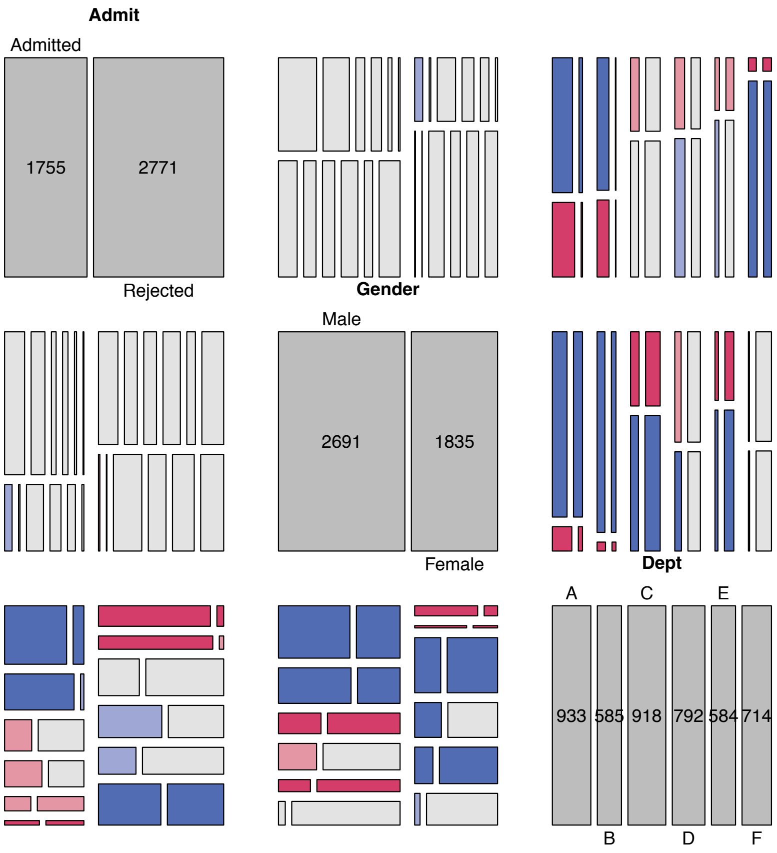


Figure 5.26: Generalized mosaic matrix of the UCBAdmissions data. The off-diagonal plots fit models of conditional indpendence.

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, which mitigates

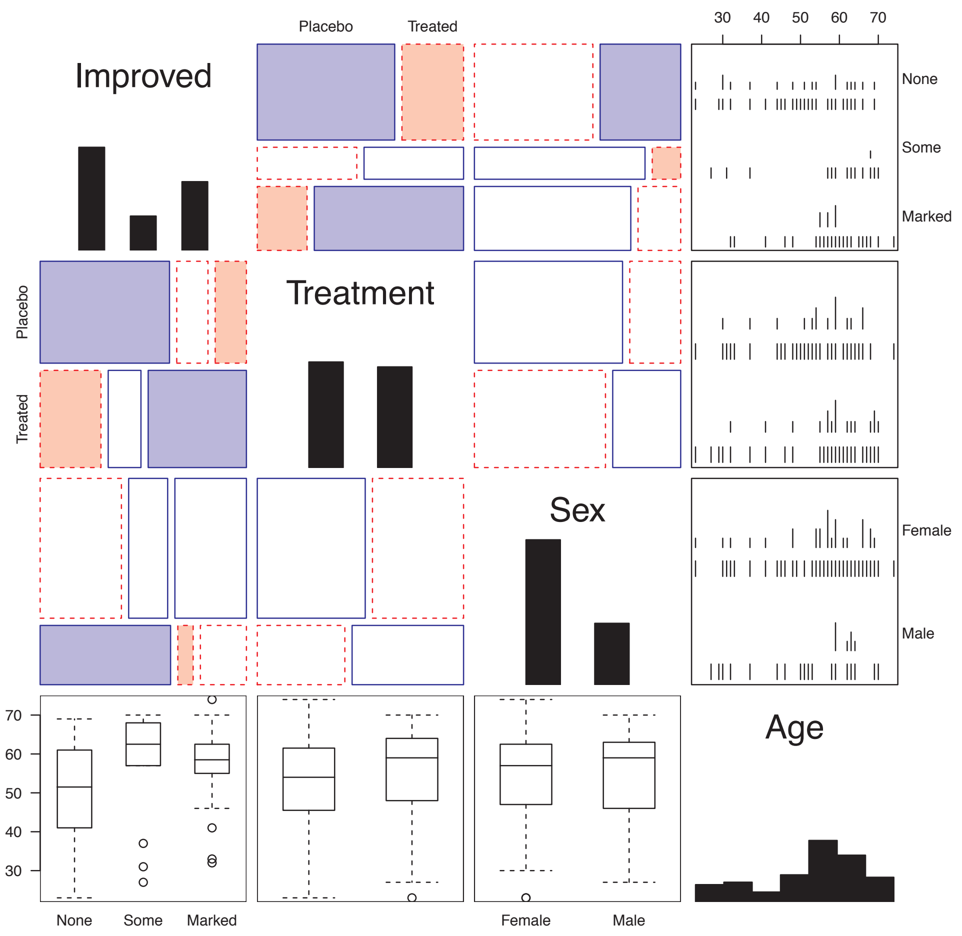


Figure 5.27: Generalized pairs plot of the Arthritis data. Combinations of categorical and quantitative variables can be rendered in various ways.

these problems by: (a) providing an interactive graphic window that can be zoomed and rotated manually with the mouse; (b) allowing dynamic graphics under program control, for example to animate a plot or make a movie; (c) providing control of the details of 3D rendering, including transparency of shapes, surface shading, lighting, and perspective.

The vcdExtra package implements 3D mosaics using rgl graphics. `mosaic3d()` provides methods for "loglm" as well as "table" (or "structable") objects. At the time of writing, only some features of 2D mosaics are available.

EXAMPLE 5.17: Bartlett data on plum-root cuttings

In Example 5.12 we showed the mosaic matrix for the *`Bartlett`*, fitting the model of mutual independence to show all associations among the table variables, `Alive`, `Time` of planting, and `Length` of cutting. Figure 5.28 shows the 3D version, produced using `mosaic3d()`:

```
> mosaic3d(Bartlett)
```

In the view of this figure, it can be seen that cuttings are more likely to be alive when planted Now and when cut Long. These relations can more easily be appreciated by rotating the 3D display.

△

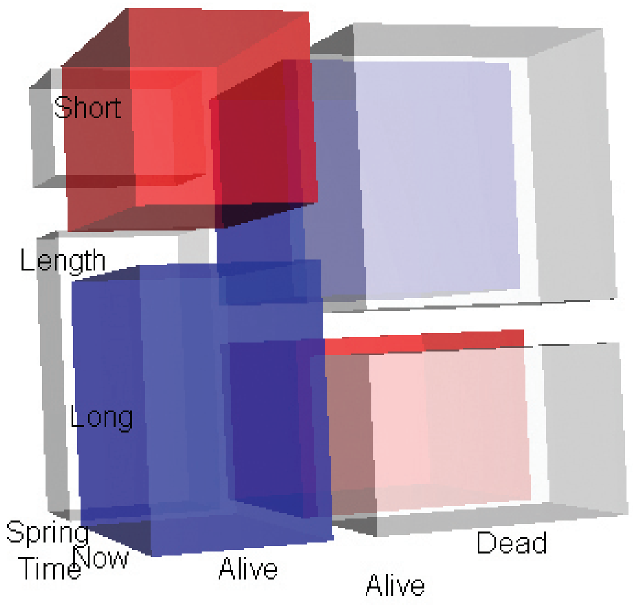


Figure 5.28: 3D mosaic plot of the Bartlett data, according to the model of mutual independence.

5.8 Visualizing the structure of loglinear models

For quantitative response data, it is easy to visualize a fitted model—for linear regression, this is just a plot of the fitted line; for multiple regression or nonlinear regression with two predictors, this is a plot of the fitted response surface. For a categorical response variable, an analog of such plots is provided by effect plots, described later in this book.

For contingency table data, mosaic displays can be used in a similar manner to illuminate the relations among variables in a contingency table represented in various loglinear models, a point described by Theus and Lauer (1999). In fact, each of the model types depicted in Table 5.2 has a characteristic shape and structure in a mosaic display. This, in turn, leads to a clearer understanding of the structure that appears in real data when a given model fits, the relations among the models, and the use of mosaic displays. The essential idea is a simple extension of what we do for more traditional models: show the *expected* (fitted) frequencies under a given model rather than observed frequencies in a mosaic-like display.

To illustrate, we use some artificial data on the relations among age, sex, and symptoms of some disease shown in the $2 \times 2 \times 2$ table `struc` below.

```
> struc <- array(c(6, 10, 312, 44,
+                  37, 31, 192, 76),
+   dim = c(2, 2, 2),
+   dimnames = list(Age = c("Young", "Old"),
+                  Sex = c("F", "M"),
+                  Disease = c("No", "Yes"))
+ )
> struc <- as.table(struc)
> structable(struc)

              Sex   F   M
Age   Disease
Young No            6 312
      Yes          37 192
```

```
Old   No             10  44
      Yes            31  76
```

First, note that there are substantial associations in this table, as shown in Figure 5.29, fitting the (default) mutual independence model.

```
> mosaic(struc, shade = TRUE)
```

The first split by Age shows strong partial associations between Sex and Disease for both young and old. However, the residuals have an opposite pattern for young and old, suggesting a more complex relationship among these variables.

In this section we are asking a different question: what would mosaic displays look like if the data were in accord with simpler models? One way to do this is simply to use the expected frequencies to construct the tiles, as in sieve diagrams. The result, in Figure 5.30, shows that the tiles for sex and disease align for each of the age groups, but it is harder to see the relations among all three variables in this plot.

```
> mosaic(struc, type = "expected")
```

We can visualize the model-implied relations among all variables together more easily using mosaic matrices.

5.8.1 Mutual independence

For example, to show the structure of a table that exactly fits the model of mutual independence, H_1, use the loglm() to find the fitted values, fit, as shown below. The function fitted() extracts these from the "loglm" object.

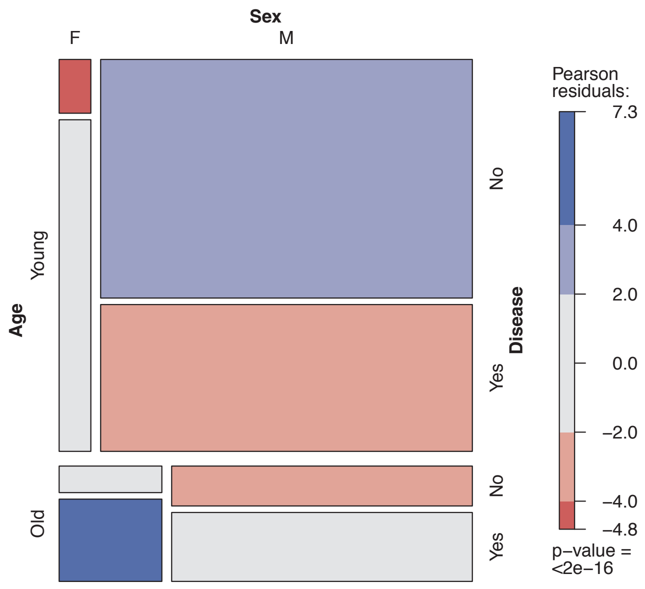


Figure 5.29: Mosaic display for the data on age, sex, and disease. Observed frequencies are shown in the plot, and residuals reflect departure from the model of mutual independence.

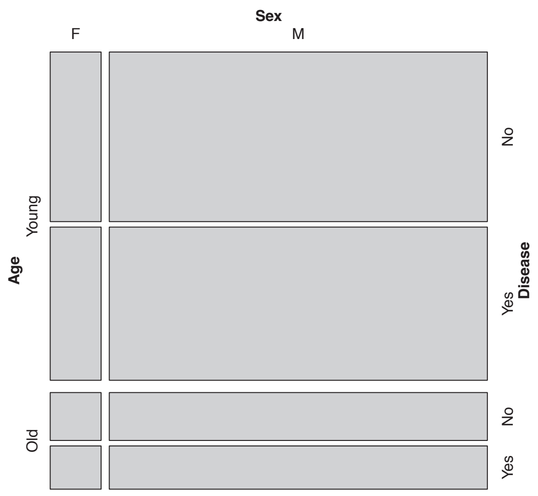


Figure 5.30: Mosaic display for the data on age, sex, and disease, using expected frequencies under mutual independence.

```
> 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 `mosaic3d()` that allows you to rotate the mosaic to see all views. In Figure 5.32, all of the 3D

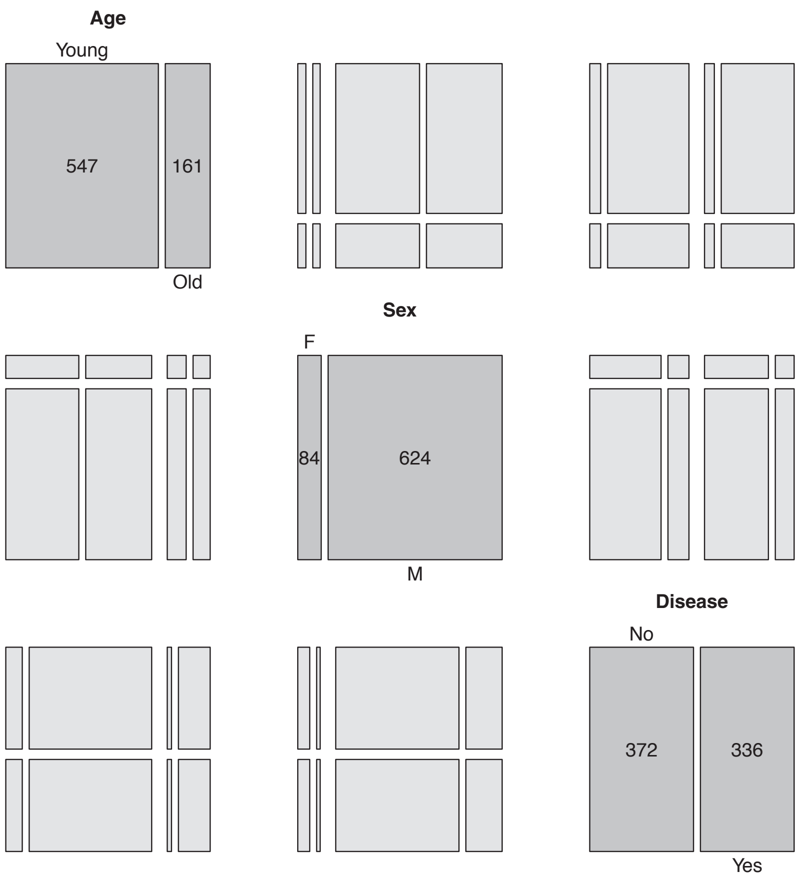


Figure 5.31: Mosaic matrix for fitted values under mutual independence. In all panels the joint frequencies conform to the one-way margins.

tiles are unshaded and you can see that the 3D unit cube has been sliced according to the marginal frequencies.

```
> mosaic3d(fit)
```

5.8.2 Joint independence

The model of joint independence, $H_2 : (A, B) \perp C$, or equivalently, the loglinear model $[AB][C]$ may be visualized similarly by a mosaic matrix in which the data are replaced by fitted values under this model. We illustrate this for the model [Age Sex][Disease], calculating the fitted values in a similar way as before.

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

                Sex        F       M
Age   Disease
Young No               22.593 264.814
      Yes              20.407 239.186
```

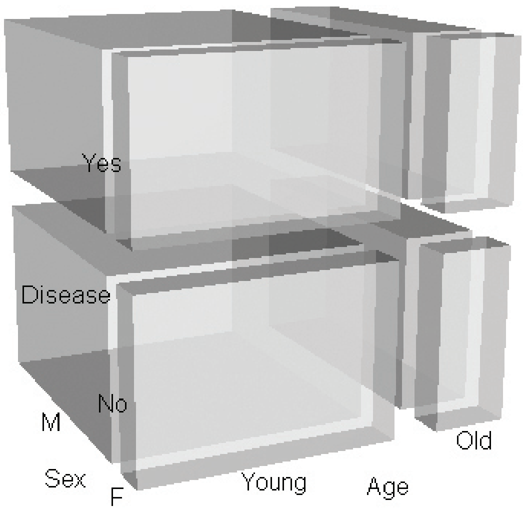


Figure 5.32: 3D mosaic plot of frequencies according to the model of mutual independence. The one-way margins are slices through the unit cube.

```
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`

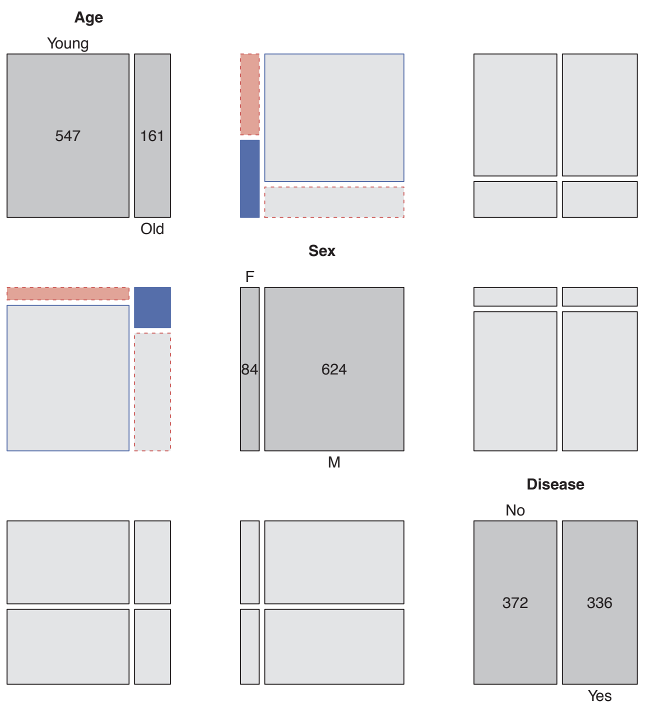


Figure 5.33: Mosaic matrix for fitted values under joint independence for the model [Age Sex][Disease].

`+ ...` where `R` is the response variable and `E1`, `E2`, . . . are the predictors in the contingency table in array form. The shorthand notation, `R ~ .` means that all variables other than `R` are taken as predictors, in their order in the array.

EXAMPLE 5.18: Berkeley admissions

Figure 5.34 shows the doubledecker plot for the *UCBAdmissions* data. By default, the levels of the response (`Admit`) are taken in their order in the array and shaded to highlight the *last* level (Rejected). We want to highlight Admitted, so we reverse this dimension in the call below.

```
> doubledecker(Admit ~ Dept + Gender, data = UCBAdmissions[2:1, , ])
```

In Figure 5.34, it is easy to see the effects of both `Dept` and `Gender` on `Admit`. Admission rate declines across departments A–E, and within departments, the proportion admitted is roughly the same, except for department A, where more female applicants are admitted. △

EXAMPLE 5.19: Titanic data

Figure 5.35 shows the doubledecker plot for the *Titanic* data. The levels of the response (`Survived`) are shaded in increasing grey levels, highlighting the proportions of survival.

```
> doubledecker(Survived ~ Class + Age + Sex, Titanic)
```

This order of variables makes it easiest to compare survival of men and women within each age–class combination, but you can also see that survival of adult women decreases with class, and survival among men was greatest in first class. Some additional visualizations of these relationships are illustrated using the next topic in Example 5.21.

△

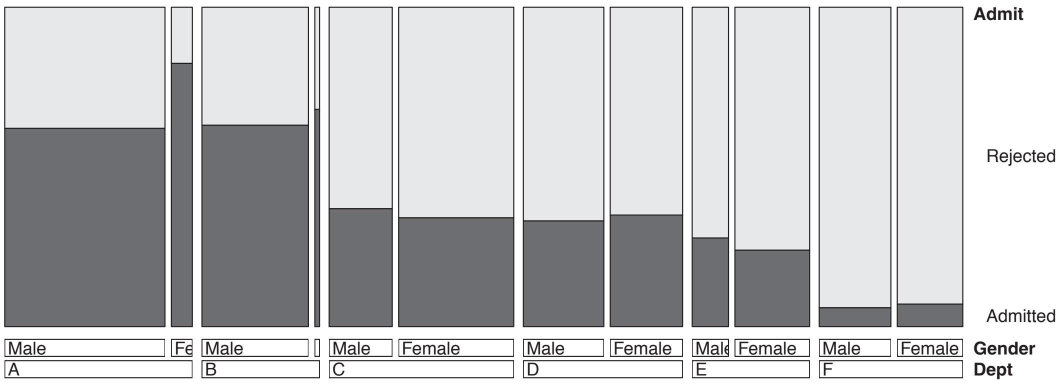


Figure 5.34: Doubledecker plot for the UCBAdmissions data.

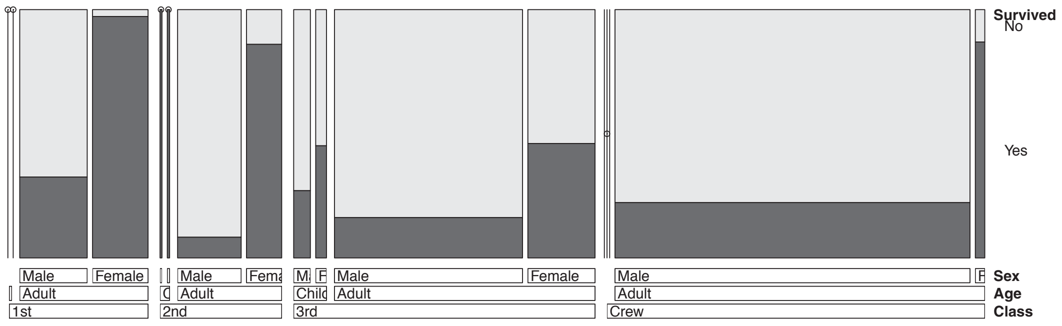


Figure 5.35: Doubledecker plot for the Titanic data.

5.9.2 Generalized odds ratios*

In Example 4.12, we used fourfold displays (Figure 4.7) to analyze the odds ratio between breathlessness and wheeze in coal miners as a function of age. Figure 4.8 showed that a plot of the odds ratio directly against age gave a simplified description of this three-way relationship.

Odds ratios for 2×2 tables can be generalized to $R \times C$ tables in a variety of ways, and these can also be calculated for n-way tables by treating all but the first two dimensions as strata. Plots of these generalized odds ratios can be quite informative, perhaps more so than in the $2 \times 2 \times k$ case.

Consider an $R \times C$ table with frequencies n_{ij}. Then a set of $(R-1) \times (C-1)$ ***local odds ratios***, $\theta_{i,j}$, can be calculated as the odds ratios for adjacent pairs of rows and columns as shown in the left panel of Figure 5.36.

$$\theta_{ij} = \frac{n_{ij}/n_{i+1,j}}{n_{i,j+1}/n_{i+1,j+1}} = \frac{n_{ij} \times n_{i+1,j+1}}{n_{i+1,j} \times n_{i,j+1}}, \quad \begin{array}{l} i = 1, 2, \ldots, R-1 \\ j = 1, 2, \ldots, C-1 \end{array}.$$

These odds ratios correspond to "profile contrasts" (or sequential contrasts or successive differences) for ordered categories. Similarly, if one row category and one column category (say, the last) are considered baseline or reference categories, odds ratios with respect to contrasts with those categories (Figure 5.36, right panel) are defined as

$$\theta_{ij} = \frac{n_{i,j} \times n_{R,C}}{n_{i,C} \times n_{R,j}}, \quad \begin{array}{l} i = 1, 2, \ldots, R-1 \\ j = 1, 2, \ldots, C-1 \end{array}.$$

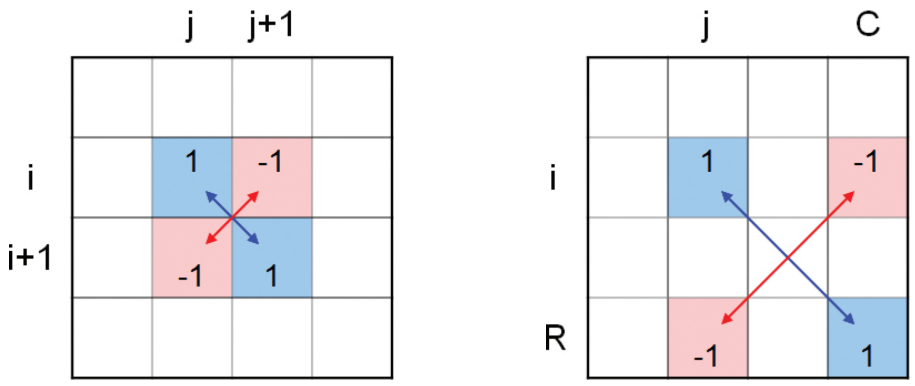


Figure 5.36: Generalized odds ratios for an $R \times C$ table. Left: local odds ratios for adjacent categories. Right: odds ratios with respect to a reference category (the last). Each log odds ratio is a contrast of the log frequencies, shown by the cell weights.

Note that all such parameterizations are equivalent, in that one can derive all other possible odds ratios from any non-redundant set, but substance-driven contrasts will be easier to interpret.

This calculation is simple in terms of log odds ratios, because it corresponds to a contrast among the log frequencies, with weights ± 1 for the four relevant cells. For local odds ratios, these are

$$\log(\theta_{ij}) = \begin{pmatrix} 1 & -1 & -1 & 1 \end{pmatrix} \log \begin{pmatrix} n_{ij} & n_{i+1,j} & n_{i,j+1} & n_{i+1,j+1} \end{pmatrix}^{\mathsf{T}} .$$

Consider an $R \times C \times K_1 \times K_2 \times \ldots$ frequency table $n_{ij\ldots}$, with factors $K_1, K_2 \ldots$ taken as strata. Let $\boldsymbol{n} = \text{vec}(n_{ij\ldots})$ be the $N \times 1$ vectorization of the frequency table. Then, all log odds ratios and their asymptotic covariance matrix can be calculated as:

$$\begin{aligned} \log(\widehat{\boldsymbol{\theta}}) &= \boldsymbol{C} \log(\boldsymbol{n}) \\ \boldsymbol{S} \equiv \mathcal{V}[\log(\boldsymbol{\theta})] &= \boldsymbol{C} \, \text{diag}\,(\boldsymbol{n})^{-1} \, \boldsymbol{C}^{\mathsf{T}} \end{aligned}$$

where $\boldsymbol{C}$ is an N-column matrix containing all zeros, except for two $+1$ elements and two -1 elements in each row that select the four cells involved in each log lodds ratio.[15]

The function `loddsratio()` in vcd calculates these values for the categories of the first two dimensions of an n-way table, together with their asymptotic covariance matrix. Additional dimensions are treated as strata. The `as.array()` and `as.data.frame()` methods can be used to convert a `loddsratio` object to a form suitable for plotting or further analysis.

EXAMPLE 5.20: Corporal punishment data

Example 5.11 used mosaic displays to describe the relationship between attitude toward corporal punishment of children in relationship to memory of having experienced that as a child and education and age of the respondent. Given that `attitude` is the response, we could examine the odds ratios among this variable and any one predictor, treating the other variables as strata. Continuing the analysis of Example 5.11, we calculate log odds ratios for the association of `attitude` and `memory`, stratified by `age` and `education`.

```
> data("Punishment", package = "vcd")
> pun_lor <- loddsratio(Freq ~ memory + attitude | age + education,
+                         data = Punishment)
```

[15]Some additional theory and applications of generalized odds ratios for ordered variables is given by Goodman (1983). Hofmann (2001) describes some connections between odds ratios, loglinear models, and visual modeling using doubledecker plots and mosaic plots.

The `as.data.frame()` method converts this to a data frame, and adds standard errors (ASE).

```
> pun_lor_df <- as.data.frame(pun_lor)
```

The plot method for `loddsratio` objects conveniently plots the log odds ratio (LOR) against the strata variables, `age` or `education`, and by default also adds error bars. The result is shown in Figure 5.37.

```
> plot(pun_lor)
```

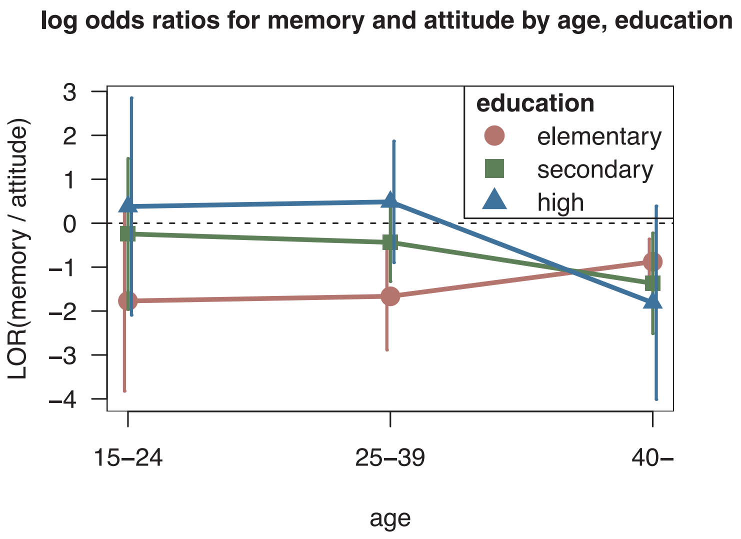


Figure 5.37: Log odds ratio for the association between attitude and memory of corporal punishment, stratified by age and education. Error bars show ± 1 standard error.

Compared to Figure 5.20, the differences among the age and education groups are now clear. For respondents less than age 40, increasing education increases the association (log odds ratio) between attitude and memory: those who remembered corporal punishment as a child are more likely to approve of it as their education increases. This result is reversed for those over 40, where all log odds ratios are negative: memory of corporal punishment makes it *less* likely to approve, and this effect becomes stronger with increased education.

Because log odds ratios have an approximate normal distribution under the null hypothesis that all $\log \theta_{ij} = 0$, you can treat these values as data, and carry out a rough analysis of the effects of the stratifying variables using ANOVA, with weights inversely proportional to the estimated sampling variances.[16] In the analysis shown below, we have treated age and education as ordered (numeric) variables.

```
> pun_mod <- lm(LOR ~ age * education, data = pun_lor_df,
+                weights = 1 / ASE^2)
> anova(pun_mod)
```

[16]This ignores the covariances among the log odds ratios, which are not independent. A proper analysis uses generalized least squares with a weight matrix $\boldsymbol{S}^{-1}$, where $\boldsymbol{S} = \mathcal{V}[\log(\boldsymbol{\theta})]$ is the covariance matrix.

```
Analysis of Variance Table

Response: LOR
              Df Sum Sq Mean Sq F value Pr(>F)
age            1   1.04    1.04    2.72  0.160
education      1   1.84    1.84    4.79  0.080 .
age:education  1   5.04    5.04   13.13  0.015 *
Residuals      5   1.92    0.38
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

This confirms the interaction of age and education on the association between attitude and memory that we described from visual inspection of Figure 5.37. △

EXAMPLE 5.21: Titanic data

For the *Titanic* data, it is useful to examine the odds ratios for survival in relation to age or sex, using the remaining variables as strata. Some preprocessing is nececessary first: These data contain ***structural zeros*** as there were no children in the crew. Accordingly, we set the corresponding cell entries to NA to avoid the calculation of nonsensical values. (Problems of zero frequencies in frequency tables are discussed in more detail in Section 9.5). Additionally, we reverse the order of the levels so that Survived=="Yes" and Age=="Adult" are first. The values calculated below then give the log odds of survival for an adult compared to a child in the combinations sex and class.

```
> Titanic2 <- Titanic[, , 2:1, 2:1]
> Titanic2["Crew", , "Child", ] <- NA
> titanic_lor1 <- loddsratio(~ Survived + Age | Class + Sex,
+                            data = Titanic2)
> titanic_lor1

log odds ratios for Survived and Age by Class, Sex

      Sex
Class      Male    Female
  1st  -3.12102  2.342518
  2nd  -5.50154 -1.510269
  3rd  -0.66874  0.031104
  Crew       NA        NA
```

Similarly, for survival and sex, we obtain the log odds ratios of survival for males versus females, for the combinations of age and class.

```
> titanic_lor2 <- loddsratio(~ Survived + Sex | Class + Age,
+                            data = Titanic2)
> titanic_lor2

log odds ratios for Survived and Sex by Class, Age

      Age
Class    Adult    Child
  1st  -4.1643  1.29928
  2nd  -4.1516 -0.16034
  3rd  -1.4786 -0.77879
  Crew -3.0156       NA
```

The plots for both tables are shown in Figure 5.38.

In the left panel of Figure 5.38 you can see that the odds ratio of survival for adults relative to children was always greater for females as compared to males, but much less so in 3^{rd} class. In the right panel, the odds ratio of survival for males versus females was always greater for children than adults, again less so in 3^{rd} class. △

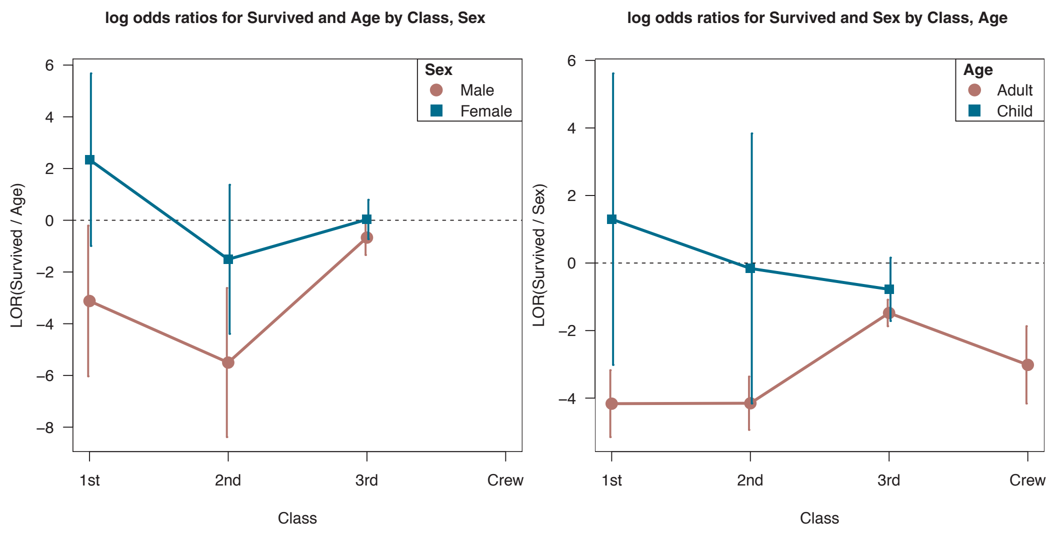


Figure 5.38: Log odds ratio plots for the Titanic data. Left: Odds ratios for survival and age, by sex and class. Right: for survival and sex, by age and class. Error bars show ±1 standard error.

Other examples and plots for log odds ratios are shown in `help(loddsratio)`.

5.10 Chapter summary

- The mosaic display depicts the frequencies in a contingency table by a collection of rectangular "tiles" whose area is proportional to the cell frequency. The residual from a specified model is portrayed by shading the tile to show the sign and magnitude of the deviation from the model.

- For two-way tables, the tiles for the second variable align at each level of the first variable when the two variables are independent (see Figure 5.10).

- The perception and understanding of *patterns of association* (deviations from independence) are enhanced by reordering the rows or columns to give the shading of the residuals a more coherent pattern. An opposite-corner pattern "explains" the association in terms of the ordering of the factor levels.

- For three-way and larger tables, a variety of models can be fit and visualized. Starting with a minimal baseline model, the pattern of residuals will often suggest additional terms that must be added to "clean the mosaic."

- It is often useful to examine the *sequential* mosaic displays for the marginal subtables with the variables in a given order. Sequential models of joint independence provide a breakdown of the total association in the full table, and are particularly appropriate when the last variable is a response.

- Partial association, which refers to the associations among a subset of variables, within the levels of other variables, may be easily studied by constructing separate mosaics for the subset variables for the levels of the other, "given" variables. These displays provide a breakdown of a model of conditional association for the whole table, and serve as an analog of coplots for quantitative data.

- Mosaic matrices, consisting of all pairwise plots of an n-way table, provide a way to visualize all marginal, joint, or conditional relations simultaneously. Doubledecker plots and plots of generalized odds ratios provide other methods to visualize n-way tables.

- The structural relations among model terms in various loglinear models themselves can also be visualized by mosaic matrices showing the expected, rather than observed, frequencies under different models.

- Related visualization techniques include doubledecker plots for binary response models and line plots for generalized odds ratios.

5.11 Lab exercises

Exercise 5.1 The data set *criminal* in the package logmult gives the 4×5 table below of the number of men aged 15–19 charged with a criminal case for whom charges were dropped in Denmark from 1955–1958.

```
> data("criminal", package = "logmult")
> criminal

      Age
Year    15  16  17  18  19
  1955 141 285 320 441 427
  1956 144 292 342 441 396
  1957 196 380 424 462 427
  1958 212 424 399 442 430
```

(a) Use `loglm()` to test whether there is an association between `Year` and `Age`. Is there evidence that dropping of charges in relation to age changed over the years recorded here?
(b) Use `mosaic()` with the option `shade=TRUE` to display the pattern of signs and magnitudes of the residuals. Compare this with the result of `mosaic()` using "Friendly shading," from the option `gp=shading_Friendly`. Describe verbally what you see in each regarding the pattern of association in this table.

Exercise 5.2 The data set *AirCrash* in vcdExtra gives a database of all crashes of commercial airplanes between 1993–2015, classified by `Phase` of the flight and `Cause` of the crash. How can you best show is the nature of the association between these variables in a mosaic plot? Start by making a frequency table, `aircrash.tab`:

```
> data("AirCrash", package = "vcdExtra")
> aircrash.tab <- xtabs(~ Phase + Cause, data = AirCrash)
```

(a) Make a default mosaic display of the data with `shade=TRUE` and interpret the pattern of the high-frequency cells.
(b) The default plot has overlapping labels due to the uneven marginal frequencies relative to the lengths of the category labels. Experiment with some of the `labeling_args` options (`abbreviate`, `rot_labels`, etc.) to see if you can make the plot more readable. *Hint*: a variety of these are illustrated in Section 4.1 of `vignette("strucplot")`
(c) The levels of `Phase` and `Cause` are ordered alphabetically (because they are factors). Experiment with other orderings of the rows/columns to make interpretation clearer, e.g., ordering `Phase` temporally or ordering both factors by their marginal frequency.

Exercise 5.3 The Lahman package contains comprehensive data on baseball statistics for Major League Baseball from 1871 through 2012. For all players, the *Master* table records the handedness of players, in terms of throwing (L, R) and batting (B, L, R), where B indicates "both." The table below was generated using the following code:

```
> library(Lahman)
> data("Master", package = "Lahman")
> basehands <- with(Master, table(throws, bats))
```

	Bats		
Throws	B	L	R
L	177	2640	527
R	924	1962	10442

- Use the code above, or else enter these data into a frequency table in R.
- Construct mosaic displays showing the relation of batting and throwing handedness, split first by batting and then by throwing.
- From these displays, what can be said about players who throw with their left or right hands in terms of their batting handedness?

Exercise 5.4 * A related analysis concerns differences in throwing handedness among baseball players according to the fielding position they play. The following code calculates such a frequency table.

```
> library(Lahman)
> MasterFielding <- data.frame(merge(Master, Fielding, by = "playerID"))
> throwPOS <- with(MasterFielding, table(POS, throws))
```

(a) Make a mosaic display of throwing hand vs. fielding position.
(b) Calculate the percentage of players throwing left-handed by position. Make a sensible graph of this data.
(c) Re-do the mosaic display with the positions sorted by percentage of left-handers.
(d) Is there anything you can say about positions that have very few left-handed players?

Exercise 5.5 For the *Bartlett* data described in Example 5.12, fit the model of no three-way association, H_4 in Table 5.2.

(a) Summarize the goodness of fit for this model, and compare to simpler models that omit one or more of the two-way terms.
(b) Use a mosaic-like display to show the lack of fit for this model.

Exercise 5.6 Red core disease, caused by a fungus, is not something you want if you are a strawberry. The data set *jansen.strawberry* from the agridat package gives a frequency data frame of counts of damage from this fungus from a field experiment reported by Jansen (1990). See the help file for details. The following lines create a $3 \times 4 \times 3$ table of crossings of 3 male parents with 4 (different) female parents, recording the number of plants in four blocks of 9 or 10 plants each showing red core disease in three ordered categories, C1, C2, or C3.

```
> data("jansen.strawberry", package = "agridat")
>
> dat <- jansen.strawberry
> dat <- transform(dat, category = ordered(category,
+                                            levels = c('C1','C2','C3')))
> levels(dat$male) <- paste0("M", 1:3)
> levels(dat$female) <- paste0("F", 1:4)
>
> jansen.tab <- xtabs(count ~ male + female + category, data = dat)
> names(dimnames(jansen.tab)) <- c("Male parent", "Female parent",
+                                  "Disease category")
> ftable(jansen.tab)
```

(a) Use `pairs(jansen.tab, shade=TRUE)` to display the pairwise associations among the three variables. Describe how disease category appears to vary with male and female parent. Why is there no apparent association between male and female parent?

(b) As illustrated in Figure 5.6, use `mosaic()` to prepare a 3-way mosaic plot with the tiles colored in increasing shades of some color according to disease category. Describe the pattern of category C3 in relation to male and female parent. (Hint: the `highlighting` arguments are useful here.)

(c) With `category` as the response variable, the minimal model for association is [MF][C], or `~ 1*2 + 3`. Fit this model using `loglm()` and display the residuals from this model with `mosaic()`. Describe the pattern of lack of fit of this model.

Exercise 5.7 The data set *`caith`* in MASS gives another classic 4×5 table tabulating hair color and eye color, this for people in Caithness, Scotland, originally from Fisher (1940). The data is stored as a data frame of cell frequencies, whose rows are eye colors and whose columns are hair colors.

```
> data("caith", package = "MASS")
> caith

       fair red medium dark black
blue    326  38    241  110     3
light   688 116    584  188     4
medium  343  84    909  412    26
dark     98  48    403  681    85
```

(a) The `loglm()` and `mosaic()` functions don't understand data in this format, so use `Caith <- as.matrix(caith)` to convert to array form. Examine the result, and use `names(dimnames(Caith))<-c()` to assign appropriate names to the row and column dimensions.

(b) Fit the model of independence to the resulting matrix using `loglm()`.

(c) Calculate and display the residuals for this model.

(d) Create a mosaic display for this data.

Exercise 5.8 The *`HairEyePlace`* data in vcdExtra gives similar data on hair color and eye color, for both Caithness and Aberdeen as a $4 \times 5 \times 2$ table.

(a) Prepare separate mosaic displays, one for each of Caithness and Aberdeen. Comment on any difference in the pattern of residuals.

(b) Construct conditional mosaic plots, using the formula `~ Hair + Eye | Place` and both `mosaic()` and `cotabplot()`. It is probably more useful here to suppress the legend in these plots. Comment on the difference in what is shown in the two displays.

Exercise 5.9 Bertin (1983, pp. 30–31) used a 4-way table of frequencies of traffic accident victims in France in 1958 to illustrate his scheme for classifying data sets by numerous variables, each of which could have various types and could be assigned to various visual attributes. His data are contained in *Accident* in vcdExtra, a frequency data frame representing his $5 \times 2 \times 4 \times 2$ table of the variables age, result (died or injured), mode of transportation, and gender.

```
> data("Accident", package = "vcdExtra")
> str(Accident, vec.len=2)

'data.frame': 80 obs. of  5 variables:
 $ age   : Ord.factor w/ 5 levels "0-9"<"10-19"<..: 5 5 5 5 5 ...
 $ result: Factor w/ 2 levels "Died","Injured": 1 1 1 1 1 ...
 $ mode  : Factor w/ 4 levels "4-Wheeled","Bicycle",..: 4 4 2 2 3 ...
 $ gender: Factor w/ 2 levels "Female","Male": 2 1 2 1 2 ...
 $ Freq  : int  704 378 396 56 742 ...
```

(a) Use `loglm()` to fit the model of mutual independence, `Freq ~ age+mode+gender+result` to this data set.
(b) Use `mosaic()` to produce an interpretable mosaic plot of the associations among all variables under the model of mutual independence. Try different orders of the variables in the mosaic. (*Hint*: the `abbreviate` component of the `labeling_args` argument to `mosaic()` will be useful to avoid some overlap of the category labels.)
(c) Treat `result` ("Died" vs. "Injured") as the response variable, and fit the model `Freq ~ age*mode*gender + result` that asserts independence of `result` from all others jointly.
(d) Construct a mosaic display for the residual associations in this model. Which combinations of the predictor factors are more likely to result in death?

Exercise 5.10 The data set *Vietnam* in vcdExtra gives a $2 \times 5 \times 4$ contingency table in frequency form reflecting a survey of student opinion on the Vietnam War at the University of North Carolina in May 1967. The table variables are sex, year in school, and response, which has categories: (A) Defeat North Vietnam by widespread bombing and land invasion; (B) Maintain the present policy; (C) De-escalate military activity, stop bombing and begin negotiations; (D) Withdraw military forces immediately. How does the chosen response vary with sex and year?

```
> data("Vietnam", package = "vcdExtra")
> str(Vietnam)

'data.frame': 40 obs. of  4 variables:
 $ sex     : Factor w/ 2 levels "Female","Male": 1 1 1 1 1 1 1 1 1 1 ...
 $ year    : int  1 1 1 1 2 2 2 2 3 3 ...
 $ response: Factor w/ 4 levels "A","B","C","D": 1 2 3 4 1 2 3 4 1 2 ...
 $ Freq    : int  13 19 40 5 5 9 33 3 22 29 ...
```

(a) With `response` (R) as the outcome variable and `year` (Y) and `sex` (S) as predictors, the minimal baseline loglinear model is the model of joint independence, [R][YS]. Fit this model, and display it in a mosaic plot.
(b) Construct conditional mosaic plots of the `response` versus `year` separately for males and females. Describe the associations seen here.
(c) Follow the methods shown in Example 5.10 to fit separate models of independence for the levels of `sex`, and the model of conditional independence, $R \perp Y \mid S$. Verify that the decomposition of G^2 in Eqn. (5.6) holds for these models.
(d) Construct a useful 3-way mosaic plot of the data for the model of conditional independence.

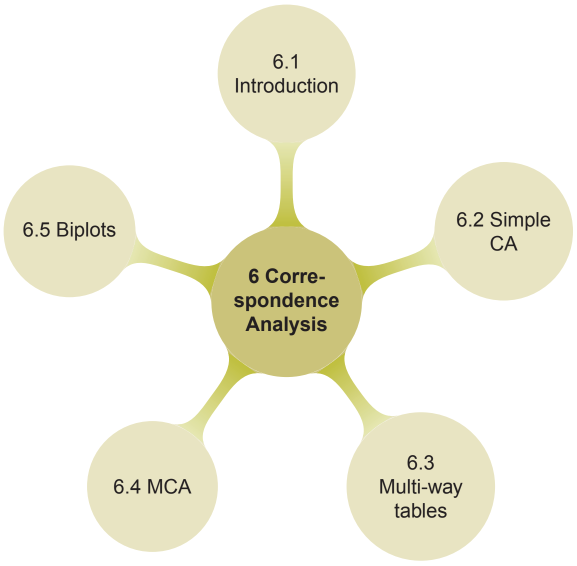
Exercise 5.11 Consider the models for 4-way tables shown in Table 5.3.

(a) For each model, give an independence interpretation. For example, the model of mutual independence corresponds to $A \perp B \perp C \perp D$.

(b) Use the functions shown in the table together with `loglin2formula()` to print the corresponding model formulas for each.

Exercise 5.12 The dataset *`Titanic`* classifies the 2,201 pasengers and crew of the *Titanic* by `Class` (1st, 2nd, 3rd, Crew), `Sex`, `Age`, and `Survived`. Treating `Survived` as the response variable,

(a) Fit and display a mosaic plot for the baseline model of joint independence, [CGA][S]. Describe the remaining pattern of associations.
(b) Do the same for a "main effects" model that allows two-way associations between each of C, G, and A with S.
(c) What three-way association term should be added to this model to allow for greater survival among women and children? Does this give an acceptable fit?
(d) Test and display models that allow additional three-way associations until you obtain a reasonable fit.



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 they maximize the correlation between the row and column variables. For graphical display two or three dimensions are typically used to give a reduced rank approximation to the data.

Correspondence analysis has a very large, multi-national literature and was rediscovered several times in different fields and different countries. The method, in slightly different forms, is also discussed under the names ***dual scaling***, ***optimal scaling***, ***reciprocal averaging***, ***homogeneity analysis***, and ***canonical analysis of categorical data***.

See Greenacre (1984) and Greenacre (2007) for an accessible introduction to CA methodology, or Gifi (1981) and Lebart et al. (1984) for a detailed treatment of the method and its applications from the Dutch and French perspectives. Greenacre and Hastie (1987) provide an excellent discussion of the geometric interpretation, while van der Heijden and de Leeuw (1985) and van der Heijden et al. (1989) develop some of the relations between correspondence analysis and log-linear methods for three-way and larger tables. Correspondence analysis is usually carried out in an exploratory, graphical way. Goodman (1981, 1985, 1986) has developed related inferential models, the RC model (see Section 10.1.3) and the canonical correlation model, with close links to CA.

One simple development of CA is as follows: For a two-way table the scores for the row categories, namely $\boldsymbol{X} = \{x_{im}\}$, and column categories, $\boldsymbol{Y} = \{y_{jm}\}$, on dimension $m = 1, \ldots, M$ are derived from a (generalized) ***singular value decomposition*** of (Pearson) residuals from independence, expressed as $d_{ij}/\sqrt{n}$, to account for the largest proportion of the χ^2 in a small number of dimensions. This decomposition may be expressed as

$$\frac{d_{ij}}{\sqrt{n}} = \frac{n_{ij} - m_{ij}}{\sqrt{n\, m_{ij}}} = \boldsymbol{X}\, \boldsymbol{D}_\lambda\, \boldsymbol{Y}^\mathsf{T} = \sum_{m=1}^{M} \lambda_m\, x_{im}\, y_{jm} \,, \tag{6.1}$$

where m_{ij} is the expected frequency and where $\boldsymbol{D}_\lambda$ is a diagonal matrix with elements $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_M$, and $M = \min(I-1,\, J-1)$. In M dimensions, the decomposition Eqn. (6.1) is exact. For example, an $I \times 3$ table can be depicted exactly in two dimensions when $I \geq 3$. The useful result for visualization purposes is that a rank-d approximation in d dimensions is obtained from the first d terms on the right side of Eqn. (6.1). The proportion of the Pearson χ^2 accounted for by this approximation is

$$n \sum_{m}^{d} \lambda_m^2 / \chi^2 \,.$$

The quantity $\chi^2/n = \sum_i \sum_j d_{ij}^2/n$ is called the total ***inertia*** and is identical to the measure of association known as Pearson's mean-square contingency, the square of the ϕ coefficient.

Thus, correspondence analysis is designed to show how the data deviate from expectation when the row and column variables are independent, as in the sieve diagram, association plot, and mosaic display. However, the sieve, association, and mosaic plots depict every *cell* in the table, and for large tables it may be difficult to see patterns. Correspondence analysis shows only row and column *categories* as points in the two (or three) dimensions that account for the greatest proportion of deviation from independence. The pattern of the associations can then be inferred from the positions of the row and column points.

6.2 Simple correspondence analysis

6.2.1 Notation and terminology

Because Correspondence analysis grew up in so many homes, the notation, formulae, and terms used to describe the method vary considerably. The notation used here generally follows Greenacre (1984, 1997, 2007).

The descriptions here employ the following matrix and vector definitions:

- $\boldsymbol{N} = \{n_{ij}\}$ is the $I \times J$ contingency table with row and column totals n_{i+} and n_{+j}, respectively. The grand total n_{++} is also denoted by n for simplicity.
- $\boldsymbol{P} = \{p_{ij}\} = \boldsymbol{N}/n$ is the matrix of joint cell proportions, called the ***correspondence matrix***.
- $\boldsymbol{r} = \sum_j p_{ij} = \boldsymbol{P}\boldsymbol{1}$ is the row margin of $\boldsymbol{P}$; $\boldsymbol{c} = \sum_i p_{ij} = \boldsymbol{P}^\mathsf{T}\boldsymbol{1}$ is the column margin. $\boldsymbol{r}$ and $\boldsymbol{c}$ are called the *row masses* and *column masses*.
- $\boldsymbol{D}_r$ and $\boldsymbol{D}_c$ are diagonal matrices with $\boldsymbol{r}$ and $\boldsymbol{c}$ on their diagonals, used as weights.
- $\boldsymbol{R} = \boldsymbol{D}_r^{-1}\boldsymbol{P} = \{n_{ij}/n_{+j}\}$ is the matrix of row conditional probabilities, called *row profiles*. Similarly, $\boldsymbol{C} = \boldsymbol{D}_c^{-1}\boldsymbol{P}^\mathsf{T} = \{n_{ij}/n_{i+}\}$ is the matrix of column conditional probabilities or *column profiles*.
- $\boldsymbol{S} = \boldsymbol{D}_r^{-1/2}(\boldsymbol{P} - \boldsymbol{r}\boldsymbol{c}^\mathsf{T})\boldsymbol{D}_c^{-1/2}$ is the matrix of standardized Pearson residuals from independence (denoted d_{ij} in the introduction).

Two types of coordinates, $\boldsymbol{X}$, $\boldsymbol{Y}$ for the row and column categories are defined, based on the singular value decomposition (SVD) of $\boldsymbol{S}$,

$$\boldsymbol{S} = \boldsymbol{U}\boldsymbol{D}_\lambda\boldsymbol{V}^\mathsf{T} \quad \text{where} \quad \boldsymbol{U}^\mathsf{T}\boldsymbol{U} = \boldsymbol{V}^\mathsf{T}\boldsymbol{V} = \boldsymbol{I}\,,$$

and $\boldsymbol{D}_\lambda$ is the diagonal matrix of singular values $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_M$. $\boldsymbol{U}$ is the orthonormal $I \times M$ matrix of left singular vectors, and $\boldsymbol{V}$ is the $J \times M$ matrix of right singular vectors.

The SVD of $\boldsymbol{S}$ is related to the eigenvalue–eigenvector decomposition of a square symmetric matrix, in that $\boldsymbol{S}\boldsymbol{S}^\mathsf{T} = \boldsymbol{U}\boldsymbol{D}_\lambda^2\boldsymbol{U}$ and $\boldsymbol{S}^\mathsf{T}\boldsymbol{S} = \boldsymbol{V}\boldsymbol{D}_\lambda^2\boldsymbol{V}$, so the values λ^2 are the eigenvalues in both cases and the singular vectors are the corresponding eigenvectors. In correspondence analysis, these eigenvalues (squares of the singular values) are called the ***principal inertia***s, and are the values used in the decomposition of the Pearson χ^2 for the dimensions, $\chi^2 = n\sum_m \lambda_m^2$.

principal coordinates: The coordinates of the row ($\boldsymbol{F}$) and column ($\boldsymbol{G}$) profiles with respect to their own principal axes are defined so that the inertia along each axis is the corresponding eigenvalue value, λ_m,

$$\begin{aligned} \boldsymbol{F} &= \boldsymbol{D}_r^{-1/2}\boldsymbol{U}\boldsymbol{D}_\lambda^2 \quad \text{scaled so that} \quad \boldsymbol{F}^\mathsf{T}\boldsymbol{D}_r\boldsymbol{F} = \boldsymbol{D}_\lambda^2\,, & (6.2)\\ \boldsymbol{G} &= \boldsymbol{D}_c^{-1/2}\boldsymbol{V}\boldsymbol{D}_\lambda^2 \quad \text{scaled so that} \quad \boldsymbol{G}^\mathsf{T}\boldsymbol{D}_c\boldsymbol{G} = \boldsymbol{D}_\lambda^2\,. & (6.3)\end{aligned}$$

The joint plot in principal coordinates, $\boldsymbol{F}$ and $\boldsymbol{G}$, is called the ***symmetric map*** because both row and column profiles are overlaid in the same coordinate system.

standard coordinates: The standard coordinates $(\boldsymbol{\Phi}, \boldsymbol{\Gamma})$ are a rescaling of the principal coordinates to unit inertia along each axis,

$$\begin{aligned} \boldsymbol{\Phi} &= \boldsymbol{D}_r^{-1}\boldsymbol{U} \quad \text{scaled so that} \quad \boldsymbol{\Phi}^\mathsf{T}\boldsymbol{D}_r\boldsymbol{\Phi} = \boldsymbol{I}\,, & (6.4)\\ \boldsymbol{\Gamma} &= \boldsymbol{D}_c^{-1}\boldsymbol{V} \quad \text{scaled so that} \quad \boldsymbol{\Gamma}^\mathsf{T}\boldsymbol{D}_c\boldsymbol{\Gamma} = \boldsymbol{I}\,. & (6.5)\end{aligned}$$

These differ from the principal coordinates in Eqn. (6.2) and Eqn. (6.3) simply by the absence of the scaling factors, $\boldsymbol{D}_\lambda^2$. An ***asymmetric map*** shows one set of points (say, the rows) in principal coordinates and the other set in standard coordinates.

Thus, the weighted average of the squared principal coordinates for the rows or columns on a principal axis equals the squared singular value, λ^2 for that axis, whereas the weighted average of the squared standard coordinates equals 1. The relative positions of the row or column points along any axis is the same under either scaling, but the distances between points differ, because the axes are weighted differentially in the two scalings.

6.2.2 Geometric and statistical properties

We summarize here some geometric and statistical properties of the Correspondence analysis solutions that are useful in interpretation.

nested solutions: Because they use successive terms of the SVD Eqn. (6.1), correspondence analysis solutions are *nested*, meaning that the first two dimensions of a three-dimensional solution will be identical to the two-dimensional solution.

centroids at the origin: In both principal coordinates and standard coordinates the points representing the row and column profiles have their centroids (weighted averages) at the origin. Thus, in CA plots, the origin represents the (weighted) average row profile and column profile.

reciprocal averages: CA assigns scores to the row and column categories such that the column scores are proportional to the weighted averages of the row scores, and vice-versa.

chi-square distances: In principal coordinates, the row coordinates may be shown equal to the row profiles $\boldsymbol{D}_r^{-1}\boldsymbol{P}$, rescaled by the inverse by the square-root of the column masses, $\boldsymbol{D}_c^{-1/2}$. Distances between two row profiles, $\boldsymbol{R}_i$ and $\boldsymbol{R}_{i'}$, are most sensibly defined as χ^2 distances, where the squared difference $[\boldsymbol{R}_{ij} - \boldsymbol{R}_{i'j}]^2$ is inversely weighted by the column frequency, to account for the different relative frequency of the column categories. The rescaling by $\boldsymbol{D}_c^{-1/2}$ transforms this weighted χ^2 metric into ordinary Euclidean distance. The same is true of the column principal coordinates.

interpretation of distances: In principal coordinates, the distance between two row points may be interpreted as described above, and so may the distance between two column points. The distance between a row and column point, however, does not have a clear distance interpretation.

residuals from independence: The distance between a row and column point do have a rough interpretation in terms of residuals or the difference between observed and expected frequencies, $n_{ij} - m_{ij}$. Two row (or column) points deviate from the origin (the average profile) when their profile frequencies have similar values. A row point appears in a similar direction away from the origin as a column point when $n_{ij} - m_{ij} > 0$, and in an opposite different direction from that column point when the residual is negative.

Because of these differences in interpretations of distances, there are different possibilities for graphical display. A joint display of principal coordinates for the rows and standard coordinates for the columns (or vice-versa), sometimes called an ***asymmetric map***, is suggested by Greenacre and Hastie (1987) and by Greenacre (1989) as the plot with the most coherent geometric interpretation (for the points in principal coordinates) and is sometimes used in the French literature.

Another common joint display is the ***symmetric map*** of the principal coordinates in the same plot. This is the default in the ca package described below. In the authors' opinion, this produces better graphical displays, because both sets of coordinates are scaled with the same weights for each axis. Symmetric plots are used exclusively in this book, but that should not imply that these plots are universally preferred. Another popular choice is to avoid the possibility of misinterpretation by making separate plots of the row and column coordinates.

6.2.3 R software for correspondence analysis

Correspondence analysis methods for computation and plotting are available in a number of R packages including:

MASS: `corresp()`; the plot method calls `biplot()` for a 2-factor solution, using a a symmetric biplot factorization that scales the row and column points by the square roots of the the singular values. There is also an `mca()` function for multiple correspondence analysis.

ca: `ca()`; provides 2D plots via the `plot.ca()` method and interactive (rgl) 3D plots via `plot3d.ca()`. This package is the most comprehensive in terms of plotting options for various coordinate types, plotting supplementary points (see Section 6.3.2), and other features. It also provides `mjca()` for multiple and joint correspondence analysis of higher-way tables.

FactoMineR (Husson et al., 2015): `CA()`; provides a wide variety of measures for the quality of the CA representation and many options for graphical display

These methods also differ in terms of the types of input they accept. For example, MASS::`corresp()` handles matrices, data frames, and "xtabs" objects, but not "table" objects. `ca()` is the most general, with methods for two-way tables, matrices, data frames, and "xtabs" objects. In the following, we largely use the ca package.

EXAMPLE 6.1: Hair color and eye color

The script below uses the two-way table `haireye` from the *`HairEyeColor`* data, collapsed over `Sex`. In this table, `Hair` colors form the rows, and `Eye` colors form the columns. By default, `ca()` produces a two-dimensional solution. In this example, the complete, exact solution would have $M = \min((I-1),(J-1)) = 3$ dimensions, and you could obtain this using the argument `nd=3` in the call to `ca()`.

```
> haireye <- margin.table(HairEyeColor, 1 : 2)
> library(ca)
> (haireye.ca <- ca(haireye))


 Principal inertias (eigenvalues):
           1        2        3
Value      0.208773 0.022227 0.002598
Percentage 89.37%   9.52%    1.11%


 Rows:
             Black    Brown      Red  Blond
Mass       0.18243  0.48311  0.1199 0.2145
ChiDist    0.55119  0.15946  0.3548 0.8384
Inertia    0.05543  0.01228  0.0151 0.1508
Dim. 1    -1.10428 -0.32446 -0.2835 1.8282
Dim. 2     1.44092 -0.21911 -2.1440 0.4667


 Columns:
             Brown   Blue    Hazel    Green
Mass       0.37162 0.3632  0.15710  0.10811
ChiDist    0.50049 0.5537  0.28865  0.38573
Inertia    0.09309 0.1113  0.01309  0.01608
Dim. 1    -1.07713 1.1981 -0.46529  0.35401
Dim. 2     0.59242 0.5564 -1.12278 -2.27412
```

In the printed output, the table labeled "Principal inertias (eigenvalues)" indicates that nearly 99% of the Pearson χ^2 for association is accounted for by two dimensions, with most of that attributed to the first dimension.

The `summary` method for "ca" objects gives a more nicely formatted display, showing a ***scree plot*** of the eigenvalues, a portion of which is shown below.

```
> summary(haireye.ca)


Principal inertias (eigenvalues):

 dim    value       %    cum%    scree plot
 1      0.208773  89.4  89.4  **********************
 2      0.022227   9.5  98.9  **
 3      0.002598   1.1 100.0
        -------- -----
 Total: 0.233598 100.0
...
```

The Pearson χ^2 for this table (given by `chisq.test(haireye)`) is 138.29. This value is n (592) times the sum of the eigenvalues (0.2336) shown above.

The result returned by `ca()` can be plotted using the `plot.ca()` method. However, it is useful to understand that `ca()` returns the CA solution in terms of *standard coordinates*, Φ (`rowcoord`) and Γ (`colcoord`). We illustrate Eqn. (6.4) and Eqn. (6.5) using the components of the "ca" object `haireye.ca`.

```
> # standard coordinates Phi (Eqn 6.4) and Gamma (Eqn 6.5)
> (Phi <- haireye.ca$rowcoord)

           Dim1     Dim2     Dim3
Black -1.10428  1.44092 -1.08895
Brown -0.32446 -0.21911  0.95742
Red   -0.28347 -2.14401 -1.63122
Blond  1.82823  0.46671 -0.31809

> (Gamma <- haireye.ca$colcoord)

           Dim1     Dim2      Dim3
Brown -1.07713  0.59242 -0.423960
Blue   1.19806  0.55642  0.092387
Hazel -0.46529 -1.12278  1.971918
Green  0.35401 -2.27412 -1.718443

> # demonstrate orthogonality of std coordinates
> Dr <- diag(haireye.ca$rowmass)
> zapsmall(t(Phi) %*% Dr %*% Phi)

     Dim1 Dim2 Dim3
Dim1    1    0    0
Dim2    0    1    0
Dim3    0    0    1

> Dc <- diag(haireye.ca$colmass)
> zapsmall(t(Gamma) %*% Dc %*% Gamma)

     Dim1 Dim2 Dim3
Dim1    1    0    0
Dim2    0    1    0
Dim3    0    0    1
```

These standard coordinates are transformed internally within the plot function according to the `map` argument, which defaults to `map="symmetric"`, giving principal coordinates. The following call to `plot.ca()` produces Figure 6.1.

```
> res <- plot(haireye.ca)
```

For use in further customizing such plots (as we will see in the next example), the function `plot.ca()` returns (invisibly) the coordinates for the row and column points actually plotted, which we saved above as `res`:

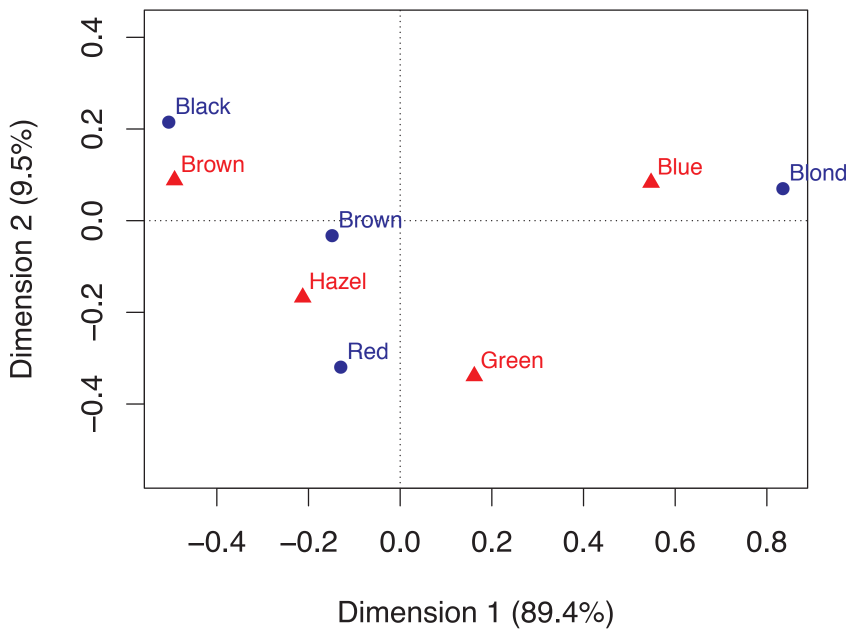


Figure 6.1: Correspondence analysis solution for the hair color and eye color data.

```
> res

$rows
          Dim1       Dim2
Black -0.50456  0.214820
Brown -0.14825 -0.032666
Red   -0.12952 -0.319642
Blond  0.83535  0.069579

$cols
          Dim1       Dim2
Brown -0.49216  0.088322
Blue   0.54741  0.082954
Hazel -0.21260 -0.167391
Green  0.16175 -0.339040
```

It is important to understand that in CA plots (and related biplots, Section 6.5), the interpretation of distances between points (and angles between vectors) is meaningful. In order to achieve this, the axes in such plots must be *equated*, meaning that the two axes are scaled so that the number of data units per inch are the same for both the horizontal and vertical axes, or an ***aspect ratio*** = 1.[1]

The interpretation of the CA plot in Figure 6.1 is then as follows:

- Dimension 1, accounting for nearly 90% of the association between hair and eye color corresponds to dark (left) vs. light (right) on both variables.
- Dimension 2 largely contrasts red hair and green eyes with the remaining categories, accounting for an additional 9.5% of the Pearson χ^2.
- With equated axes, and a symmetric map, the distances between row points and distances between column points are meaningful. Along Dimension 1, the eye colors could be considered

[1]In base R graphics, this is achieved with the `plot()` option `asp=1`.

roughly equally spaced, but for the hair colors, Blond is quite different in terms of its frequency profile.

△

EXAMPLE 6.2: Mental impairment and parents' SES

In Example 4.3 we introduced the data set *Mental*, relating mental health status to parents' SES. As in Example 4.7, we convert this to a two-way table, `mental.tab`, to conduct a correspondence analysis.

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

We calculate the CA solution, and save the result in `mental.ca`:

```
> mental.ca <- ca(mental.tab)
> summary(mental.ca)


Principal inertias (eigenvalues):

 dim    value      %   cum%   scree plot
 1      0.026025  93.9  93.9  ***********************
 2      0.001379   5.0  98.9  *
 3      0.000298   1.1 100.0
        -------- -----
 Total: 0.027702 100.0
...
```

The scree plot produced by `summary(mental.ca)` shows that the association between mental health and parents' SES is almost entirely 1-dimensional, with 94% of the χ^2 (45.98, with 15 df) accounted for by Dimension 1.

We then plot the solution as shown below, giving Figure 6.2. For this example, it is useful to connect the row points and the column points by lines, to emphasize the pattern of these ordered variables.

```
> res <- plot(mental.ca,  ylim = c(-.2, .2))
> lines(res$rows, col = "blue", lty = 3)
> lines(res$cols, col = "red", lty = 4)
```

The plot of the CA scores in Figure 6.2 shows that diagnostic mental health categories are well-aligned with Dimension 1. The mental health scores are approximately equally spaced, except that the two intermediate categories are a bit closer on this dimension than the extremes. The SES categories are also aligned with Dimension 1, and approximately equally spaced, with the exception of the highest two SES categories, whose profiles are extremely similar, suggesting that these two categories could be collapsed.

Because both row and column categories have the same pattern on Dimension 1, we may interpret the plot as showing that the profiles of both variables are ordered, and their relation can be explained as a positive association between high parents' SES and higher mental health status of children. A mosaic display of these data (Exercise 6.5) would show a characteristic opposite corner pattern of association.

From a modeling perspective, we might ask how strong is the evidence for the spacing of categories noted above. For example, we might ask whether assigning integer scores to the levels of SES and mental impairment provides a simpler, but satisfactory account of their association. Questions of this type can be explored in connection with loglinear models in Chapter 9.

△

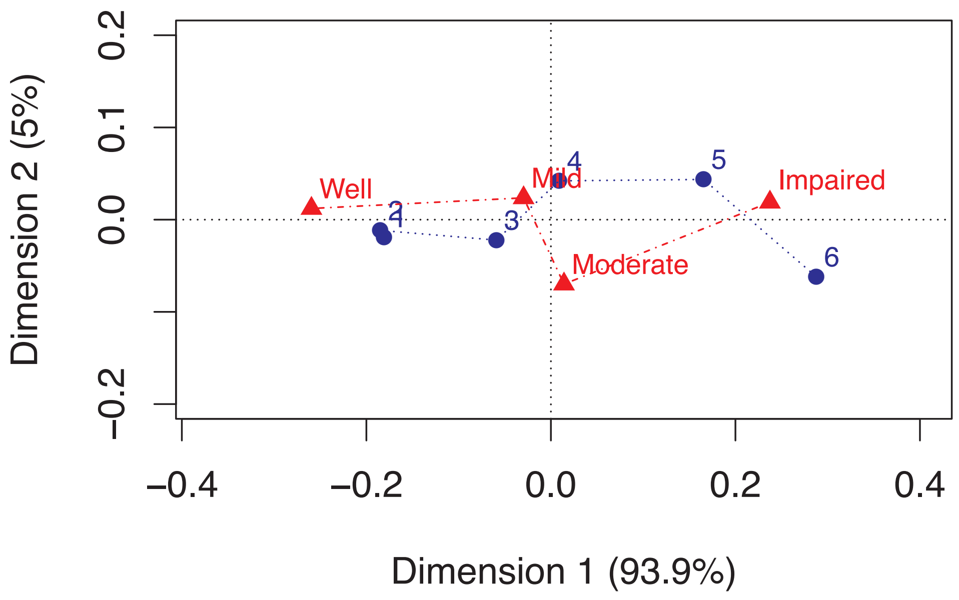


Figure 6.2: Correspondence analysis solution for the Mental health data.

Example 6.3: Repeat victimization

The data set *RepVict* in the vcd package gives an 8×8 table (from Fienberg (1980, Table 2-8)) on repeat victimization for various crimes among respondents to a U.S. National Crime Survey. A special feature of this data set is that row and column categories reflect the *same* crimes, so substantial association is expected. Here we examine correspondence analysis results in a bit more detail and also illustrate how to customize the displays created by `plot(ca(...))`.

```
> data("RepVict", package = "vcd")
> victim.ca <- ca(RepVict)
> summary(victim.ca)


Principal inertias (eigenvalues):

 dim    value      %   cum%   scree plot
 1      0.065456  33.8  33.8  ********
 2      0.059270  30.6  64.5  ********
 3      0.029592  15.3  79.8  ****
 4      0.016564   8.6  88.3  **
 5      0.011140   5.8  94.1  *
 6      0.007587   3.9  98.0  *
 7      0.003866   2.0 100.0
        -------- -----
 Total: 0.193474 100.0
...
```

The results above show that, for this 8×8 table, 7 dimensions are required for an exact solution, of which the first two account for 64.5% of the Pearson χ^2. The lines below illustrate that the Pearson χ^2 is n times the sum of the squared singular values, $n \sum \lambda_i^2$.

```
> chisq.test(RepVict)


Pearson's Chi-squared test

data:  RepVict
X-squared = 11100, df = 49, p-value <2e-16

> (chisq <- sum(RepVict) * sum(victim.ca$sv^2))

[1] 11131
```

The default plot produced by `plot.ca(victim.ca)` plots both points and labels for the row and column categories. However, what we want to emphasize here is the relation between the *same* crimes on the first and second occurrence.

To do this, we label each crime just once (using `labels=c(2,0)`) and connect the two points for each crime by a line, using `segments()`, as shown in Figure 6.3. The addition of a `legend()` makes the plot more easily readable.

```
> res <- plot(victim.ca, labels = c(2, 0))
> segments(res$rows[,1], res$rows[,2], res$cols[,1], res$cols[,2])
> legend("topleft", legend = c("First", "Second"), title = "Occurrence",
+         col = c("blue", "red"), pch = 16 : 17, bg = "gray90")
```

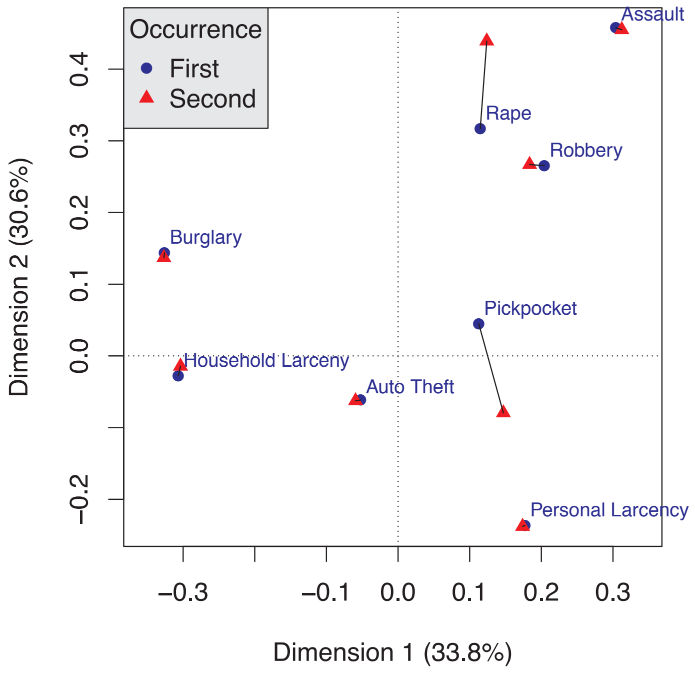


Figure 6.3: 2D CA solution for the repeat victimization data. Lines connect the category points for first and second occurrence to highlight these relations.

In Figure 6.3 it may be seen that most of the points are extremely close for the first and second occurrence of a crime, indicating that the row profile for a crime is very similar to its corresponding column profile, with Rape and Pickpocket as exceptions.

In fact, if the table was symmetric, the row and column points in Figure 6.3 would be identical, as can be easily demonstrated by analyzing a symmetric version.

```
> RVsym <- (RepVict + t(RepVict)) / 2
> RVsym.ca <- ca(RVsym)
> res <- plot(RVsym.ca)
> all.equal(res$rows, res$cols)

[1] TRUE
```

The first dimension appears to contrast crimes against the person (right) with crimes against property (left), and it may be that the second dimension represents degree of violence associated with each crime. The latter interpretation is consistent with the movement of Rape towards a higher position and Pickpocket towards a lower one on this dimension.

△

6.2.4 Correspondence analysis and mosaic displays

For a two-way table, CA and mosaic displays give complementary views of the pattern of association between the row and column variables, but both are based on the (Pearson) residuals from independence. CA shows the row and column categories as points in a 2D (or 3D) space accounting for the largest proportion of the Pearson χ^2, while mosaics show the association by the pattern of shading in the mosaic tiles. It is useful to compare them directly to see how associations can be interpreted from these graphs.

Example 6.4: TV viewing data

The data on television viewership from Hartigan and Kleiner (1984) was used as an example of manipulating complex categorical data in Section 2.9. The main association here concerns how viewership across days of the week varies by TV network, so we first collapse the *TV* data to a 5×3 two-way table.

```
> data("TV", package = "vcdExtra")
> TV2 <- margin.table(TV, c(1, 3))
> TV2

           Network
Day          ABC  CBS  NBC
  Monday    2847 2923 2629
  Tuesday   3110 2403 2568
  Wednesday 2434 1283 2212
  Thursday  1766 1335 5886
  Friday    2737 1479 1998
```

In this case, the 2D CA solution is exact, meaning that two dimensions account for 100% of the association.

```
> TV.ca <- ca(TV2)
> TV.ca


 Principal inertias (eigenvalues):
           1        2
Value      0.081934 0.010513
Percentage 88.63%   11.37%
...
```

The plot of this solution is shown in the left panel of Figure 6.4, using lines from the origin to the category points for the networks.

```
> res <- plot(TV.ca)
> segments(0, 0, res$cols[,1], res$cols[,2], col = "red", lwd = 2)
```

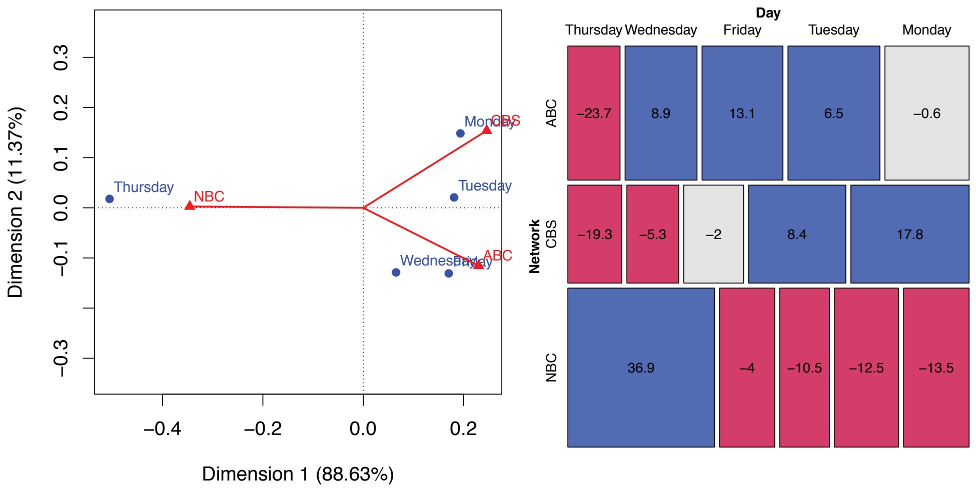


Figure 6.4: CA plot and mosaic display for the TV viewing data. The days of the week in the mosaic plot were permuted according to their order in the CA solution.

An analogous mosaic display, informed by the CA solution, is shown in the right panel of Figure 6.4. Here, the days of the week are reordered according to their positions on the first CA dimension, another example of effect ordering.

```
> days.order <- order(TV.ca$rowcoord[,1])
> mosaic(t(TV2[days.order,]), shade = TRUE, legend = FALSE,
+         labeling = labeling_residuals, suppress=0)
```

In the CA plot, you can see that the dominant dimension separates viewing on Thursday, with the largest share of viewers watching NBC, from the other weekdays. In the mosaic plot, Thursday stands out as the only day with a higher than expected frequency for NBC, and this is the largest residual in the entire table. The second dimension in the CA plot separates CBS, with its greatest proportion of viewers on Monday, from ABC, with greater viewership on Wednesday and Friday.

Emerson (1998, Fig. 2) gives a table listing the shows in each half-hour time slot. Could the overall popularity of NBC on Thursday have been due to *Friends* or *Seinfeld*? An answer to this and similar questions requires analysis of the three-way table (Exercise 6.9) and model-based methods for polytomous outcome variables described in Section 8.3.

△

6.3 Multi-way tables: Stacking and other tricks

A three- or higher-way table can be analyzed by correspondence analysis in several ways. Multiple correspondence analysis (MCA), described in Section 6.4, is an extension of simple correspondence analysis that analyzes simultaneously all possible two-way tables contained within a multiway table. Another approach, described here, is called ***stacking*** or ***interactive coding***. This is a bit of a trick, to force a multiway table into a two-way table for a standard correspondence analysis, but it is a useful one.

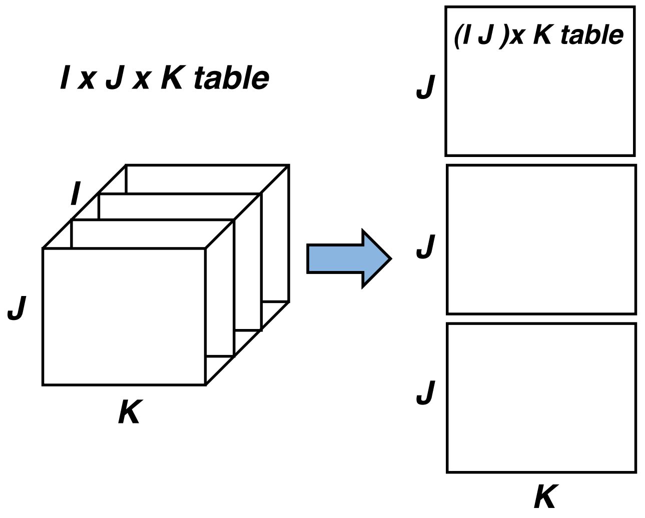


Figure 6.5: Stacking approach for a three-way table. Two of the table variables are combined interactively to form the rows of a two-way table.

A three-way table of size $I \times J \times K$ can be sliced into I two-way tables, each $J \times K$. If the slices are concatenated vertically, the result is one two-way table, of size $(I \times J) \times K$, as illustrated in Figure 6.5. In effect, the first two variables are treated as a single composite variable with IJ levels, which represents the main effects and interaction between the original variables that were combined. Van der Heijden and de Leeuw (1985) discuss this use of correspondence analysis for multi-way tables and show how *each* way of slicing and stacking a contingency table corresponds to the analysis of a specified loglinear model. Like the mosaic display, this provides another way to visualize the relations in a loglinear model.

In particular, for the three-way table with variables A, B, C that is reshaped as a table of size $(I \times J) \times K$, the correspondence analysis solution analyzes residuals from the log-linear model $[AB][C]$. That is, for such a table, the $I \times J$ rows represent the joint combinations of variables A and B. The expected frequencies under independence for this table are

$$m_{[ij]k} = \frac{n_{ij+}\, n_{++k}}{n} , \tag{6.6}$$

which are the ML estimates of expected frequencies for the log-linear model $[AB][C]$. The χ^2 that is decomposed by correspondence analysis is the Pearson χ^2 for this log-linear model. When the table is stacked as $I \times (J \times K)$ or $J \times (I \times K)$, correspondence analysis decomposes the residuals from the log-linear models $[A][BC]$ and $[B][AC]$, respectively, as shown in Table 6.1. In this approach, only the associations in separate [ ] terms are analyzed and displayed in the correspondence analysis maps. Van der Heijden and de Leeuw (1985) show how a generalized form of correspondence analysis can be interpreted as decomposing the difference between two specific loglinear models, so their approach is more general than is illustrated here.

6.3.1 Interactive coding in R

In the general case of an n-way table, the stacking approach is similar to that used by `ftable()` and `structable()` in vcd as described in Section 2.5 to flatten multiway tables to a two-way, printable form, where some variables are assigned to the rows and the others to the columns. Both

Table 6.1: Each way of stacking a three-way table corresponds to a loglinear model

Stacking structure	Loglinear model
$(I \times J) \times K$	$[AB][C]$
$I \times (J \times K)$	$[A][BC]$
$J \times (I \times K)$	$[B][AC]$

`ftable()` and `structable()` have `as.matrix()` methods[2] that convert their result into a matrix suitable as input to `ca()`.

With data in the form of a frequency data frame, you can easily create interactive coding using `interaction()` or simply use `paste()` to join the levels of stacked variables together.

To illustrate, create a 4-way table of random Poisson counts (with constant mean, $\lambda = 15$) of types of Pet, classified by Age, Color, and Sex.

```
> set.seed(1234)
> dim <- c(3, 2, 2, 2)
> tab <- array(rpois(prod(dim), 15), dim = dim)
> dimnames(tab) <- list(Pet = c("dog", "cat", "bird"),
+                       Age = c("young", "old"),
+                       Color = c("black", "white"),
+                       Sex = c("male", "female"))
```

You can use `ftable()` to print this, with a formula that assigns `Pet` and `Age` to the columns and `Color` and `Sex` to the rows.

```
> ftable(Pet + Age ~ Color + Sex, tab)

             Pet   dog       cat       bird
             Age young old young old young old
Color Sex
black male          10  12    16  16    16  12
      female         8  12    13  15    11  13
white male          18  11    12  18    13  20
      female        13  13    16  15    12  15
```

Then, `as.matrix()` creates a matrix with the levels of the stacked variables combined with some separator character. Using `ca(pet.mat)` would then calculate the CA solution for the stacked table, analyzing only the associations in the loglinear model [Pet Age][Color Sex].[3]

```
> (pet.mat <- as.matrix(ftable(Pet + Age ~ Color + Sex, tab), sep = '.'))

              Pet.Age
Color.Sex      dog.young dog.old cat.young cat.old bird.young bird.old
  black.male          10      12        16      16         16       12
  black.female         8      12        13      15         11       13
  white.male          18      11        12      18         13       20
  white.female        13      13        16      15         12       15
```

With data in a frequency data frame, a similar result (as a frequency table) can be obtained using `interaction()` as shown below. The result of `xtabs()` looks the same as `pet.mat`.

```
> tab.df <- as.data.frame(as.table(tab))
> tab.df <- within(tab.df,
+   {Pet.Age = interaction(Pet, Age)
```

[2]This requires at least R version 3.1.0 or vcd 1.3-2 or later.
[3]The result would not be at all interesting here. Why?

```
+   Color.Sex = interaction(Color, Sex)
+   })
> xtabs(Freq ~ Color.Sex + Pet.Age, data = tab.df)
```

EXAMPLE 6.5: Suicide rates in Germany

To illustrate the use of correspondence analysis for the analysis for three-way tables, we use data on suicide rates in West Germany classified by sex, age, and method of suicide used. The data, from Heuer (1979, Table 1) have been discussed by Friendly (1991, 1994b), van der Heijden and de Leeuw (1985), and others.

The original $2 \times 17 \times 9$ table contains 17 age groups from 10 to 90 in 5-year steps and 9 categories of suicide method, contained in the frequency data frame *Suicide* in vcd, with table variables sex, age, and method. To avoid extremely small cell counts and cluttered displays, this example uses a reduced table in which age groups are combined in the variable age.group, a factor with 15-year intervals except for the last interval, which includes ages 70–90; the methods "toxic gas" and "cooking gas" were collapsed (in the variable method2) giving the $2 \times 5 \times 8$ table shown in the output below. These changes do not affect the general nature of the data or conclusions drawn from them.

In this example, we decided to stack the combinations of age and sex, giving an analysis of the loglinear model $[AgeSex][Method]$, to show how the age–sex categories relate to method of suicide.

In the case of a frequency data frame, it is quite simple to join two or more factors to form the rows of a new two-way table. Here we use paste() to form a new, composite factor, called age_sex here, abbreviating sex for display purposes.

```
> data("Suicide", package = "vcd")
> # interactive coding of sex and age.group
> Suicide <- within(Suicide, {
+   age_sex <- paste(age.group, toupper(substr(sex, 1, 1)))
+   })
```

Then, use xtabs() to construct the two-way table suicide.tab:

```
> suicide.tab <- xtabs(Freq ~ age_sex + method2, data = Suicide)
> suicide.tab

         method2
age_sex   poison  gas hang drown  gun knife jump other
  10-20 F    921   40  212    30   25    11  131   100
  10-20 M   1160  335 1524    67  512    47  189   464
  25-35 F   1672  113  575   139   64    41  276   263
  25-35 M   2823  883 2751   213  852   139  366   775
  40-50 F   2224   91 1481   354   52    80  327   305
  40-50 M   2465  625 3936   247  875   183  244   534
  55-65 F   2283   45 2014   679   29   103  388   296
  55-65 M   1531  201 3581   207  477   154  273   294
  70-90 F   1548   29 1355   501    3    74  383   106
  70-90 M    938   45 2948   212  229   105  268   147
```

The results of the correspondence analysis of this table are shown below:

```
> suicide.ca <- ca(suicide.tab)
> summary(suicide.ca)


Principal inertias (eigenvalues):
 dim    value      %    cum%   scree plot
 1      0.096151  57.2  57.2  **************
 2      0.059692  35.5  92.6  *********
 3      0.008183   4.9  97.5  *
 4      0.002158   1.3  98.8
 5      0.001399   0.8  99.6
 6      0.000557   0.3 100.0
 7      6.7e-050   0.0 100.0
        -------- -----
 Total: 0.168207 100.0
...
```

It can be seen that 92.6% of the χ^2 for this model is accounted for in the first two dimensions. Plotting these gives the display shown in Figure 6.6.

```
> plot(suicide.ca)
```

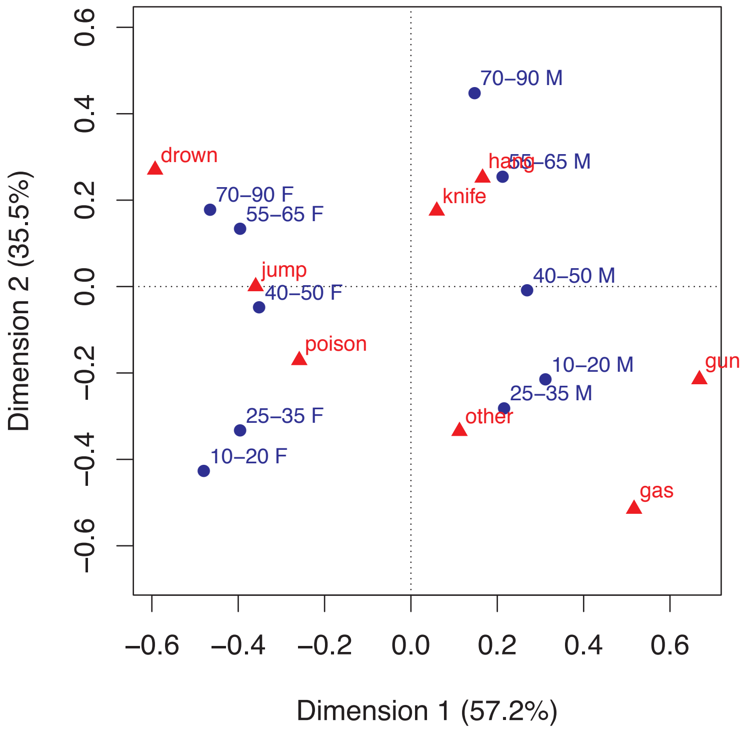


Figure 6.6: 2D CA solution for the stacked [AgeSex][Method] table of the suicide data.

Dimension 1 in the plot separates males (right) and females (left), indicating a large difference between suicide profiles of males and females with respect to methods of suicide. The second dimension is mostly ordered by age with younger groups at the bottom and older groups at the top. Note also that the positions of the age groups are roughly parallel for the two sexes. Such a pattern indicates that sex and age do not interact in this analysis.

The relation between the age–sex groups and methods of suicide can be approximately interpreted in terms of similar distance and direction from the origin, which represents the marginal row and column profiles. Young males are more likely to commit suicide by gas or a gun, older males by hanging, while young females are more likely to ingest some toxic agent and older females by jumping or drowning. △

EXAMPLE 6.6: Suicide rates in Germany — mosaic plot

For comparison, it is useful to see how to construct a mosaic display showing the same associations for the loglinear model $[AS][M]$ as in the correspondence analysis plot. To do this, we first construct the three-way table, `suicide.tab3`,

```
> suicide.tab3 <- xtabs(Freq ~ sex + age.group + method2, data = Suicide)
```

As discussed in Chapter 5, mosaic plots are sensitive both to the order of variables used in successive splits, and to the order of levels within variables and are most effective when these orders are chosen to reflect the some meaningful ordering.

In the present example, `method2` is an unordered table factor, but Figure 6.6 shows that the methods of suicide vary systematically with both sex and age, corresponding to dimensions 1 and 2, respectively. Here we choose to reorder the table according to the coordinates on Dimension 1. We also delete the low-frequency `"other"` category to simplify the display.

```
> # methods, ordered as in the table
> suicide.ca$colnames

[1] "poison" "gas"    "hang"   "drown"  "gun"    "knife"
[7] "jump"   "other"

> # order of methods on CA scores for Dim 1
> suicide.ca$colnames[order(suicide.ca$colcoord[,1])]

[1] "drown"  "jump"   "poison" "knife"  "other"  "hang"
[7] "gas"    "gun"

> # reorder methods by CA scores on Dim 1
> suicide.tab3 <- suicide.tab3[, , order(suicide.ca$colcoord[,1])]
> # delete "other"
> suicide.tab3 <- suicide.tab3[,, -5]
> ftable(suicide.tab3)

                method2 drown jump poison knife hang  gas  gun
sex    age.group
male   10-20               67  189   1160    47 1524  335  512
       25-35              213  366   2823   139 2751  883  852
       40-50              247  244   2465   183 3936  625  875
       55-65              207  273   1531   154 3581  201  477
       70-90              212  268    938   105 2948   45  229
female 10-20               30  131    921    11  212   40   25
       25-35              139  276   1672    41  575  113   64
       40-50              354  327   2224    80 1481   91   52
       55-65              679  388   2283   103 2014   45   29
       70-90              501  383   1548    74 1355   29    3
```

To construct the mosaic display for the same model analyzed by correspondence analysis, we use the argument `expected=~age.group*sex + method2` to supply the model formula. For this large table, it is useful to tweak the labels for the `method2` variable to reduce overplotting; the `labeling_args` argument provides many options for customizing `strucplot` displays.

```
> library(vcdExtra)
> mosaic(suicide.tab3, shade = TRUE, legend = FALSE,
+        expected = ~ age.group * sex + method2,
+        labeling_args = list(abbreviate_labs = c(FALSE, FALSE, 5)),
+                             rot_labels = c(0, 0, 0, 90))
```

This figure (Figure 6.7) again shows the prevalence of `gun` and `gas` among younger males and decreasing with age, whereas use of `hang` increases with age. For females, these three methods are used less frequently, whereas `poison`, `jump`, and `drown` occur more often. You can also see that

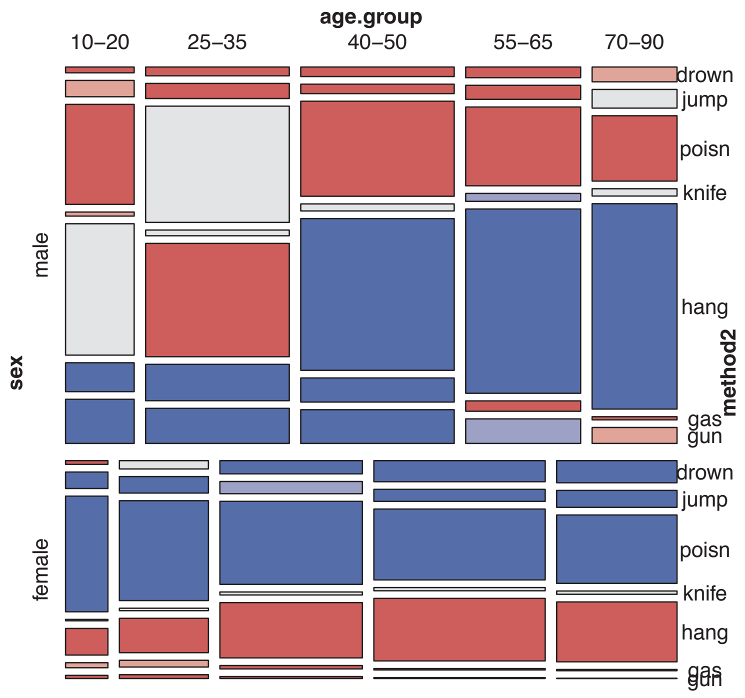


Figure 6.7: Mosaic display showing deviations from the model [AgeSex][Method] for the suicide data.

for females the excess prevalence of these high-frequency methods varies somewhat less with age than it does for males.

△

6.3.2 Marginal tables and supplementary variables

An *n*-way table in frequency form or case form is automatically collapsed over factors that are not listed in the call to `xtabs()` when creating the table input for `ca()`. The analysis gives a ***marginal model*** for the categorical variables that *are* listed.

The positions of the categories of the omitted variables may nevertheless be recovered, by treating them as ***supplementary variables***, given as additional rows or columns in the two-way table. A supplementary variable is ignored in finding the CA solution, but its categories are then projected into that space. This is another useful trick to extend traditional CA to higher-way tables.

To illustrate, the code below lists only the `age` and `method2` variables, and hence produces an analysis collapsed over `sex`. This ignores not only the effect of sex itself, but also all associations of age and method with sex, which are substantial. We don't show the `ca()` result or the plot yet.

```
> # two way, ignoring sex
> suicide.tab2 <- xtabs(Freq ~ age.group + method2, data = Suicide)
> suicide.tab2

         method2
age.group poison  gas hang drown  gun knife jump other
    10-20   2081  375 1736    97  537    58  320   564
    25-35    4495   996 3326    352   916    180   642   1038
    40-50    4689   716 5417    601   927    263   571    839
    55-65    3814   246 5595    886   506    257   661    590
    70-90    2486    74 4303    713   232    179   651    253

> suicide.ca2 <- ca(suicide.tab2)
```

To treat the levels of sex as supplementary points, we calculate the two-way table of sex and method, and append this to the suicide.tab2 as additional rows:

```
> # relation of sex and method
> suicide.sup <- xtabs(Freq ~ sex + method2, data = Suicide)
> suicide.tab2s <- rbind(suicide.tab2, suicide.sup)
```

In the call to ca(), we then indicate these last two rows as supplementary:

```
> suicide.ca2s <- ca(suicide.tab2s, suprow = 6 : 7)
> summary(suicide.ca2s)


Principal inertias (eigenvalues):

 dim     value       %    cum%    scree plot
 1       0.060429   93.9   93.9   ***********************
 2       0.002090    3.2   97.1   *
 3       0.001479    2.3   99.4   *
 4       0.000356    0.6  100.0
         --------- -----
 Total: 0.064354 100.0

...
```

This CA analysis has the same total Pearson chi-square, $\chi^2(28) = 3422.5$, as the result of chisq.test(suicide.tab2). However, the scree plot display above shows that the association between age and method is essentially one-dimensional, but note also that dimension 1 ("age–method") in this analysis has nearly the same inertia (0.0604) as the second dimension (0.0596) in the analysis of the stacked table. We plot the CA results as shown below (see Figure 6.8), and add a line connecting the supplementary points for sex.

```
> res <- plot(suicide.ca2s, pch = c(16, 15, 17, 24))
> lines(res$rows[6 : 7,])
```

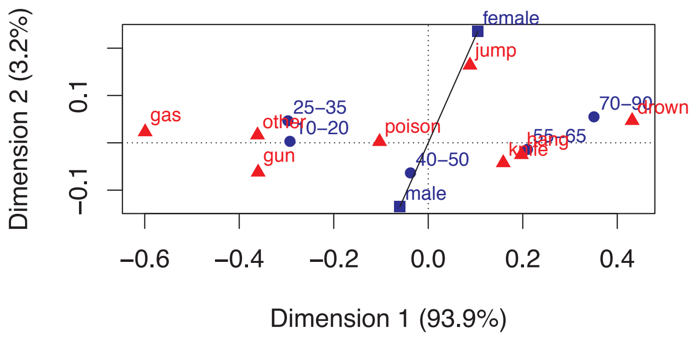


Figure 6.8: 2D CA solution for the [Age] [Method] marginal table. Category points for Sex are shown as supplementary points.

Comparing this graph with Figure 6.6, you can see that ignoring sex has collapsed the differences between males and females, which were the dominant feature of the analysis including sex. The dominant feature in Figure 6.8 is the Dimension 1 ordering of both age and method. However, as in Figure 6.6, the supplementary points for sex point toward the methods that are more prevalent for females and males.

6.4 Multiple correspondence analysis

Multiple correspondence analysis (MCA) is designed to display the relationships of the categories of two or more discrete variables, but it is best used for multiway tables where the extensions of classical CA described in Section 6.3 do not suffice. Again, this is motivated by the desire to provide an *optimal scaling* of categorical variables, giving scores for the discrete variables in an n-way table with desirable properties, and which can be plotted to visualize the relations among the category points.

The most typical development of MCA starts by defining indicator ("dummy") variables for each category and reexpresses the n-way contingency table in the form of a cases-by-variables indicator matrix, $\boldsymbol{Z}$. Simple correspondence analysis for a two-way table can, in fact, be derived as the canonical correlation analysis of the indicator matrix.

Unfortunately, the generalization to more than two variables follows a somewhat different path, so that simple CA does not turn out to be precisely a special case of MCA in some respects, particularly in the decomposition of an interpretable χ^2 over the dimensions in the visual representation.

Nevertheless, MCA does provide a useful graphic portrayal of the *bivariate* relations among any number of categorical variables, and has close relations to the mosaic matrix (Section 5.6). If its limitations are understood, it is helpful in understanding large, multivariate categorical data sets, in a similar way to the use of scatterplot matrices and dimension-reduction techniques (e.g., principal component analysis) for quantitative data.

6.4.1 Bivariate MCA

For the hair color–eye color data, the indicator matrix $\boldsymbol{Z}$ has 592 rows and $4 + 4 = 8$ columns. The columns refer to the eight categories of hair color and eye color and the rows to the 592 students in Snee's 1974 sample.

For simplicity, we show the calculation of the indicator matrix below in frequency form, using `model.matrix()` to compute the dummy (0/1) variables for the levels of hair color (`Hair1`–`Hair4`) and eye color (`Eye1`–`Eye4`).

```
> haireye.df <- cbind(
+     as.data.frame(haireye),
+     model.matrix(Freq ~ Hair + Eye, data=haireye,
+         contrasts.arg=list(Hair=diag(4), Eye=diag(4)))[,-1]
+     )
> haireye.df

    Hair   Eye Freq Hair1 Hair2 Hair3 Hair4 Eye1 Eye2 Eye3 Eye4
1  Black Brown   68     1     0     0     0    1    0    0    0
2  Brown Brown  119     0     1     0     0    1    0    0    0
3    Red Brown   26     0     0     1     0    1    0    0    0
4  Blond Brown    7     0     0     0     1    1    0    0    0
5  Black  Blue   20     1     0     0     0    0    1    0    0
6  Brown  Blue   84     0     1     0     0    0    1    0    0
7    Red  Blue   17     0     0     1     0    0    1    0    0
8  Blond  Blue   94     0     0     0     1    0    1    0    0
9  Black Hazel   15     1     0     0     0    0    0    1    0
10 Brown Hazel   54     0     1     0     0    0    0    1    0
```

```
11    Red Hazel    14      0      0      1      0      0      0      1      0
12  Blond Hazel    10      0      0      0      1      0      0      1      0
13  Black Green     5      1      0      0      0      0      0      0      1
14  Brown Green    29      0      1      0      0      0      0      0      1
15    Red Green    14      0      0      1      0      0      0      0      1
16  Blond Green    16      0      0      0      1      0      0      0      1
```

Thus, the first row in `haireye.df` represents the 68 individuals having black hair (`Hair1=1`) and brown eyes (`Eye1=1`). The indicator matrix $\boldsymbol{Z}$ is then computed by replicating the rows in `haireye.df` according to the `Freq` value, using the function `expand.dft`. The result has 592 rows and 8 columns.

```
> Z <- expand.dft(haireye.df)[,-(1:2)]
> vnames <- c(levels(haireye.df$Hair), levels(haireye.df$Eye))
> colnames(Z) <- vnames
> dim(Z)

[1] 592   8
```

Note that if the indicator matrix is partitioned as $\boldsymbol{Z} = [\boldsymbol{Z}_1, \boldsymbol{Z}_2]$, corresponding to the two sets of categories, then the contingency table is given by $\boldsymbol{N} = \boldsymbol{Z}_1^{\mathsf{T}} \boldsymbol{Z}_2$.

```
> (N <- t(as.matrix(Z[,1:4])) %*% as.matrix(Z[,5:8]))

         Brown Blue Hazel Green
Black      68   20    15     5
Brown     119   84    54    29
Red        26   17    14    14
Blond       7   94    10    16
```

With this setup, MCA can be described as the application of the simple correspondence analysis algorithm to the indicator matrix $\boldsymbol{Z}$. This analysis would yield scores for the rows of $\boldsymbol{Z}$ (the cases), usually not of direct interest, and for the columns (the categories of both variables). As in simple CA, each row point is the weighted average of the scores for the column categories, and each column point is the weighted average of the scores for the row observations.[4]

Consequently, the point for any category is the centroid of all the observations with a response in that category, and all observations with the same response pattern coincide. As well, the origin reflects the weighted average of the categories for *each* variable. As a result, category points with low marginal frequencies will be located further away from the origin, while categories with high marginal frequencies will be closer to the origin. For a binary variable, the two category points will appear on a line through the origin, with distances inversely proportional to their marginal frequencies.

Example 6.7: Hair color and eye color

For expository purposes, we illustrate the analysis of the indicator matrix below for the hair color–eye color data using `ca()`, rather than the function `mjca()`, which is designed for a more general approach to MCA.

```
> Z.ca <- ca(Z)
> res <- plot(Z.ca, what = c("none", "all"))
```

In the call to `plot.ca`, the argument `what` is used to suppress the display of the row points for the cases. The plot shown in Figure 6.9 is an enhanced version of this basic plot.

[4]Note that, in principle, this use of an indicator matrix could be extended to three (or more) variables. That extension is more easily described using an equivalent form, the ***Burt matrix***, described in Section 6.4.2.

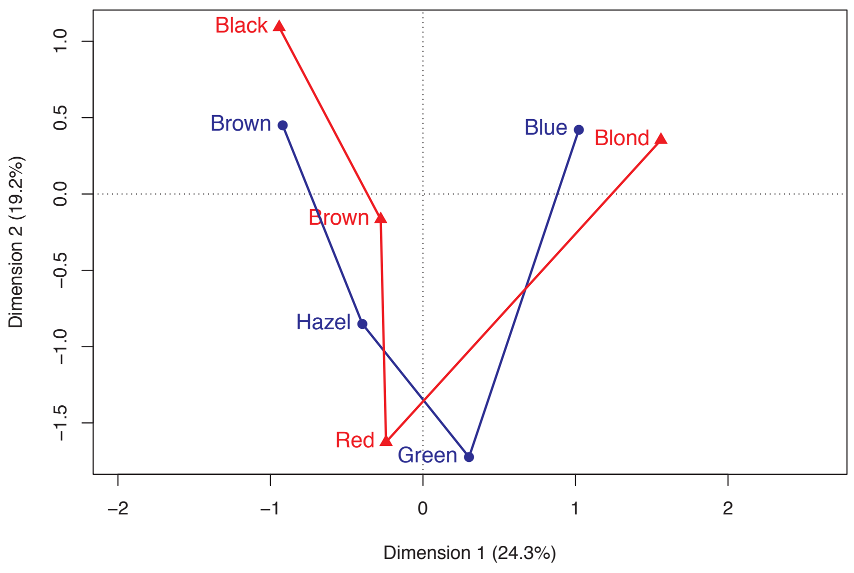


Figure 6.9: Correspondence analysis of the indicator matrix Z for the hair color–eye color data. The category points are joined separately by lines for the hair color and eye color categories.

```
        Dim1      Dim2  factor  levels
1 -0.94250  1.09220    Hair   Black
2 -0.27693 -0.16608    Hair   Brown
3 -0.24194 -1.62513    Hair     Red
4  1.56039  0.35376    Hair   Blond
5 -0.91933  0.44905     Eye   Brown
6  1.02254  0.42176     Eye    Blue
7 -0.39712 -0.85105     Eye   Hazel
8  0.30215 -1.72375     Eye   Green
```

Comparing Figure 6.9 with Figure 6.1, we see that the general pattern of the hair color and eye color categories is the same in the analysis of the contingency table (Figure 6.1) and the analysis of the indicator matrix (Figure 6.9), except that the axes are scaled differently—the display has been stretched along the second (vertical) dimension. The interpretation is the same: Dimension 1 reflects a dark–light ordering of both hair and eye colors, and Dimension 2 reflects something that largely distinguishes red hair and green eyes from the other categories.

Indeed, it can be shown (Greenacre, 1984, 2007) that the two displays are identical, except for changes in scales along the axes. There is no difference at all between the displays in standard coordinates. Greenacre (1984, pp. 130–134) describes the precise relations between the geometries of the two analyses.

$\triangle$

Aside from the largely cosmetic difference in relative scaling of the axes, a major difference between analysis of the contingency table and analysis of the indicator matrix is in the decomposition of principal inertia and corresponding χ^2 contributions for the dimensions. The plot axes in Figure 6.9 indicate 24.3% and 19.2% for the contributions of the two dimensions, whereas Figure 6.1 shows 89.4% and 9.5%. This difference is the basis for the more general development of MCA methods and is reflected in the `mcja()` function illustrated later in this chapter. But first, we describe a second approach to extending simple CA to the multivariate case based on the ***Burt matrix***.

6.4.2 The Burt matrix

The same solution for the category points as in the analysis of the indicator matrix may be obtained more simply from the so-called ***Burt matrix*** (Burt, 1950),

$$\boldsymbol{B} = \boldsymbol{Z}^\mathsf{T}\boldsymbol{Z} = \begin{bmatrix} \boldsymbol{N}_1 & \boldsymbol{N} \\ \boldsymbol{N}^\mathsf{T} & \boldsymbol{N}_2 \end{bmatrix},$$

where $\boldsymbol{N}_1$ and $\boldsymbol{N}_2$ are diagonal matrices containing the marginal frequencies of the two variables (the column sums of $\boldsymbol{Z}_1$ and $\boldsymbol{Z}_2$). In this representation, the contingency table of the two variables, $\boldsymbol{N}$, appears in the off-diagonal block in this equation. This calculation is shown below.

```
> Burt <- t(as.matrix(Z)) %*% as.matrix(Z)
> rownames(Burt) <- colnames(Burt) <- vnames
> Burt

       Black Brown Red Blond Brown Blue Hazel Green
Black    108     0   0     0    68   20    15     5
Brown      0   286   0     0   119   84    54    29
Red        0     0  71     0    26   17    14    14
Blond      0     0   0   127     7   94    10    16
Brown     68   119  26     7   220    0     0     0
Blue      20    84  17    94     0  215     0     0
Hazel     15    54  14    10     0    0    93     0
Green      5    29  14    16     0    0     0    64
```

The standard coordinates from an analysis of the Burt matrix $\boldsymbol{B}$ are identical to those of $\boldsymbol{Z}$. (However, the singular values of $\boldsymbol{B}$ are the squares of those of $\boldsymbol{Z}$.) Then, the following code, using the `Burt` matrix produces the same display of the category points for hair color and eye color as shown for the indicator matrix `Z` in Figure 6.9.

```
> Burt.ca <- ca(Burt)
> plot(Burt.ca)
```

6.4.3 Multivariate MCA

The coding of categorical variables in an indicator matrix and the relationship to the Burt matrix provides a direct and natural way to extend this analysis to more than two variables. If there are Q categorical variables, and variable q has J_q categories, then the Q-way contingency table, of size $J = \prod_{q=1}^{Q} J_q = J_1 \times J_2 \times \cdots \times J_Q$, with a total of $n = n_{++\cdots}$ observations, may be represented by the partitioned $(n \times J)$ indicator matrix $[\boldsymbol{Z}_1\, \boldsymbol{Z}_2\, \ldots\, \boldsymbol{Z}_Q]$.

Then the Burt matrix is the symmetric partitioned matrix

$$\boldsymbol{B} = \boldsymbol{Z}^\mathsf{T}\boldsymbol{Z} = \begin{bmatrix} \boldsymbol{N}_1 & \boldsymbol{N}_{12} & \cdots & \boldsymbol{N}_{1Q} \\ \boldsymbol{N}_{21} & \boldsymbol{N}_2 & \cdots & \boldsymbol{N}_{2Q} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{N}_{Q1} & \boldsymbol{N}_{Q2} & \cdots & \boldsymbol{N}_Q \end{bmatrix}, \tag{6.7}$$

where again the diagonal blocks $\boldsymbol{N}_i$ contain the one-way marginal frequencies. The off-diagonal blocks $\boldsymbol{N}_{ij}$ contain the bivariate marginal contingency tables for each pair (i, j) of variables.

Classical MCA (see, e.g., Greenacre (1984), Gower and Hand (1996)) can then be defined as a singular value decomposition of the matrix $\boldsymbol{B}$, which produces scores for the categories of *all* variables so that the greatest proportion of the bivariate, pairwise associations in all blocks (including the diagonal blocks) is accounted for in a small number of dimensions.

In this respect, MCA resembles multivariate methods for quantitative data based on the joint bivariate correlation or covariance matrix ($\boldsymbol{\Sigma}$) and there is some justification for regarding the Burt matrix as the categorical analog of $\boldsymbol{\Sigma}$.[5]

There is a close connection between this analysis and the bivariate mosaic matrix (Section 5.6): The mosaic matrix displays the residuals from independence for each pair of variables, and thus provides a visual representation of the Burt matrix. The one-way margins shown (by default) in the diagonal cells reflect the diagonal matrices $\boldsymbol{N}_i$ in Eqn. (6.7). The total amount of shading in all the individual mosaics portrays the total pairwise associations decomposed by MCA. See Friendly (1999a) for further details.

For interpretation of MCA plots, we note the following relations (Greenacre, 1984, Section 5.2):[6]

- The inertia contributed by a given variable increases with the number of response categories.
- The centroid of the categories for each discrete variable is at the origin of the display.
- For a particular variable, the inertia contributed by a given category increases as the marginal frequency in that category *decreases*. Low frequency points therefore appear further from the origin.
- The category points for a binary variable lie on a line through the origin. The distance of each point to the origin is inversely related to the marginal frequency.

Example 6.8: Marital status and pre- and extramarital sex

The data on the relation between marital status and reported premarital and extramarital sex was explored earlier using mosaic displays in Example 5.9 and Example 5.13.

Using the ca package, an MCA analysis of the `PreSex` data is carried out using `mjca()`. This function typically takes a data frame in *case form* containing the factor variables, but converts a table to this form. This example analyzes the Burt matrix calculated from the `PreSex` data, specified as `lambda="Burt"`

```
> data("PreSex", package = "vcd")
> PreSex <- aperm(PreSex, 4:1)    # order variables G, P, E, M
> presex.mca <- mjca(PreSex, lambda = "Burt")
> summary(presex.mca)


Principal inertias (eigenvalues):

 dim    value      %    cum%   scree plot
 1      0.149930  53.6   53.6  *************
 2      0.067201  24.0   77.6  ******
 3      0.035396  12.6   90.2  ***
 4      0.027365   9.8  100.0  **
        -------- -----
 Total: 0.279892 100.0
...
```

The output from `summary()` seems to show that 77.6% of the total inertia is accounted for in two dimensions. A basic, default plot of the MCA solution is provided by the `plot()` method for "mjca" objects.

[5]For multivariate normal data, however, the mean vector and covariance matrix are sufficient statistics, so all higher-way relations are captured in the covariance matrix. This is not true of the Burt matrix. Moreover, the covariance matrix is typically expressed in terms of mean-centered variables, while the Burt matrix involves the marginal frequencies. A more accurate statement is that the uncentered covariance matrix is analogous to the Burt matrix.

[6]This book, now out of print, is available for free download at `http://www.carme-n.org/`.

```
> plot(presex.mca)
```

This plotting method is not very flexible in terms of control of graphical parameters or the ability to add additional annotations (labels, lines, legend) to ease interpretation. Instead, we use the plot method to create an empty plot (with no points or labels), and return the calculated plot coordinates (`res`) for the categories. A bit of processing of the coordinates provides the customized display shown in Figure 6.10.

```
> # plot, but don't use point labels or points
> res <- plot(presex.mca, labels = 0, pch = ".", cex.lab = 1.2)
>
> # extract factor names and levels
> coords <- data.frame(res$cols, presex.mca$factors)
> nlev <- presex.mca$levels.n
> fact <- unique(as.character(coords$factor))
>
> cols <- c("blue", "red", "brown", "black")
> points(coords[,1:2], pch=rep(16:19, nlev), col=rep(cols, nlev), cex=1.2)
> text(coords[,1:2], label=coords$level, col=rep(cols, nlev), pos=3,
+       cex=1.2, xpd=TRUE)
> lwd <- c(2, 2, 2, 4)
> for(i in seq_along(fact)) {
+   lines(Dim2 ~ Dim1, data = coords, subset = factor==fact[i],
+          lwd = lwd[i], col = cols[i])
+ }
>
> legend("bottomright",
+         legend = c("Gender", "PreSex", "ExtraSex", "Marital"),
+         title = "Factor", title.col = "black",
+         col = cols, text.col = cols, pch = 16:19,
+         bg = "gray95", cex = 1.2)
```

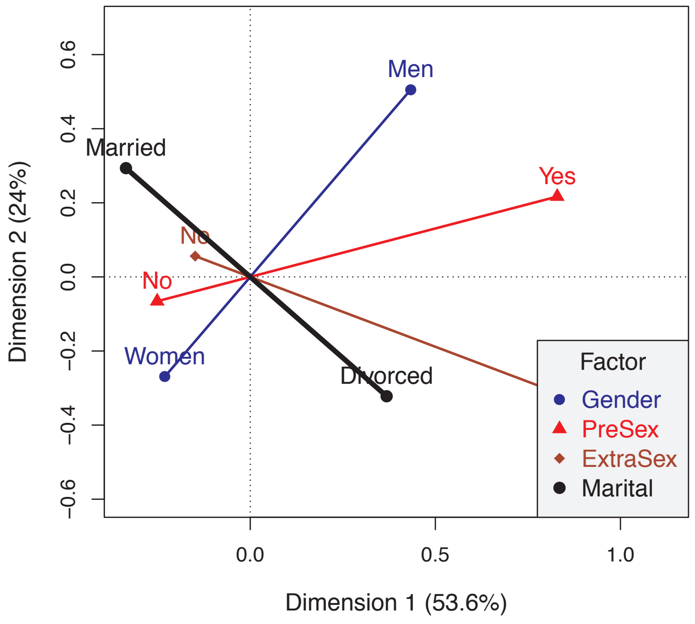


Figure 6.10: MCA plot of the Burt matrix for the PreSex data. The category points are joined separately by lines for the factor variables.

As indicated above, the category points for each factor appear on lines through the origin, with distances inversely proportional to their marginal frequencies. For example, the categories for No premarital and extramarital sex are much larger than the corresponding Yes categories, so the former are positioned closer to the origin. In contrast, the categories of gender and marital status are more nearly equal marginally.

Another aspect of interpretation of Figure 6.10 concerns the alignment of the lines for different factors. The positions of the category points on Dimension 1 suggest that women are less likely to have had pre-marital and extra-marital sex and that still being married is associated with the absence of pre- and extra-marital sex. As well, the lines for gender and marital status are nearly at right angles, suggesting that these variables are unassociated. This interpretation is more or less correct, but it is only approximate in this MCA scaling of the coordinate axes. An alternative scaling, based on a ***biplot*** representation is described in Section 6.5.

If you compare the MCA result in Figure 6.10 with the mosaic matrix in Figure 5.23, you will see that they are both showing the bivariate pairwise associations among these variables, but in different ways. The mosaic plots show the details of marginal and joint frequencies together with residuals from independence for each 2×2 marginal subtable. The MCA plot using the Burt matrix summarizes each category point in terms of a 2D representation of contributions to total inertia (association). △

6.4.3.1 Inertia decomposition

The transition from simple CA to MCA is straightforward in terms of the category scores derived from the indicator matrix $\boldsymbol{Z}$ or the Burt matrix, $\boldsymbol{B}$. It is less so in terms of the calculation of total inertia, and therefore in the chi-square values and corresponding percentages of association accounted for in some number of dimensions.

In simple CA, the total inertia is χ^2/n, and it therefore makes sense to talk of percentage of association accounted for by each dimension. But in MCA of the indicator matrix, the total inertia, $\sum \lambda^2$, is simply $(J - Q)/Q$, because the inertia of each subtable, $\boldsymbol{Z}_i$, is equal to its dimensionality, $J_i - 1$, and the total inertia of an indicator matrix is the average of the inertias of its subtables. Consequently, the average inertia per dimension is $1/Q$, and it is common to interpret only those dimensions that exceed this average (analogous to the use of 1 as a threshold for eigenvalues in principal components analysis).

To more adequately reflect the percentage of association in MCA, Greenacre (1990), revising an earlier proposal by Benzécri (1977), suggested the calculation of ***adjusted inertia***, which ignores the contributions of the diagonal blocks in the Burt matrix,

$$(\lambda_i^\star)^2 = \left[\frac{Q}{Q-1} (\lambda_i^Z - \frac{1}{Q}) \right]^2 \tag{6.8}$$

as the principal inertia due to the dimensions with $(\lambda^Z)^2 > 1/Q$. This adjustment expresses the contribution of each dimension as $(\lambda_i^\star)^2 / \sum (\lambda_i^\star)^2$, with the summation over only dimensions with $(\lambda^Z)^2 > 1/Q$.

A related method, also handled by `mjca()`, is ***joint correspondence analysis*** (Greenacre, 1994, Greenacre, 2007, Chapter 19), an iterative method that replaces the diagonal blocks of the Burt matrix with values that minimize their impact on inertia. Unlike MCA, solutions in JCA are not nested, however.

Example 6.9: Survival on the *Titanic*

An MCA analysis of the `Titanic` data is carried out using `mjca()` as shown below.

```
> titanic.mca <- mjca(Titanic)
```

mjca() allows different scaling methods for the contributions to inertia of the different dimensions. The default (lambda="adjusted"), used here, is the adjusted inertias as in Eqn. (6.8).

```
> summary(titanic.mca)


Principal inertias (eigenvalues):

 dim    value       %    cum%   scree plot
 1      0.067655  76.8  76.8   **********************
 2      0.005386   6.1  82.9   **
 3      00000000   0.0  82.9
        -------- -----
 Total: 0.088118
...
```

Using similar code to that used in Example 6.8, Figure 6.11 shows an enhanced version of the default plot that connects the category points for each factor by lines using the result returned by the plot() function.

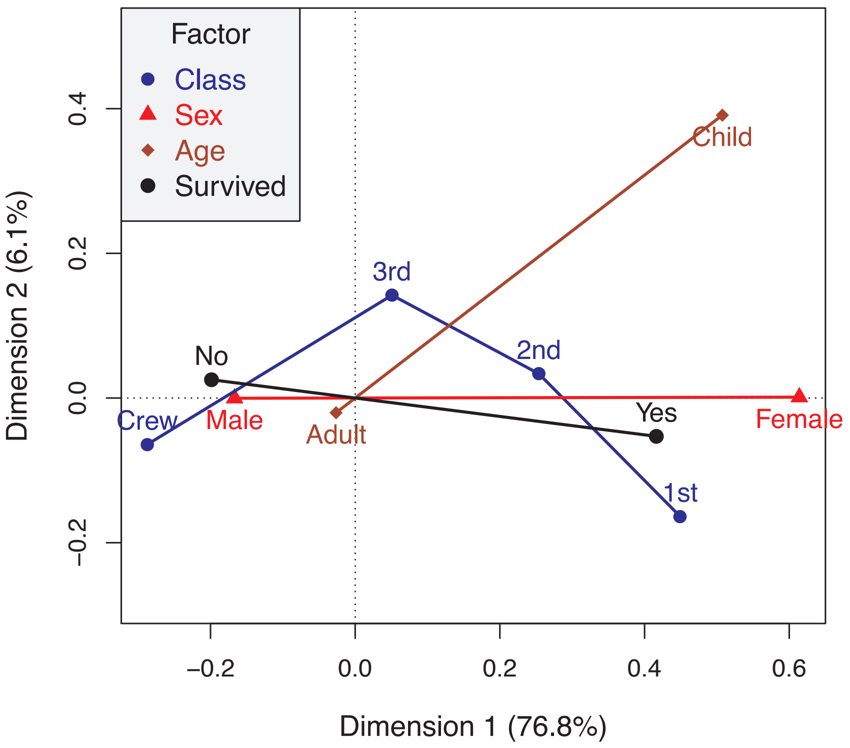


Figure 6.11: MCA plot of the Titanic data. The category points are joined separately by lines for the factor variables.

In this plot, the points for each factor have the property that the sum of coordinates on each dimension, weighted inversely by the marginal proportions, equals zero. Thus high-frequency categories (e.g., Adult and Male) are close to the origin.

The first dimension is perfectly aligned with gender, and also strongly aligned with Survival. The second dimension pertains mainly to Class and Age effects. Consider those points that differ from the origin most similarly (in distance and direction) to the point for Survived ("Yes"); this gives the interpretation that survival was associated with being female or upper class or (to a lesser degree) being a child.

△

6.5 Biplots for contingency tables

Like correspondence analysis, the ***biplot*** (Bradu and Gabriel, 1978, Gabriel, 1971, 1980, 1981, Gower et al., 2011) is a visualization method that uses the SVD to display a matrix in a low-dimensional (usually 2-dimensional) space. They differ in the relationships in the data that are portrayed, however:

- In correspondence analysis the (weighted, χ^2) *distances* between row points and distances between column points are designed to reflect *differences* between the row profiles and column profiles.

- In the biplot, on the other hand, row and column points are represented by *vectors* from the origin such that the projection (inner product) of the vector $\boldsymbol{a}_i$ for row i on $\boldsymbol{b}_j$ for column j approximates the data element y_{ij},

$$\boldsymbol{Y} \approx \boldsymbol{A}\boldsymbol{B}^\mathsf{T} \Longleftrightarrow y_{ij} \approx \boldsymbol{a}_i^\mathsf{T}\boldsymbol{b}_j \; . \tag{6.9}$$

Geometrically, Eqn. (6.9) may be described as approximating the data value y_{ij} by the projection of the end point of vector $\boldsymbol{a}_i$ on $\boldsymbol{b}_j$ (and vice-versa), as shown in Figure 6.12.

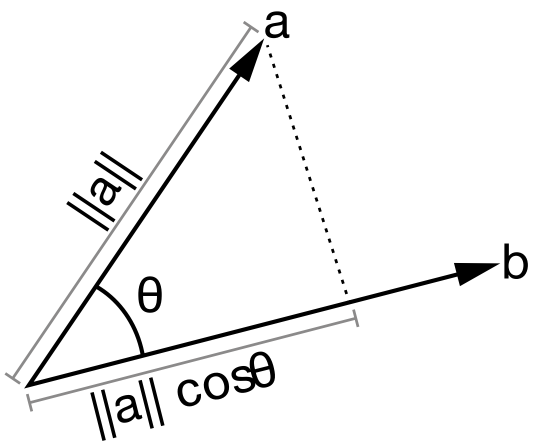


Figure 6.12: The scalar product of vectors of two points from the origin is the length of the projection of one vector on the other.

6.5.1 CA bilinear biplots

As in CA, there are a number of different representations of coordinates for row and column points for a contingency table within a biplot framework. One set of connections between CA and the biplot can be seen through the *reconstitution formula*, giving the decomposition of the correspondence matrix $\boldsymbol{P} = \boldsymbol{N}/n$ in terms of the standard coordinates $\boldsymbol{\Phi}$ and $\boldsymbol{\Gamma}$, defined in Eqn. (6.4) and Eqn. (6.5) as:

$$p_{ij} = r_i c_j \left(1 + \sum_{m=1}^{M} \sqrt{\lambda_m} \phi_{im} \gamma_{jm} \right) , \tag{6.10}$$

or, in matrix terms,

$$\boldsymbol{P} = \boldsymbol{D}_r (\mathbf{1}\mathbf{1}^\mathsf{T} + \boldsymbol{\Phi} \boldsymbol{D}_\lambda^{1/2} \Gamma^\mathsf{T}) \boldsymbol{D}_c \, . \tag{6.11}$$

The CA solution approximates this by a sum over $d \ll M$ dimensions, or by using only the first d (usually 2) columns of $\boldsymbol{\Phi}$ and $\boldsymbol{\Gamma}$.

Eqn. (6.10) can be re-written in biplot scalar form as

$$\left(\frac{p_{ij}}{r_i c_j} \right) - 1 \approx \sum_{m=1}^{d} (\sqrt{\lambda_m} \phi_{im}) \gamma_{jm} = \sum_{m=1}^{d} f_{im} \gamma_{jm} \tag{6.12}$$

where $f_{im} = (\sqrt{\lambda_m} \phi_{im})$ gives the principal coordinates of the row points. The left-hand side of Eqn. (6.12) contains the ***contingency ratios***, $p_{ij}/r_i c_j$, of the observed cell probabilities to their expected values under independence. This shows that an ***asymmetric CA plot*** of row principal coordinates $\boldsymbol{F}$ and the column standard coordinates $\boldsymbol{\Gamma}$ is a biplot that approximates the deviations of the contingency ratios from their values under independence.

In the ca package, this plot is obtained by specifying `map="rowprincipal"` in the call to `plot()`, or `map="colprincipal"` to plot the column points in principal coordinates. It is typical in such biplots to display one set of coordinates as points and the other as vectors from the origin, as controlled by the `arrows` argument, so that one can interpret the data values represented as approximated by the projections of the points on the vectors.

Two other types of asymmetric "maps" are also defined with different scalings that turn out to have better visual properties in terms of representing the relations between the row and column categories, particularly when the strength of association (inertia) in the data is low.

- The option `map="rowgab"` (or `map="colgab"`) gives a biplot form proposed by Gabriel and Odoroff (1990) with the rows (columns) shown in principal coordinates and the columns (rows) in standard coordinates multiplied by the mass c_j (r_i) of the corresponding point.
- The *contribution biplot* for CA (Greenacre, 2013), with the option `map="rowgreen"` (or `map="colgreen"`) provides a reconstruction of the standardized residuals from independence, using the points in standard coordinates multiplied by the square root of the corresponding masses. This has the nice visual property of showing more directly the contributions of the vectors to the low-dimensional solution.

Example 6.10: Suicide rates in Germany — biplot

To illustrate the biplot representation, we continue with the data on suicide rates in Germany from Example 6.5, using the stacked table `suicide.tab` comprised of the age–sex combinations as rows and methods of suicide as columns.

```
> suicide.tab <- xtabs(Freq ~ age_sex + method2, data = Suicide)
> suicide.ca <- ca(suicide.tab)
```

Using this result, `suicide.ca`, in the call to `plot()` below, we use `map="colgreen"`, and vectors represent the methods of suicide, as shown in Figure 6.13.

```
> plot(suicide.ca, map = "colgreen", arrows = c(FALSE, TRUE))
```

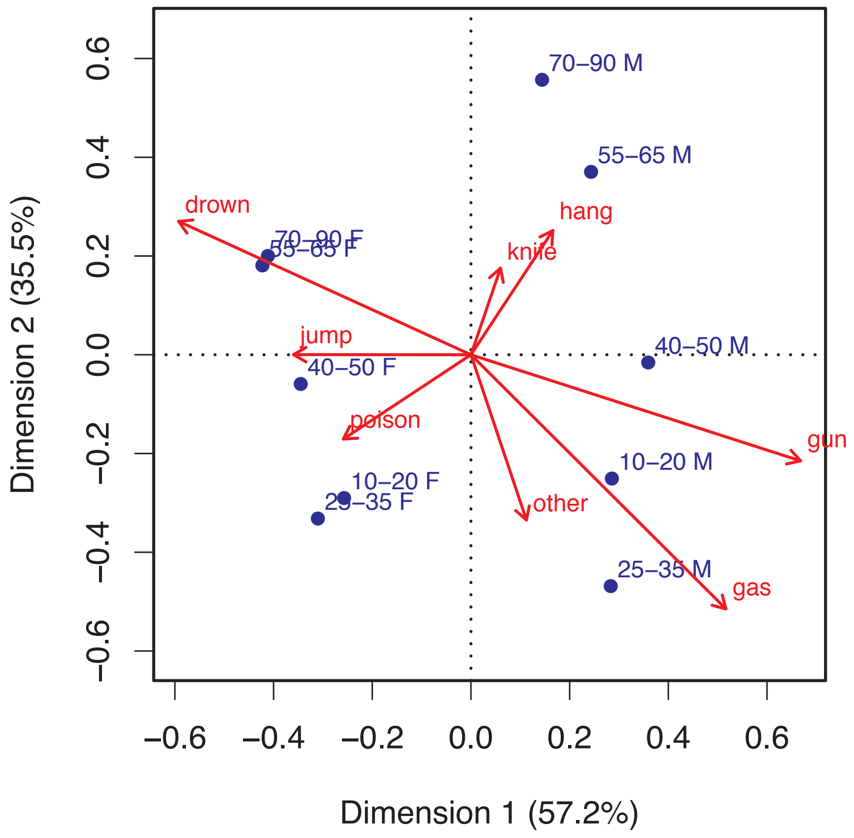


Figure 6.13: CA biplot of the suicide data using the contribution biplot scaling. Associations between the age–sex categories and the suicide methods can be read as the projections of the points on the vectors. The lengths of the vectors for the suicide categories reflect their contributions to this representation in a 2D plot.

The interpretation of the row points for the age–sex categories is similar to what we saw earlier in Figure 6.6. But now, the vectors for the suicide categories reflect the contributions of those methods to the representation of association. Thus, the methods `drown`, `gun`, and `gas` have large contributions, while `knife`, `hang`, and `poison` are relatively small. Moreover, the projections of the points for the age–sex combinations on the method vectors reflect the standardized residuals from independence.

The most comprehensive modern treatment of biplot methodology is the book *Understanding Biplots* (Gower et al., 2011). Together with the book, they provide an R package, UBbipl (le Roux and Lubbe, 2013), that is capable of producing an astounding variety of high-quality plots. Unfortunately, that package is only available on their publisher's web site,[7] and you need the book to be able to use it because all the documentation is in the book. Nevertheless, we illustrate the use of the `cabipl()` function to produce the version of the CA biplot shown in Figure 6.14.

[7]`http://www.wiley.com/legacy/wileychi/gower/material.html.`

```
> library(UBbipl)
> cabipl(as.matrix(suicide.tab),
+       axis.col = gray(.4), ax.name.size = 1,
+       ca.variant = "PearsonResA",
+       markers = FALSE,
+       row.points.size = 1.5,
+       row.points.col = rep(c("red", "blue"), 4),
+       plot.col.points = FALSE,
+       marker.col = "black", marker.size = 0.8,
+       offset = c(2, 2, 0.5, 0.5),
+       offset.m = rep(-0.2, 14),
+       output = NULL)
```

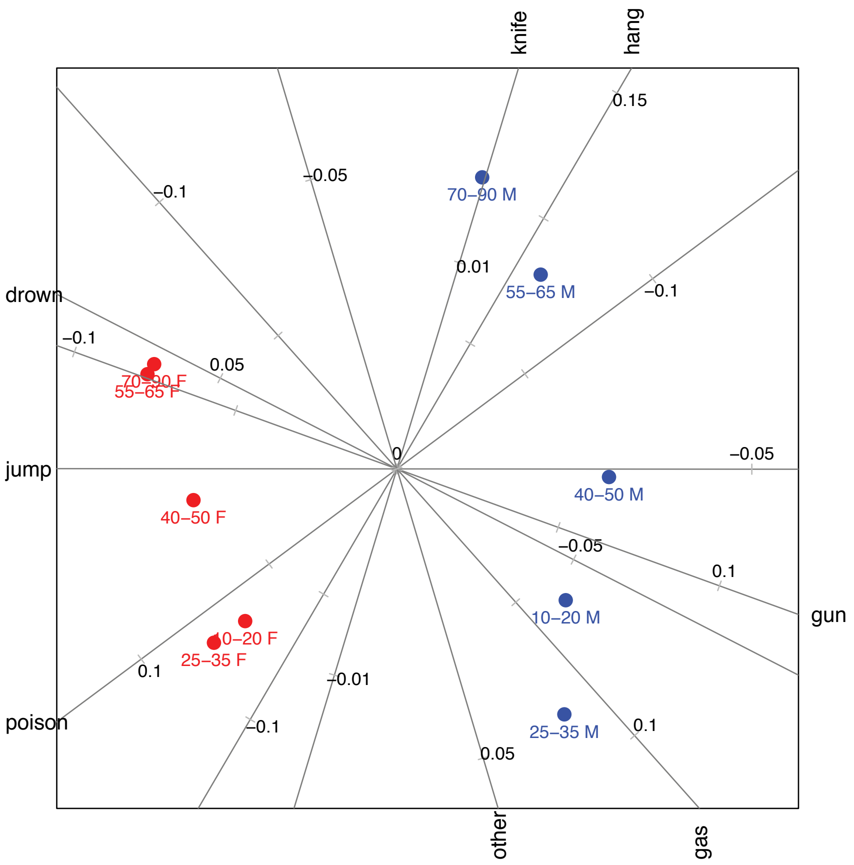


Figure 6.14: CA biplot of the suicide data, showing calibrated axes for the suicide methods.

This plot uses `ca.variant = "PearsonResA"` to specify that the biplot is to approximate the standardized Pearson residuals by the inner product of each row point on the vector for the column point for the suicide methods, as also in Figure 6.13. However, Figure 6.14 represents the methods as calibrated axis lines, designed to be read as scales for the projections of the row points (age–sex) on the methods. The UBbipl package has a huge number of options for controlling the details of the biplot display. See Gower et al. (2011, Ch. 2) for all the details.

△