

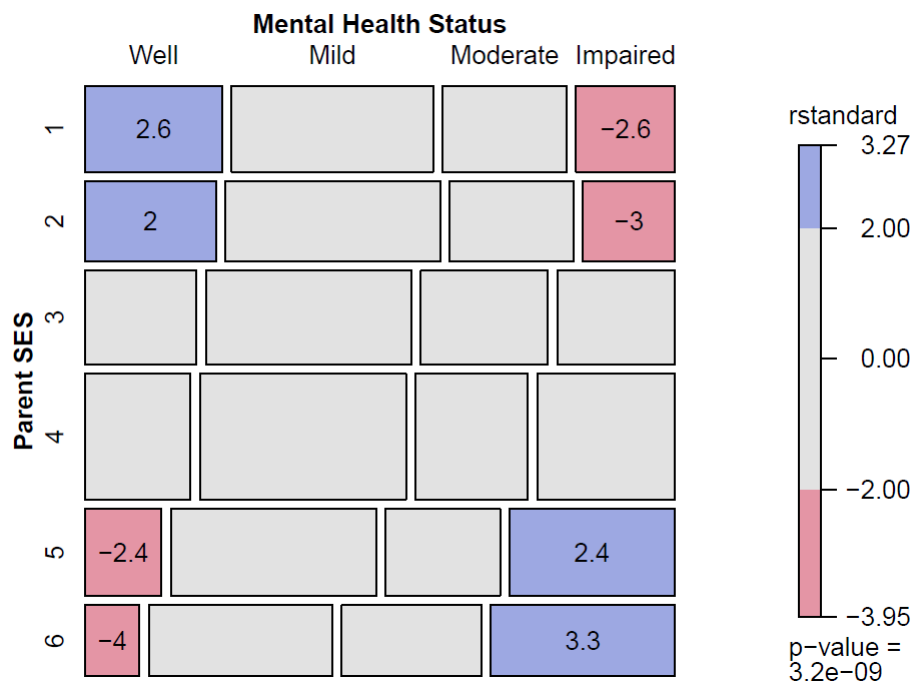
Visualizing Categorical Data with R

Michael Friendly
York University

David Meyer
Wirtschaftsuniversität Wien

with contributions,
Achim Zeileis
Universität Innsbruck

February 23, 2014



Contents

Table of Contents	i
1 Introduction	1
1.1 Data visualization and categorical data: Overview	1
1.2 What is categorical data?	2
1.2.1 Case form vs. frequency form	3
1.2.2 Frequency data vs. count data	4
1.2.3 Univariate, bivariate, and multivariate data	5
1.2.4 Explanatory vs. Response variables	5
1.3 Strategies for categorical data analysis	6
1.3.1 Hypothesis testing approaches	6
1.3.2 Model building approaches	8
1.4 Graphical methods for categorical data	9
1.4.1 Goals and design principles for visual data display	10
1.4.2 Categorical data require different graphical methods	12
1.5 Visualization = Graphing + Fitting + Graphing	12
1.6 Further reading	13
2 Working with categorical data	15
2.1 Working with R data: vectors, matrices, arrays and data frames	15
2.1.1 Vectors	16
2.1.2 Matrices	17
2.1.3 Arrays	19
2.1.4 data frames	20
2.2 Forms of categorical data: case form, frequency form and table form	22
2.2.1 Case form	23
2.2.2 Frequency form	23
2.2.3 Table form	24
2.3 Ordered factors and reordered tables	26
2.4 Generating tables: table and xtabs	28
2.4.1 xtabs()	29
2.5 Printing tables: structable and ftable	30
2.5.1 Publishing tables to L ^A T _E X or HTML	33
2.6 Collapsing over table factors	35
2.6.1 Collapsing table levels	36
2.7 Converting among frequency tables and data frames	37
2.7.1 Table form to frequency form	37
2.7.2 Case form to table form	39
2.7.3 Table form to case form	40
2.8 A complex example	40

2.8.1	Creating data frames and arrays	41
2.8.2	Subsetting and collapsing	42
2.9	Further reading	44
2.10	Lab exercises	44
3	Fitting and graphing discrete distributions	47
3.1	Introduction to discrete distributions	48
3.1.1	Binomial data	48
3.1.2	Poisson data	51
3.1.3	Type-token distributions	54
3.2	Characteristics of discrete distributions	55
3.2.1	The binomial distribution	56
3.2.2	The Poisson distribution	59
3.2.3	The negative binomial distribution	65
3.2.4	The geometric distribution	69
3.2.5	The logarithmic series distribution	69
3.2.6	Power series family	70
3.3	Fitting discrete distributions	71
3.3.1	R tools for discrete distributions	72
3.3.2	Plots of observed and fitted frequencies	76
3.4	Diagnosing discrete distributions: Ord plots	79
3.5	Poissonness plots and generalized distribution plots	84
3.5.1	Features of the Poissonness plot	84
3.5.2	Plot construction	84
3.5.3	The distplot function	85
3.5.4	Leverage and influence	87
3.5.5	Plots for other distributions	87
3.6	Fitting discrete distributions as generalized linear models	89
3.7	Chapter summary	92
3.8	Further reading	93
3.9	Lab exercises	93
4	Two-way contingency tables	97
4.1	Introduction	97
4.2	Tests of association for two-way tables	100
4.2.1	Notation and terminology	100
4.2.2	2 by 2 tables	102
4.2.3	Larger tables: Overall analysis	104
4.2.4	Tests for ordinal variables	105
4.2.5	Sample CMH Profiles	106
4.3	Stratified analysis	108
4.3.1	Assessing homogeneity of association	109
4.4	Fourfold display for 2 x 2 tables	110
4.4.1	Confidence rings for odds ratio	113
4.4.2	fourfold() details	113
4.4.3	Stratified analysis for $2 \times 2 \times k$ tables	113
4.5	Sieve diagrams	118
4.5.1	Larger tables: The strucplot framework	120
4.6	Association plots	123
4.7	Observer agreement	126

4.7.1	Measuring agreement	127
4.7.2	Observer Agreement Chart	129
4.7.3	Observer bias in agreement	131
4.8	Trilinear plots	132
4.9	Chapter summary	137
4.10	Further reading	138
4.11	Lab exercises	138
5	Mosaic displays for n-way tables	139
5.1	Introduction	139
5.2	Two-way tables	140
5.3	The strucplot framework	145
5.3.1	Shading schemes	147
5.4	Three-way and larger tables	154
5.4.1	Fitting models	155
5.4.2	Sequential plots and models	160
5.4.3	Causal models	162
5.4.4	Partial association	165
5.5	Mosaic matrices for categorical data	173
5.5.1	Generalized mosaic matrices and pairs plots	178
5.6	3D mosaics	180
5.7	Parallel coordinate plots for categorical data	182
5.7.1	Parallel set plots: Hammock plots and common angle plots	185
5.8	Visualizing the structure of loglinear models	186
5.8.1	Mutual independence	188
5.8.2	Joint independence	190
5.9	Chapter summary	191
5.10	Further reading	192
5.11	Lab exercises	192
6	Correspondence analysis	193
6.1	Introduction	193
6.2	Simple correspondence analysis	194
6.2.1	Notation and terminology	194
6.2.2	Geometric and statistical properties	195
6.2.3	R software for correspondence analysis	196
6.3	Properties of category scores	202
6.3.1	Optimal category scores	203
6.3.2	Simultaneous linear regressions	203
6.4	Multi-way tables: Stacking and other tricks	203
6.4.1	Marginal tables and supplementary variables	209
6.5	Multiple correspondence analysis	210
6.5.1	Bivariate MCA	211
6.6	Extended MCA: Showing interactions in 2^Q tables	212
6.7	Biplots for contingency tables	212
6.8	Chapter summary	212
6.9	Further reading	212
6.10	Lab exercises	212
	References	213

Subject Index**221**

Chapter 1

Introduction

{ch:intro}

Categorical data consists 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

{sec:viscat}

Beauty is truth; truth, beauty.
That is all ye know on Earth, all ye need to know.

John Keats, *Ode on a Grecian urn*

“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 which 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” data 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 χ^2 , 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 ??–?? 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 1.5 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?

{sec:whatis}

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, ...

As these examples suggest, categorical variables differ in the number of categories: we often distinguish *binary variables* such as *Gender* from those with more than two categories (called *polytomous variables*). For example, Table 1.1 gives data on 4526 applicants to graduate departments at the University of California at Berkeley in 1973, classified by two binary variables, gender and admission status.

{tab:berk220}

Table 1.1: Admissions to Berkeley graduate programs

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

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

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.

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

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

{sec:case-freq}

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,

```
data("Arthritis", package="vcd")
head(Arthritis, 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
```

¹An 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.

²This may be wasteful 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 non-linear relations with the outcome.

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==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.

```
as.data.frame(xtabs(~Treatment+Sex+Improved, data=Arthritis))
```

##	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

{sec:freq-count}

In many cases the observations represent 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 observation tabulated 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, 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.

ab:saxdata}

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	1033	1343	1112	829	478	181	45	7

1.2.3 Univariate, bivariate, and multivariate data

:uni-multi}

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*.

```
as.data.frame(xtabs(~Treatment + Improved, data=Arthritis))

##      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 19th 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

{sec:exp-resp}

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.

However, 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 ??), 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

{sec:strategies}

Methods of analysis for categorical data can be classified into two broad categories: those concerned with hypothesis testing *per se*, and those concerned with model building.

1.3.1 Hypothesis testing approaches

{sec:strategies-hyp}

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, and tests for **homogeneity of association** help 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-square 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

Chapter 4–6, though the emphasis is on graphical methods which help to understand the nature of association between variables.

{ex:haireye0}

EXAMPLE 1.1: Hair color and eye color

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

```
library(vcd)
(HairEye <- margin.table(HairEyeColor, c(1, 2)))

##           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, the **phi coefficient**, $\phi = \sqrt{\chi^2/N} = 0.483$, indicates a substantial relationship between hair and eye color.

```
assocstats(HairEye)

##           X^2 df P(> X^2)
## Likelihood Ratio 146.44 9      0
## Pearson          138.29 9      0
##
## Phi-Coefficient   : 0.483
## 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.

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. \triangle

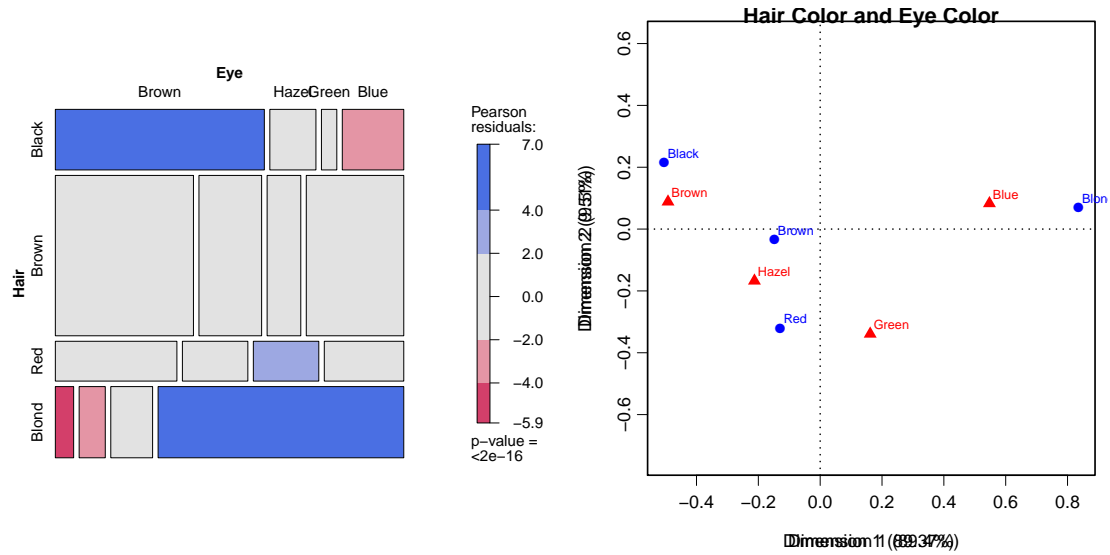


Figure 1.1: Graphical displays for the hair color and eye color data. Left: mosaic display; right: correspondence analysis plot.

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) 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, β_1, β_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.

{ex:nasa0}

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 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 ??.

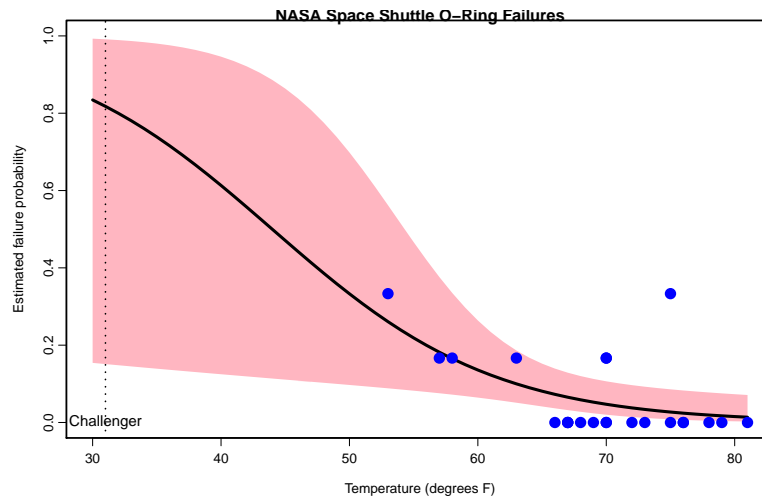


Figure 1.2: Space shuttle O-ring failure, observed and predicted probabilities. The dotted vertical line at 31° shows the prediction for the launch of the *Challenger*. fig:spaceshuttle

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.

When the *Challenger* was launched, the temperature was only 31° . The shaded region in Figure 1.2 show 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.

△

TODO: Perhaps replace this example with a similar one for the Donner data

1.4 Graphical methods for categorical data

{sec:methods}

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 ideas 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* package (Wickham, 2009, Wickham and Chang, 2013).

Another perspective on visual data display is presented in Section 1.4.1. 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.

1.4.1 Goals and design principles for visual data display

{sec:intro-goals}

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.

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 1.3 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 1.4 shows one organization of visualization methods in terms of the *primary* use or

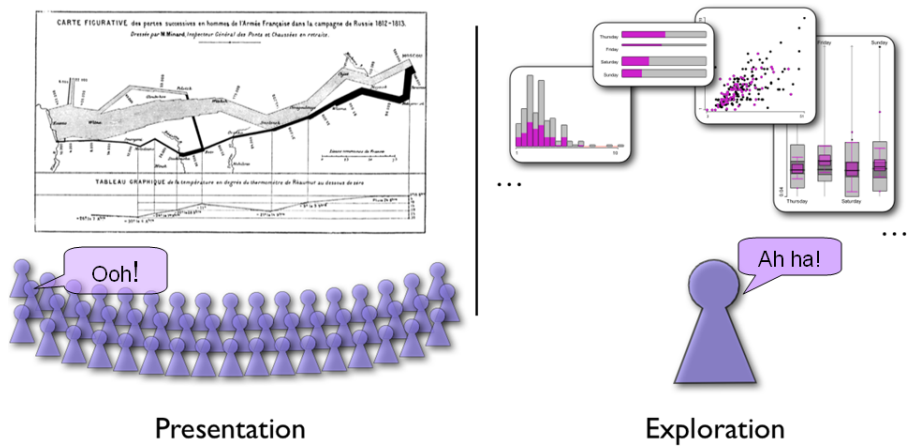


Figure 1.3: Different communication purposes require different graphs. For presentations, a single, carefully crafted graph may appeal best to a large audience; for exploratory analysis, many related images from different perspectives for a narrow audience (often you!). Source: Adapted from a blog entry by Martin Theus, <http://www.theusrus.de/blog/presentation-vs-exploration/>.

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

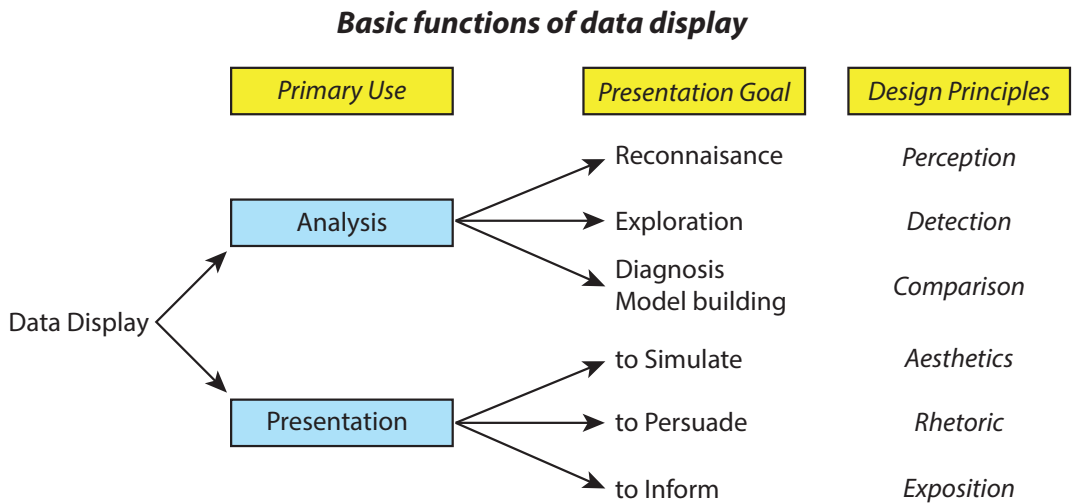


Figure 1.4: A taxonomy of the basic functions of data display by intended use, presentation goal and design principles.

TODO: Complete this section. Show a collection of analysis and presentation graphs for categorical data. It is probably better to wait until more chapters are written to provide examples here.

1.4.2 Categorical data require different graphical methods

{sec:intro-

We mentioned earlier, and will see in greater detail in Chapter ?? and Chapter ?? 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 and loglinear models. **TODO: Add forward references to these sections.**

Unhappily, the familiar techniques for displaying raw data are often disappointing when applied to categorical data. The simple scatterplot, for example, is widely used to show the relation between quantitative response and predictors, together with the fitted linear model.

TODO: Complete this section.

1.5 Visualization = Graphing + Fitting + Graphing

{sec:vis}

Look here, upon this picture, and on this.

Shakespeare, Hamlet

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

As shown below, 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.

```
UCB <- margin.table(UCBAdmissions, 1:2)
assocstats(UCB)

##              X^2  df P(> X^2)
## Likelihood Ratio 93.449   1      0
## Pearson          92.205   1      0
##
## Phi-Coefficient   : 0.143
## Contingency Coeff.: 0.141
## Cramer's V        : 0.143

oddsratio(UCB, log=FALSE)

## [1] 1.841
```

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. **TODO: Use this for a lab exercise in Ch 2.**

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.

1.6 Further reading

{sec:ch01-reading}

Chapter 2

Working with categorical data

{ch:working}

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.

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, 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.

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 graphical user interface (e.g., RStudio), 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 beyond such basic skills needed in the book.¹

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

{sec:Rdata}

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, logicals (`TRUE`, `FALSE`) or dates, 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

¹Some excellent introductory treatments of R are: Fox and Weisberg (2011, Chapter 2), ... 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.

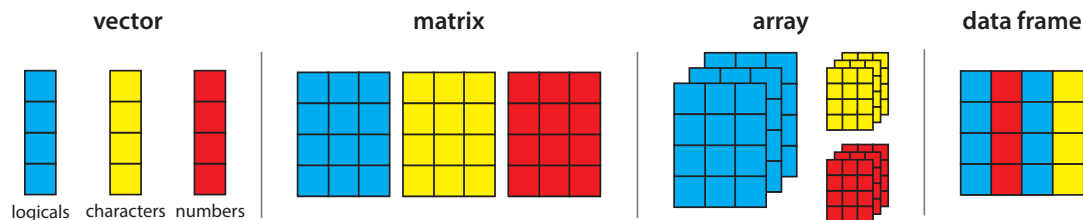


Figure 2.1: Principal data structures and data types in R.

{fig:dataty

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.

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()`, 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=)` where the optional argument `by` specifies the interval between adjacent values and `length` 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, to a given length or each a given multiple.

```
seq(10, 100, by=10)           # give interval
## [1] 10 20 30 40 50 60 70 80 90 100

seq(0, 1, length=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 takes the 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, I frequently 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])
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"
```

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.

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

- `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
```

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.

```
(arrayA <- array(1:16, dim=c(2, 4, 2)))      # 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, columns and variables 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

{sec: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 *factors*, 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 (`rep=TRUE`). Finally, all four variables are combined into the data frame `mydata`.

```
set.seed(12345) # reproducibility
n=100
A <- factor(sample(c("a1", "a2"), n, rep=TRUE))
B <- factor(sample(c("b1", "b2"), n, rep=TRUE))
sex <- factor(sample(c("M", "F"), n, rep=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 ...
```

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") one or more strings which 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`.

{ex:ch2-arth-csv}

EXAMPLE 2.1: Arthritis treatment

³The foreign 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`, `XLConnect`, `xlsx`). The R manual, *R Data Import/Export* covers many other variations, including data in relational data bases.

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. 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.csv()` as shown below, using all the default options:

```
Arthritis <- read.csv("ch02/Arthritis.csv")
str(Arthritis)

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

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

```
levels(Arthritis$Improved)

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

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

△

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

{sec:forms}

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.

{ex:ch2-arth}

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"
## [5] "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 ...
## $ Sex      : Factor w/ 2 levels "Female","Male": 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 `nrowX`, and the total number of observations is the sum of the frequency variable, `sum(X$Freq)`, `sum(X[, "Freq"])` or similar expression.

{ex:ch2-GSS}

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:

sex	party		
	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
GSS <- data.frame(
  expand.grid(sex=c("female", "male"),
             party=c("dem", "indep", "rep")),
  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,

```
ex:ch2-hec} sum(X).
```

EXAMPLE 2.4: Hair-eye color data

A classic data set on frequencies of hair color, eye color and sex is given in table form in `HairEyeColor` in the `vcd` 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 `strctable()` 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.⁴ You can 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"
```

Here is another similar example, entering data on job satisfactory 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), 4, 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

{sec:ordered}

As we saw above (Example 2.1), factor variables in data frames (case form or frequency form) can be re-ordered and declared as ordered factors using `ordered()`. As well, the order of the factors 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 reflects 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.

⁴There 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.

```
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" ...

UCB <- aperm(UCBAdmissions, c(2, 1, 3))
dimnames(UCB)[[2]] <- 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()`

{sec:table}

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, rep=TRUE))
B <- factor(sample(c("b1", "b2"), n, rep=TRUE))
sex <- factor(sample(c("M", "F"), n, rep=TRUE))
age <- round(rnorm(n, mean=30, sd=5))
mydata <- data.frame(A, B, sex, age)
```

`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, 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

(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 percentages

##           B
## A         b1  b2
## a1    0.18  0.30
## a2    0.22  0.30

prop.table(mytab, 1)        # row percentages

##           B
## A         b1  b2
## a1    0.3750 0.6250
## a2    0.4231 0.5769

prop.table(mytab, 2)        # column percentages

##           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.

2.4.1 xtabs()

{sec: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.

```
# 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-square 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 = 1.5, df = 4, p-value = 0.8
```

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. With the option `shade=TRUE`, the cells are shaded according to the deviations (residuals) from an independence model. Mosaic plots are discussed in detail in Chapter 5.

```
plot(mytable)
plot(GSStab, shade=TRUE)
```

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

{sec:structable}

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.

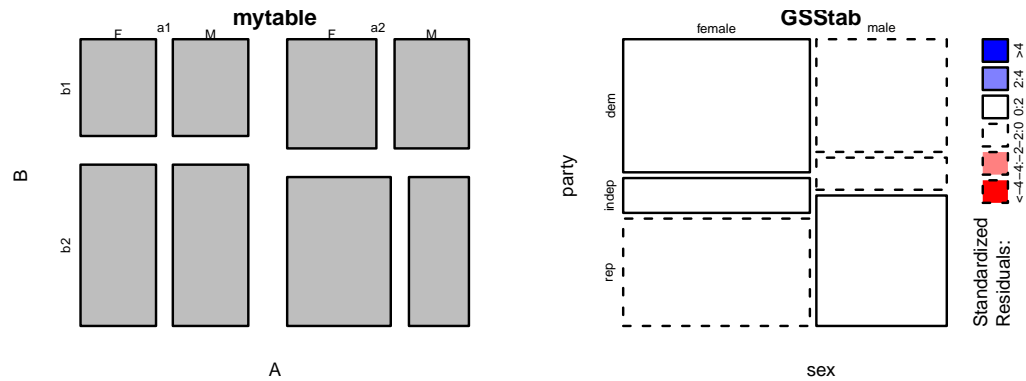


Figure 2.2: Mosaic plot of tables using the plot method for table objects^{fig:plot-xtab}

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.

```
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

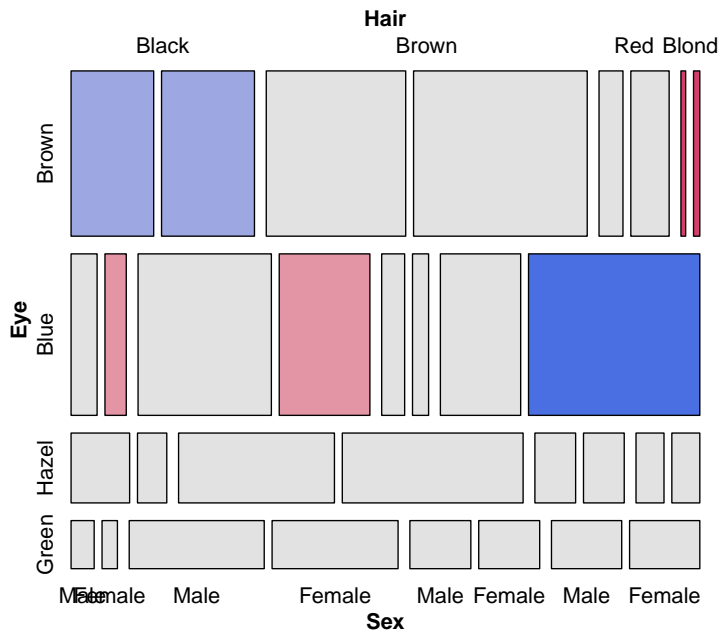
##          Hair Black      Brown      Red      Blond
##          Sex  Male Female Male Female Male Female Male Female
## Eye
## Brown      32      36      53      66      10      16      3      4
## Blue       11      9      50      34      10      7      30     64
## Hazel      10      5      25      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 `t(structable(HairEyeColor))` produces the second result shown just above; "structable" objects can be subset using the `[]` and `[[` operators, using either level indices or names. There are also plot methods, so that `plot()` and `mosaic()` produce mosaic plots.

```
HSE <- structable(Hair+Sex ~ Eye, HairEyeColor) # save structable object
HSE[1:2,] # subset rows

##          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

mosaic(HSE, shade=TRUE, legend=FALSE) # plot it
```



2.5.1 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 description of the `texreg` package and a comparison with some of the other packages.

Here, we simply illustrate the `xtable` 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 containing 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 (without the usual `##` comment used to indicate R output).

```
library(xtable)
xtable(HorseKicks)

% latex table generated in R 3.0.1 by xtable 1.7-3 package
% Sun Feb 23 15:56:34 2014
\begin{table}[ht]
\centering
\begin{tabular}{rr}
\hline
& nDeaths \\
\hline
0 & 109 \\
1 & 65 \\
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)
```

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

	nDeaths
0	109
1	65
2	22
3	3
4	1

HorseKicks table to a data frame (see Section 2.7 for details), add the appropriate `colnames`, and use the `xtable.print` 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

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

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

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

Table 2.1: von Bortkiewicz's data on deaths by horse kicks

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

`c:collapse}`

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

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.

`{ex:dayton1}`

EXAMPLE 2.5: 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.

```
{ex:dayton2}
```

EXAMPLE 2.6: 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      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
```

△

Many of these operations can be performed using the `**ply()` functions in the `plyr` package. For example, with the data in a frequency form data frame, use `ddply()` to collapse over unmentioned factors, and `plyr::summarise()`⁵ as the function to be applied to each piece.

⁵Ugh. This `plyr` function clashes with a function of the same name in `vcdExtra`. In this book I will use the explicit double-colon notation to keep them separate.

```
Dayton.ACM.df <- ddpoly(DaytonSurvey, .(cigarette, alcohol, marijuana),
  plyr::summarise, Freq=sum(Freq))
```

2.6.1 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.

{ex:collapse-cat}

EXAMPLE 2.7: 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
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	91	110	106	95	107	98
##		med	108	104	97	100	107	112
##		high	104	104	106	101	92	95
##	Female	low	115	103	96	93	112	94
##		med	96	88	92	103	98	93
##		high	84	93	103	93	95	103

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	413	398	424
##		high	208	207	187
##	Female	<high	402	384	397
##		high	177	196	198

2.7 Converting among frequency tables and data frames

{sec:conver

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 2.2 shows some handy tools for converting from one form to another.

{tab:convert}

Table 2.2: Tools for converting among different forms for categorical data

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

2.7.1 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.

{ex:GSS-convert}

EXAMPLE 2.8: General social survey

Convert the `GSStab` 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.

{ex:horse.df}

EXAMPLE 2.9: Horse kicks data

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.

⁶Because R is object-oriented, this is actually a short-hand for the function `as.data.frame.table()`.

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

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

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

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

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

horse.df

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

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

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

## [1] 2
```

△

2.7.2 Case form to table form

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

{ex:Arth-convert}

EXAMPLE 2.10: Arthritis treatment

Convert the Arthritis data in case form to a 3-way table of Treatment \times Sex \times 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
```

△

2.7.3 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** only operates on data in this format. The function `expand.dft()`⁷ in **vcdExtra** does this, converting a table into a case form.

```
{ex:Arth-convert2}
```

EXAMPLE 2.11: 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`.

```
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 A complex example

If you've followed so far, congratulations! You're 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 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 Neilson ratings. This data set contains sample television audience data from Nielsen 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`.

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

The table variables are:

- V1– values 1:5 correspond to the days Monday–Friday;
- V2– values 1:11 correspond to the quarter hour times 8:00PM through 10:30PM;
- 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.

2.8.1 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")
```

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 supplies 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 the (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()` (or `plyr::daply()`) 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.

2.8.2 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 times into a smaller number. Here, we use `as.data.frame.table()` to convert the table back to a data frame, `levels()` to re-assign the values of `Time`, and finally, `xtabs()` to give a new, collapsed frequency table.

⁸This 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.

```
TV.df <- as.data.frame.table(TV)
levels(TV.df$Time) <- c(rep("8:00-8:59", 4),
                        rep("9:00-9:59", 4), rep("10:00-10:44", 3))
TV2 <- xtabs(Freq ~ Day + Time + Network, TV.df)
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.

As a final step and a prelude to what follows, we construct a mosaic plot, below (Figure 2.3) that focuses on the associations between the combinations of Day and Time and the Network viewed. In terms of a loglinear model, this is represented by the model formula `~Day:Time + Network`, which asserts that Network is independent of the Day:Time combinations.

```
dimnames(TV2)$Time <- c("8", "9", "10") # re-level for mosaic display
mosaic(~ Day + Network + Time, data=TV2, expected=~Day:Time + Network,
       legend=FALSE, gp=shading_Friendly)
```

The cells shaded in blue show positive associations (observed frequency > expected) and red shows negative associations. From this it is easy to read how network choice varies with day and time. For example, CBS dominates in all time slots on Monday; ABC and NBC dominate on Tuesday, particularly in the later time slots; Thursday is an NBC day, while on Friday, ABC gets the greatest share.

2.9 Further reading

{sec:ch02-reading}

If you're new to the R language but keen to get started with linear modeling or logistic regression in the language, take a look at this *Introduction to R*, <http://data.princeton.edu/R/introducingR.pdf>, by Germán Rodríguez.

2.10 Lab exercises

{sec:ch02-exercises}

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.

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

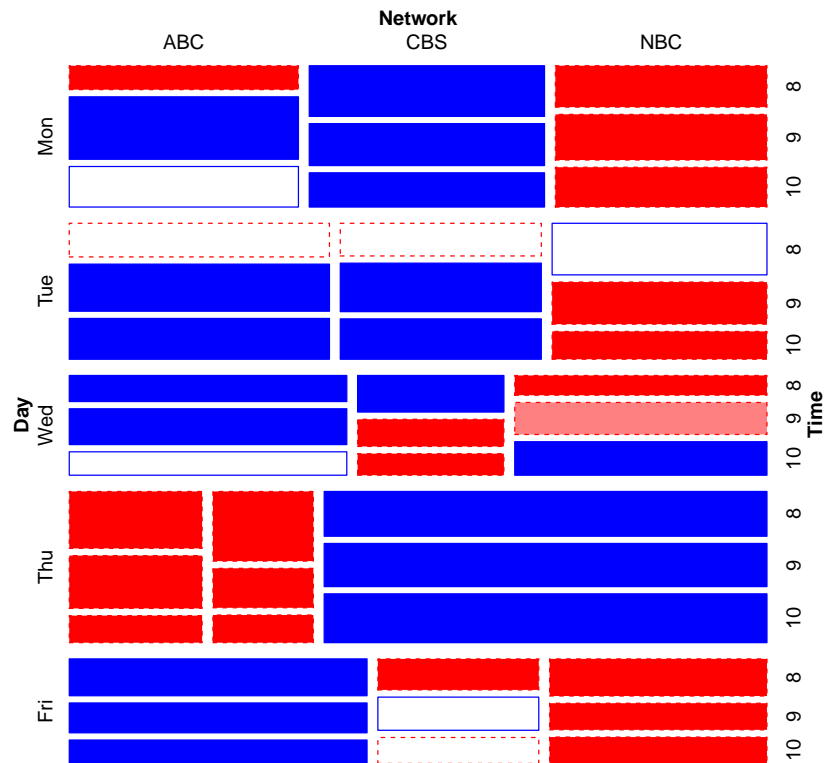


Figure 2.3: Mosaic plot for the TV data showing model of joint independence, Day:Time + Network

```
ds <- datasets(package=c("vcd", "vcdExtra"))
str(ds)

## 'data.frame': 65 obs. of 5 variables:
## $ Package: chr "vcd" "vcd" "vcd" "vcd" ...
## $ Item : chr "Arthritis" "Baseball" "BrokenMarriage" "Bundesliga" ...
## $ class : chr "data.frame" "data.frame" "data.frame" "data.frame" ...
## $ dim : chr "84x5" "322x25" "20x4" "14018x7" ...
## $ Title : chr "Arthritis Treatment Data" "Baseball Data" "Broken Marriage Data"
```

- How many data sets are there altogether? How many are there in each package?
 - Make a tabular display of the frequencies by Package and class.
 - 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.
- The data set `UCBAdmissions` is a 3-way table of frequencies classified by *Admit*, *Gender* and *Dept*.
 - Find the total number of cases contained in this table.
 - For each department, find the total number of applicants.
 - 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.
3. 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.
- Find the total number of cases represented in this table.
 - In this form, the variables `Alcohol` and `Income` should arguably be considered *ordered* factors. Change them to make them ordered.
 - Convert this data frame to table form, `DanishWelfare.tab`, a 4-way array containing the frequencies with appropriate variable names and level names.
 - 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`?
 - 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.
4. 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.

- Verify that the total number of games represented in this table is 380.
 - Find the marginal total of the number of goals scored by each of the home and away teams.
 - Express each of the marginal totals as proportions.
 - 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?
5. 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
##      3     24    104    286    670   1033   1343   1112    829    478    181    45
##     12
##      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 ...
```

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

6. *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 `strctable()` 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

Hint: convert to a data frame, manipulate the factors, then convert back to a table.

TODO: Cleanup some local variables

```
.locals["ch02"] <- setdiff(ls(), .globals)
ls(.locals["ch02"])

remove(list=objects(pattern="array|mat|my|\\.tab|\\.df"))
remove(list=c("A", "B", "age", "count", "ds", "n", "passed", "sex", "tab", "tv.dat",
ls()
```

Chapter 3

Fitting and graphing discrete distributions

{ch:discrete}

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 ?) where they serve as one component. They also provide the basis for a variety of recent extensions of regression models for count data (Chapter ?), allowing excess counts of zeros (zero-inflated models), 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.

It then describes methods for fitting data to a distribution of a given form and simple, 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 determine the impact of individual observations on estimated parameters.

3.1 Introduction to discrete distributions

{sec:discre

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, prediction of 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, $k = 0, 1, \dots$, 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

{sec:binom-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 $k = 0, 1, 2, \dots, n$ successes in the n trials.

{ex:arbuthnot1}

EXAMPLE 3.1: Arbuthnot data

Sex ratios—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 the 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 `HistData` 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.

```
data(Arbuthnot, package="HistData")
with(Arbuthnot, {
  prob = Males/(Males+Females)
  plot(Year, 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")
  text(x=1640, y=0.5, expression(H[0]: "Pr(Male)=0.5"), pos=3, col="red")
  Arb.smooth <- loess.smooth(Year, prob)
  lines(Arb.smooth$x, Arb.smooth$y, col="blue", lwd=2)
})
```

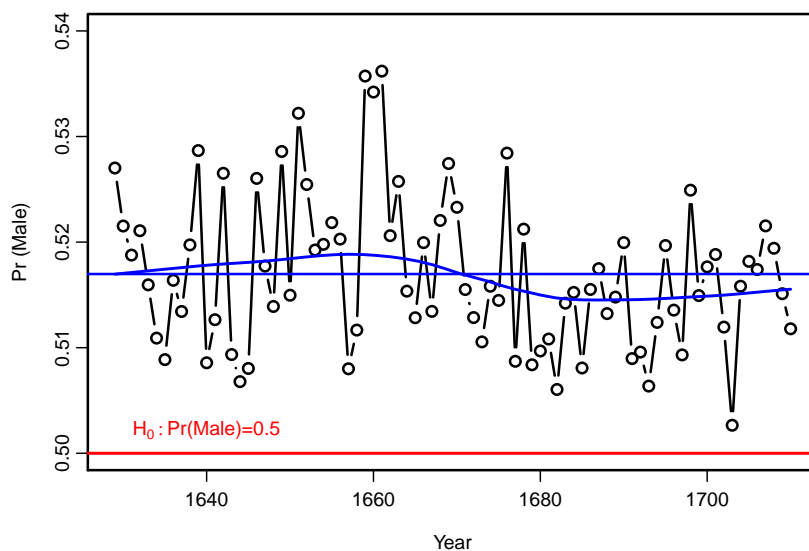


Figure 3.1: Arbuthnot's data on male/female sex ratios in London, 1629–1710, together with a (loess) smoothed curve over time and the mean $\Pr(\text{Male})$ fig:arbuthnot1

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. \triangle

{ex:saxony1}

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

¹Edwards (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 in the 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.

{tab:saxtab}

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

Males (k)	0	1	2	3	4	5	6	7	8	9	10	11	12	Sum
Families (n_k)	3	24	104	286	670	1033	1343	1112	829	478	181	45	7	6115

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)
```

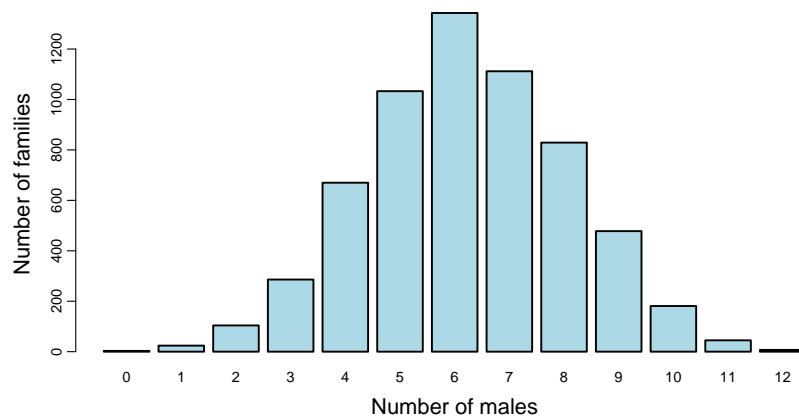


Figure 3.2: Males in Saxony families of size 12^{fig:saxony-barplot}

△

{ex:dice}

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 times, 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 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.

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.

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

# 5s or 6s (k)	0	1	2	3	4	5	6	7	8	9	10+	Sum
Frequency (n_k)	185	1149	3265	5475	6114	5194	3067	1331	403	105	18	26306

```
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)
```

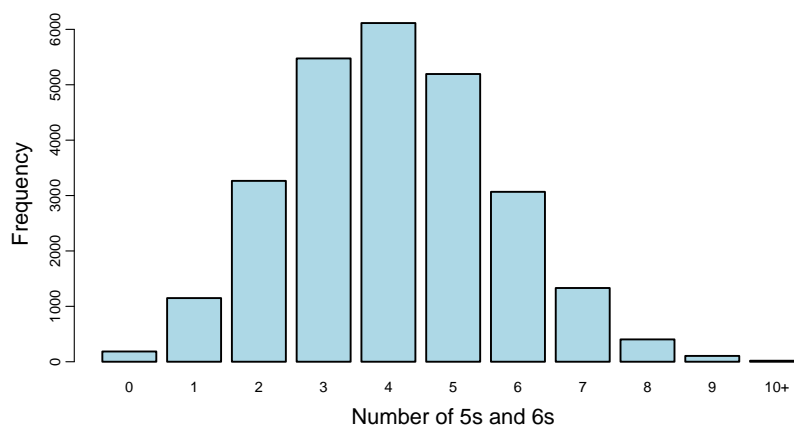


Figure 3.3: Weldon's dice data fig:dice

△

3.1.2 Poisson data

{sec:pois-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

²The 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.

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.

`{ex:horskick1}` 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, 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.

`{tab:horsetab}`

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

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

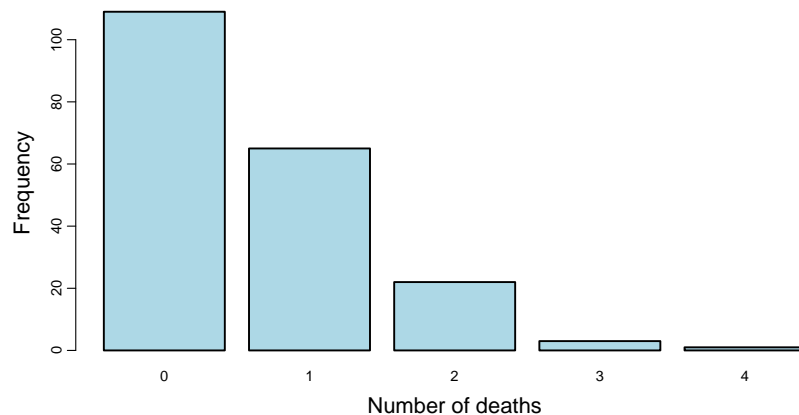


Figure 3.4: HorseKicks data `fig:horsekicks`

△

`{ex:madison1}`

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.

Table 3.4: Number of occurrences of the word *may* in texts written by James Madison^{tab:fedtab}

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

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

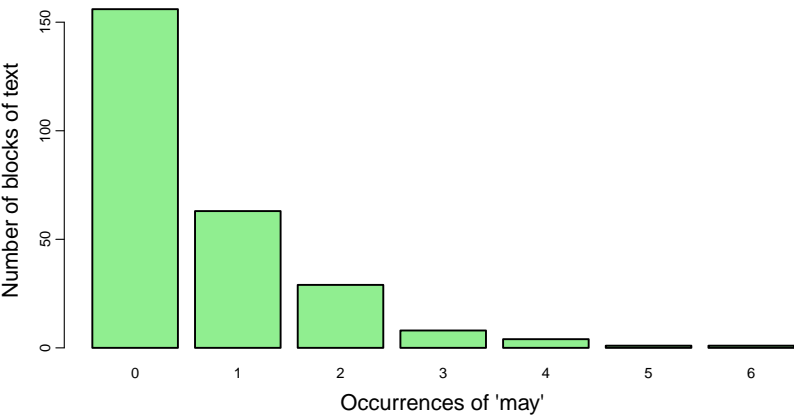


Figure 3.5: Mosteller and Wallace Federalist data^{fig:federalist}

△
{ex:cyclists1}

EXAMPLE 3.6: London cycling deaths

Aberdeen 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 question, they obtained data from the UK Department of Transport

³It should be noted that this is a landmark work in the development and application of statistical methods 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. Amongst these, the word *upon* was the best discriminator between the works known by Hamilton (3 per 1000 words) and Madison (1/6 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.

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, . . . 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
```

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

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

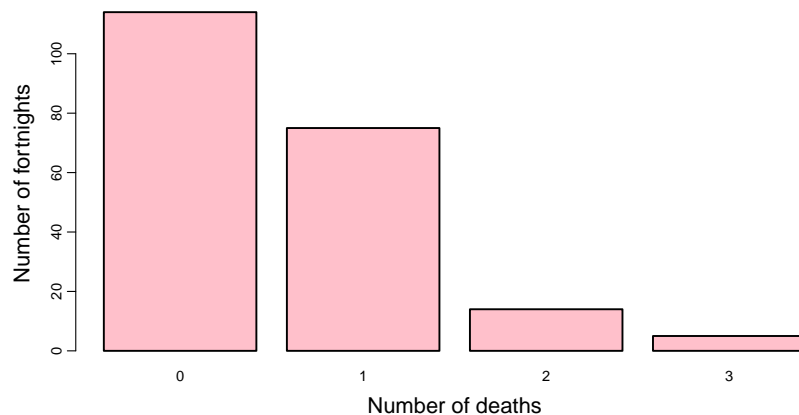


Figure 3.6: Frequencies of number of cyclist deaths in two-week periods in London, 2005–2012 fig:cyclists2

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

{sec:type-token}

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,

$k = 1, 2, \dots$ 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 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.

{ex:butterfly}

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 $k = 1, 2, \dots$ 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 3.6.

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

{tab:buttertab}

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

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

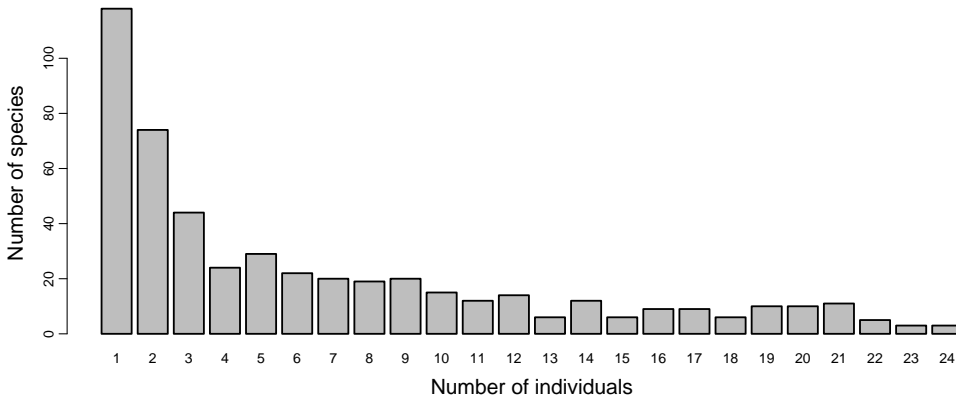


Figure 3.7: Butterfly species in Malaya

fig:butterfly

3.2 Characteristics of discrete distributions

{sec:discre

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.

Table 3.6: Discrete probability distributions^{tab:distns}

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

For each distribution, we describe properties and generating mechanisms, and show how its parameters can be estimated and how to plot the frequency distribution. **R** has a wealth of functions 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 $\text{Pr}\{X = x\} \equiv p(x)$ for the probability that the variable X takes the value x .

p a cumulative probability function, or CDF, $F(x) = \sum_{X \leq x} p(x)$.

q a quantile function, the inverse of the CDF, $x = F^{-1}(p)$. The quantile is defined as the smallest value x such that $F(x) \geq p$.

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

{sec:binomial}

The binomial distribution, $\text{Bin}(n, p)$, arises as the distribution of the number of events of interest which occur in n independent trials when the probability of the event on any one trial is the constant value $p = \text{Pr}(\text{event})$. For example, if 15% of the population has red hair, the number of red-heads 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 $\text{Bin}(12, \frac{1}{3})$.

⁴The 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.

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

Table 3.7: R functions for discrete probability distributions^{tab:distfuns}

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

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

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

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 $q = 1 - p$),

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

It is easy to verify that the binomial distribution has its maximum variance when $p = \frac{1}{2}$. It is symmetric ($\text{Skew}[X]=0$) 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 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 $\text{Var}(\hat{p}) = pq/n$.

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 up 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)`.

For example, to find and plot the binomial probabilities corresponding to Weldon's tosses of 12 dice, with $n = 0, \dots, 12$ and $p = \frac{1}{3}$, we could do the following

```
x <- seq(0, 12)
plot(x=x, y=dbinom(x, 12, 1/3), type="h",
     xlab="Number of successes", ylab="Probability",
     lwd=8, lend="square")
lines(x=x, y=dbinom(x, 12, 1/3))
```

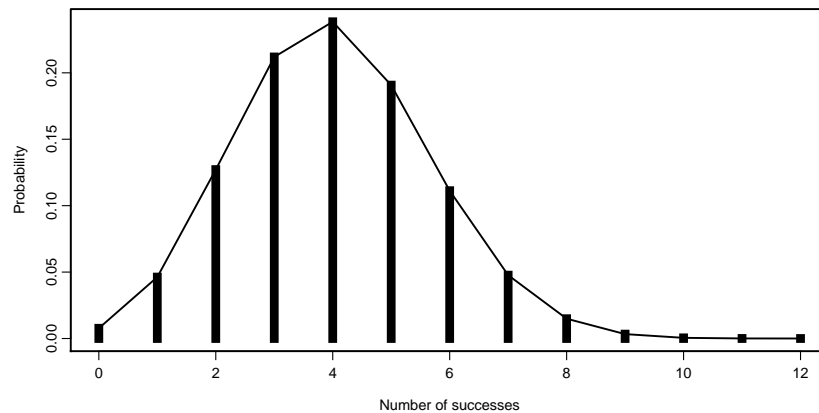


Figure 3.8: Binomial distribution for $n=0-12$ trials and $p=1/3$ ^{fig:dbinom1}

Note that in the call to `plot()`, `type="h"` draws histogram type lines to the bottom of the vertical axis, and `lwd=8` makes them wide. The call to `lines()` shows another way to plot the data, as a probability polygon. We illustrate other styles for plotting in Section 3.2.2, Example 3.11 below.

{ex:dice2}

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 $N = 26306$ tosses. The χ^2 test for goodness of fit is described later in Section 3.3, but a glance at the `Diff` column shows that these are all negative for $k = 0, \dots, 4$ and positive thereafter.

```
Weldon.df <- as.data.frame(WeldonDice) # convert to data frame

x <- seq(0, 12)
Prob <- dbinom(x, 12, 1/3) # binomial probabilities
Prob <- c(Prob[1:10], sum(Prob[11:13])) # sum values for 10+
Exp = round(26306*Prob) # expected frequencies
Diff = Weldon.df[, "Freq"] - Exp # raw residuals
Chisq = Diff^2 / Exp
data.frame(Weldon.df, Prob=round(Prob, 5), Exp, Diff, Chisq)
```

##	n56	Freq	Prob	Exp	Diff	Chisq
## 1	0	185	0.00771	203	-18	1.5961
## 2	1	1149	0.04624	1216	-67	3.6916
## 3	2	3265	0.12717	3345	-80	1.9133
## 4	3	5475	0.21195	5576	-101	1.8294
## 5	4	6114	0.23845	6273	-159	4.0301
## 6	5	5194	0.19076	5018	176	6.1730
## 7	6	3067	0.11127	2927	140	6.6963
## 8	7	1331	0.04769	1255	76	4.6024
## 9	8	403	0.01490	392	11	0.3087
## 10	9	105	0.00331	87	18	3.7241
## 11	10+	18	0.00054	14	4	1.1429

Finally, we can use programming features in R to calculate and plot probabilities for binomial distributions over a range of both x and p as follows, for the purposes of graphing the distributions as one or both varies. The following code uses `expand.grid()` to create a data frame `XP` containing all combinations of $x=0:12$ and $p=c(1/6, 1/3, 1/2, 2/3)$. These values are then supplied as arguments to `dbinom()`. For the purpose of plotting, the decimal value of p is declared as a factor.

```
XP <- expand.grid(x=0:12, p=c(1/6, 1/3, 1/2, 2/3))
bin.df <- data.frame(XP, prob=dbinom(XP[, "x"], 12, XP[, "p"]))
bin.df$p <- factor(bin.df$p, labels=c("1/6", "1/3", "1/2", "2/3"))
str(bin.df)

## 'data.frame': 52 obs. of 3 variables:
## $ x : int 0 1 2 3 4 5 6 7 8 9 ...
## $ p : Factor w/ 4 levels "1/6","1/3","1/2",...: 1 1 1 1 1 1 1 1 1 1 ...
## $ prob: num 0.1122 0.2692 0.2961 0.1974 0.0888 ...
```

This data can be plotted using `xyplot()` in `lattice`, using the `groups` argument to make separate curves for each value of p . The following code generates Figure 3.8.

```
library(lattice)
mycol <- palette()[2:5]
xyplot( prob ~ x, data=bin.df, groups=p,
  xlab=list('Number of successes', cex=1.25),
  ylab=list('Probability', cex=1.25),
  type='b', pch=15:17, lwd=2, cex=1.25, col=mycol,
  key = list(
    title = 'Pr(success)',
    points = list(pch=15:17, col=mycol, cex=1.25),
    lines = list(lwd=2, col=mycol),
    text = list(levels(bin.df$p)),
    x=0.9, y=0.98, corner=c(x=1, y=1)
  )
)
```

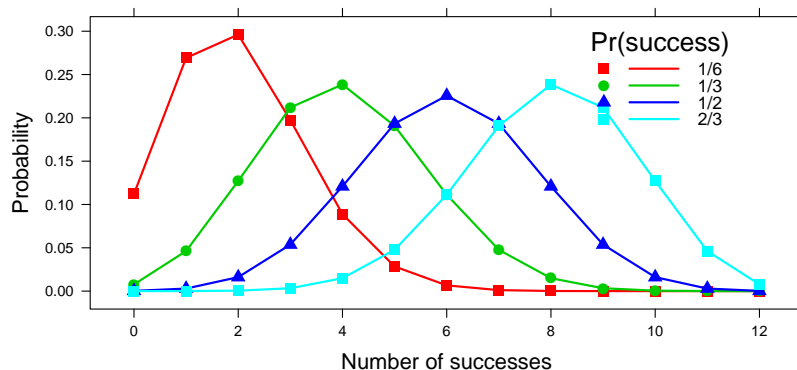


Figure 3.9: Binomial distributions for $n=0-12$ trials and four values of p fig:dbinom2-plot

3.2.2 The Poisson distribution

{sec:poisson}

The Poisson distribution gives the probability of an event occurring $k = 0, 1, 2, \dots$ 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, 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

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

where the rate parameter, $\lambda (> 0)$ 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,

$$\text{\{eq:pois-lambda\}} \quad \hat{\lambda} = \bar{x} = \frac{\sum_k k n_k}{\sum_k n_k} . \quad (3.3)$$

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.⁶ Among these:

- 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*, $\sum_j X_j = k$ is binomial, $\text{Bin}(k, p)$, where $p = \lambda_i / \sum_j \lambda_j$.
- As λ increases, the Poisson distribution becomes increasingly symmetric, and approaches the normal distribution $N(\lambda, \lambda)$ with mean and variance λ as $\lambda \rightarrow \infty$. 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.

\text{\{ex:soccer\}}

⁶See: http://en.wikipedia.org/wiki/Poisson_distribution

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

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

```
ab:soccer1}
```

Home Team Goals	Away Team Goals					Total
	0	1	2	3	4+	
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

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 scores 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 independent⁷, 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

```
{tab:soccer2}
```

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

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

In the R code below, we first convert the two-way frequency table `UKSoccer` to a data frame in frequency form. We use `within()` to convert *Home* and *Away* to numeric variables,

⁷This question is examined visually in Chapter 5 (Example 5.5) and Chapter 6 (Example ??), where we find that the answer is “basically, yes”.

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 1.487 1.063 2.550
## var  1.316 1.173 2.618
```

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.

△

{ex:cyclists2}

EXAMPLE 3.10: London cycling deaths

A quick check of whether the numbers 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.

```
(mean <- mean(CyclingDeaths$deaths))

## [1] 0.5673

(var <- var(CyclingDeaths$deaths))

## [1] 0.5268

var/mean

## [1] 0.9287
```

Thus, there was an average of 0.58 deaths per fortnight, or a bit more than 1 per month, and no evidence for over- or under- dispersion.

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()`.

```
1 - ppois(5, mean)

## [1] 2.854e-05

ppois(5, mean, lower.tail=FALSE)

## [1] 2.854e-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. \triangle

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 with the values of `x` we wish to plot.

{ex:dpois-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.

```
XL <- expand.grid(x=0:20, lambda=c(1, 4, 10))
pois.df <- data.frame(XL, prob=dpois(XL[, "x"], XL[, "lambda"]))
pois.df$lambda = factor(pois.df$lambda)
str(pois.df)

## 'data.frame': 63 obs. of 3 variables:
## $ x      : 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 3.8. 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 ~ x | lambda` instructs `xyplot()` to produce a multi-panel plot, conditioned on values of `lambda`. These lines produce Figure 3.9.

```
xyplot( prob ~ x | 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))
```

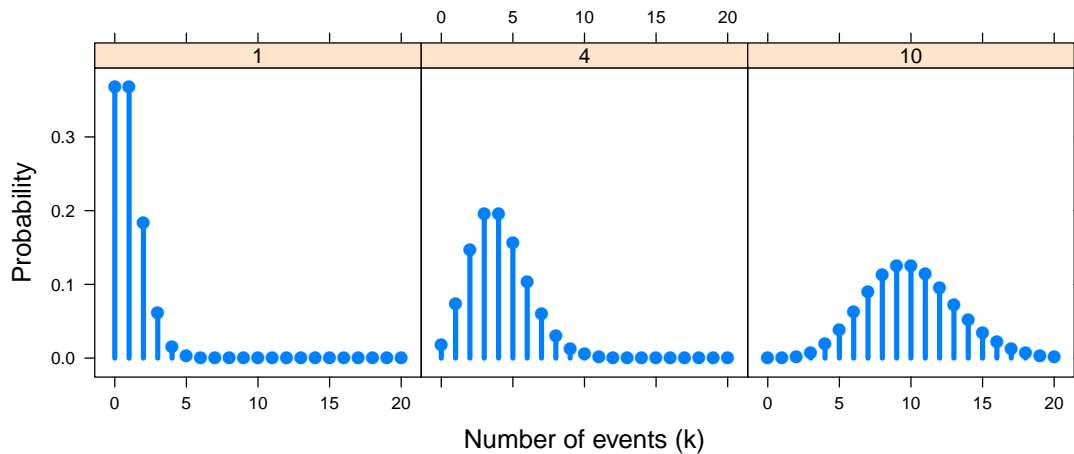


Figure 3.10: Poisson distributions for $\lambda = 1, 4, 10$, in a multi-panel display ^{fig:dpois-xyplot1}

The line-graph plot style of Figure 3.8 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 3.10 solves both problems using the `directlabels` package.

```
mycol <- palette()[2:4]
plt <- xyplot( prob ~ x, 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))

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=x, y=prob, ...)` to produce a plot of `prob` vs. `x`, assigning color and shape attributes to the values of `lambda`.

```
library(ggplot2)
gplt <- ggplot(pois.df, aes(x=x, 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.

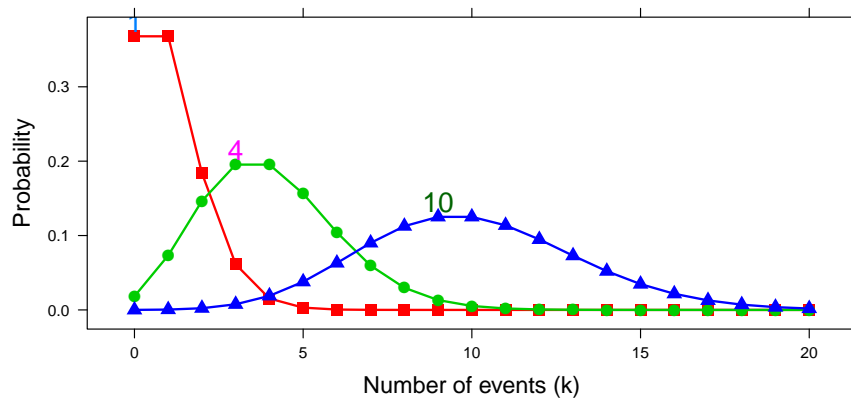


Figure 3.11: Poisson distributions for $\lambda = 1, 4, 10$, using direct labels fig:dpois-xyplot2

```
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"))
```

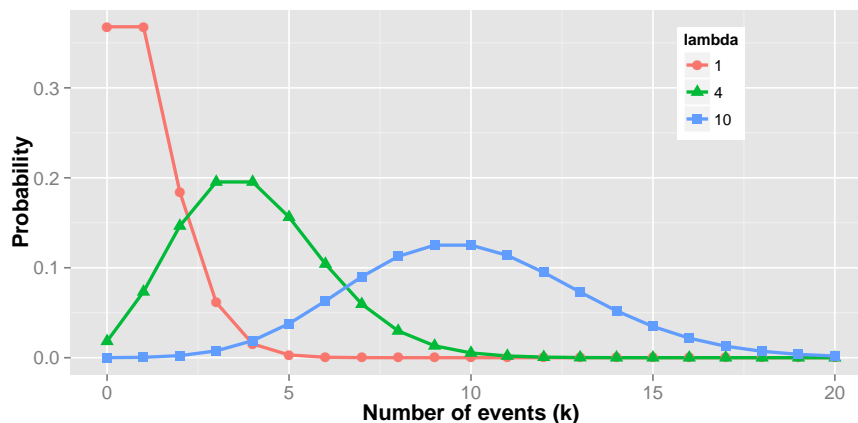


Figure 3.12: Poisson distributions for $\lambda = 1, 4, 10$, using ggplot fig:dpois-ggplot2

△

3.2.3 The negative binomial distribution

{sec:negbin}

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 $k = 0, 1, 2, \dots$ 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 written⁸

$$\text{\{eq:negbinf\}} \quad \text{NBin}(n, p) : \Pr\{X = k\} \equiv p(k) = \binom{n+k-1}{k} p^n (1-p)^k \quad k = 0, 1, \dots, \infty \quad (3.4)$$

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

$$\text{\{eq:negbinf2\}} \quad \Pr\{X = k\} \equiv p(k) = \frac{\Gamma(n+k)}{\Gamma(n)\Gamma(k+1)} p^n (1-p)^k \quad k = 0, 1, \dots, \infty \quad (3.5)$$

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,

⁸There 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 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.

$$\text{Mean}[X] = \mu = \frac{n(1-p)}{p} \implies p = \frac{n}{n+\mu} \quad (3.6)$$

$$\text{Var}[X] = \frac{n(1-p)}{p^2} \implies \text{Var}[X] = \mu + \frac{\mu^2}{n} \quad (3.7)$$

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*.⁹ Increasing this parameter corresponds to less heterogeneity, variance closer to the mean, and therefore greater applicability of the Poisson distribution.

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 `dnbinom(k, n, p)` or the call `dnbinom(k, n, mu=)`, as illustrated below:

```
k = 2
n = 2:4
p = .2
dnbinom(k, n, p)

## [1] 0.07680 0.03072 0.01024

mu = n*(1-p)/p
mu

## [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)
```

⁹Other terms are “shape parameter,” with reference to the mixing distribution of Poissons with varying λ , “heterogeneity parameter,” or “aggregation parameter.”


```
## '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. 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)
        )
```

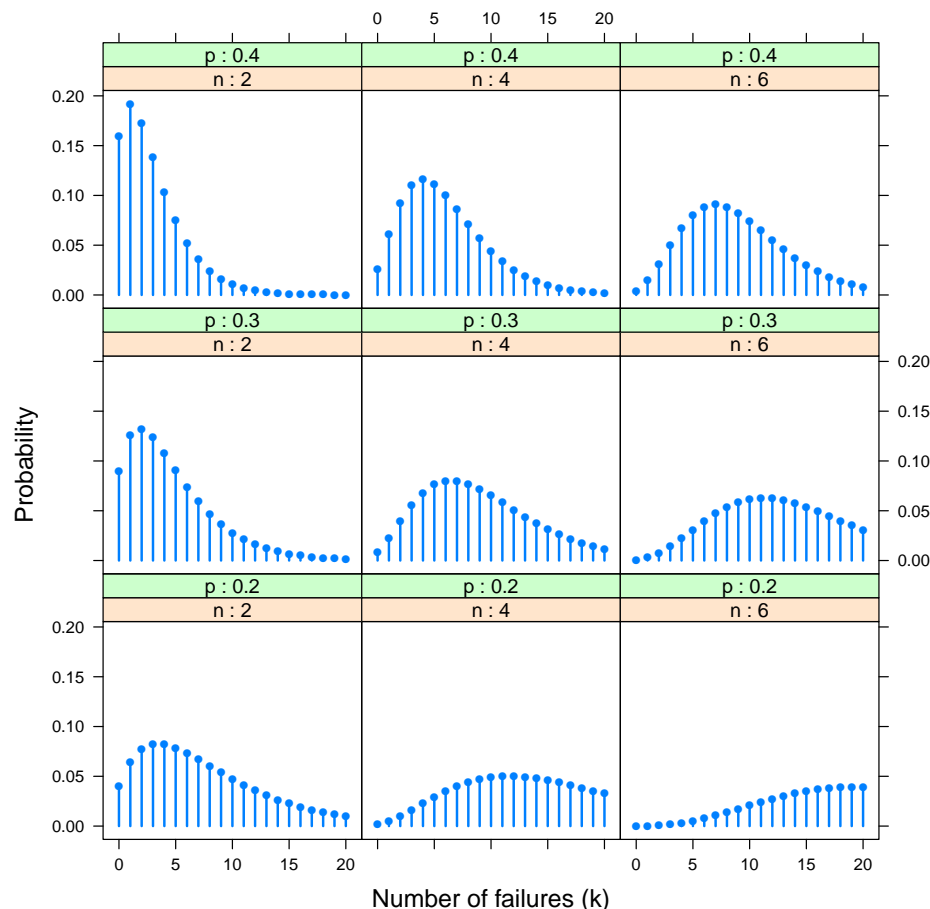


Figure 3.13: Negative binomial distributions for $n = 2, 4, 6$ and $p = 0.2, 0.3, 0.4$, using `xyplot` ^{fig:dnbin3}

TODO: Modify Figure 3.12 to show the mean and standard deviation

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, $k = 0, 1, \dots, \infty$, as shown below.

```

NP <- expand.grid(n=c(2, 4, 6), p=c(0.2, 0.3, 0.4))
NP <- within(NP, { mu = n*(1-p)/p })
# show as matrix
matrix(NP$mu, 3, 3, dimnames=list(n=c(2,4,6), p=(2:4)/10))

##      p
## n    0.2    0.3 0.4
## 2     8   4.667  3
## 4    16   9.333  6
## 6    24  14.000  9

```

3.2.4 The geometric distribution

{sec:geometric}

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

$$\text{Geom}(p) : \Pr\{X = k\} \equiv p(k) = p(1 - p)^k \quad k = 0, 1, \dots \quad (3.8) \quad \{\text{eq:geomf}\}$$

For this distribution the central moments are:

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

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, \dots, X_n are independent geometrically distributed as $\text{Geom}(p)$, then their sum, $Y = \sum_{j=1}^n X_j$ 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

$$\text{LogSer}(p) : \Pr\{X = k\} \equiv p(k) = \frac{p^k}{-(k \log(1 - p))} = \alpha p^k / k \quad k = 1, 2, \dots, \infty, \quad (3.9) \quad \{\text{eq:logseriesf}\}$$

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 (1963). 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.

TODO: implement the log-series in `goodfit()` and `distplot()` so this distribution can be used in later sections.

3.2.6 Power series family

{sec:pwrseries}

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

$$p(k) = a(k)\theta^k / f(\theta) \quad k = 0, 1, \dots,$$

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.

Table 3.10: The Power Series family of discrete distributions^{tab:pwrseries}

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

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 3.5.5).

3.3 Fitting discrete distributions

{sec:discrete-fit}

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 test measuring the departure between the observed and fitted frequencies.

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

$$\chi^2 = \sum_{k=1}^K \frac{(n_k - N\hat{p}_k)^2}{N\hat{p}_k} \sim \chi_{(K-s-1)}^2, \quad (3.10) \quad \{\text{eq:chi2}\}$$

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,

$$G^2 = \sum_{k=1}^K n_k \log(n_k / N\hat{p}_k), \quad (3.11) \quad \{\text{eq:g2}\}$$

when the \hat{p}_k are estimated by maximum likelihood, which also has an asymptotic $\chi_{(K-s-1)}^2$ 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. {\text{ex:horsekick3}}

EXAMPLE 3.12: Death by horse kick

We illustrate the basic ideas of goodness-of fit tests with the `HorseKick` 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	0.543351	108.6702	0.001001
## 2	1	65	0.331444	66.2888	0.025057
## 3	2	22	0.101090	20.2181	0.157048
## 4	3	3	0.020555	4.1110	0.300253
## 5	4	1	0.003135	0.6269	0.222006

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.

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

## [1] 0.7054

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

## [1] 0.8719
```

The result, $\chi^2_3 = 0.70537$ shows an extremely good fit of these data to the Poisson distribution, perhaps exceptionally so.¹⁰

△

3.3.1 R tools for discrete distributions

{sec:fitdistr}

In R, the function *fitdistr()* in the MASS 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

¹⁰An exceptionally good fit occurs when the *p*-value for the test χ^2 statistic is so high, as to suggest that 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 2^{nd} generation were uncomfortably too close to the 3 : 1 ratio predicted by genetic theory.

are estimated either by 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.

{ex:saxfit}

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")
Sax.fit$par           # estimated parameters

## $prob
## [1] 0.5192
##
## $size
## [1] 12
```

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 which 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
##      0         3    0.9328
##      1        24   12.0888
##      2       104   71.8032
##      3       286  258.4751
##      4       670  628.0550
##      5      1033 1085.2107
##      6      1343 1367.2794
##      7      1112 1265.6303
##      8       829  854.2466
##      9       478  410.0126
##     10       181  132.8357
##     11        45   26.0825
##     12         7    2.3473

summary(Sax.fit)         # summary method
```

```
##
##      Goodness-of-fit test for binomial distribution
##
##              X^2 df  P(> X^2)
## Likelihood Ratio 97.01 11 6.978e-16
```

Note that the GOF test gives a highly significant p -value, indicating significant lack of fit to the binomial distribution.¹¹ Some further analysis of this result is explored in examples below.

△

{ex:dicefit}

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
## 0.3377 12.0000
```

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

TODO: Fix infelicity in `vcd:::print.goodfit` to provide control of number of digits in the fitted column. – `print(dice.fit)` uses E notation.

```
summary(dice.fit)

##
##      Goodness-of-fit test for binomial distribution
##
##              X^2 df P(> X^2)
## Likelihood Ratio 11.51 9 0.2426
```

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

△

{ex:HKfit}

EXAMPLE 3.15: Death by horse kick

This example reproduces the calculations done “manually” in Example ?? 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
```

¹¹A 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.

```
HK.fit

##
## Observed and fitted values for poisson distribution
## with parameters estimated by `ML'
##
##   count observed   fitted
##      0      109 108.6702
##      1       65  66.2888
##      2       22  20.2181
##      3        3   4.1110
##      4        1   0.6269
```

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 0.8682  3  0.8331
```

△

{ex:Fedfit}

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
## 0.6565

Fed.fit0

##
## Observed and fitted values for poisson distribution
## with parameters estimated by `ML'
##
##   count observed   fitted
##      0      156 135.89139
##      1       63  89.21114
##      2       29  29.28305
##      3        8   6.40799
##      4        4   1.05169
##      5        1   0.13808
##      6        1   0.01511
```

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 25.24  5 0.0001251
```

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
## 1.1863 0.6438

summary(Fed.fit1)

##
## Goodness-of-fit test for nbinomial distribution
##
##           X^2 df P(> X^2)
## Likelihood Ratio 1.964 4 0.7424
```

Recall that the Poisson 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 greater the variance to be greater than the mean.

△

3.3.2 Plots of observed and fitted frequencies

```
{sec:fitplot}
```

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

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

In this section we illustrate some simple plotting tools for these purposes, using the `plot.goodfit()` method for "goodfit" objects.¹² The left panel of Figure 3.13 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

¹²Quantile-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 want to also see the shape of a distribution, as is the case here.

(1977) suggest plotting the frequencies on a square-root scale, which he calls a *rootogram*. This plot is shown in the right panel of Figure 3.13.

```
plot(Fed.fit0, scale="raw", type="standing")
plot(Fed.fit0, type="standing")
```

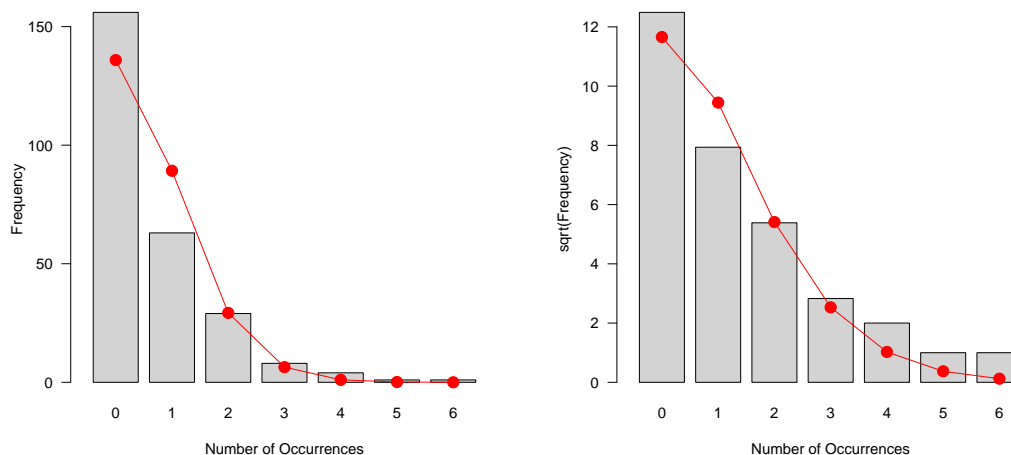


Figure 3.14: Plots for the Federalist Papers data, fitting the Poisson model. Each panel shows the observed frequencies as bars and the fitted frequencies as a smooth curve. Left: raw frequencies; right: plotted on a square root scale to emphasize smaller frequencies.

Additional improvements over the standard plot on the scale of raw frequencies are shown in Figure 3.14, 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")
plot(Fed.fit0, type="deviation")
```

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 3.14, right).

All of these plots are actually produced by the `rootogram()` function in `vcd`. The default is `type="hanging"`, and there are many options to control the plot details.

The plots in Figure 3.13 and Figure 3.14 used the ill-fitting Poisson model on purpose to highlight how these plots show the departure between the observed and fitted frequencies. Figure 3.15 compares this with the negative binomial model, `Fed.fit1` which we saw has a much better, and acceptable fit.

```
plot(Fed.fit0, main="Poisson")
plot(Fed.fit1, main="Negative binomial")
```

Comparing the two plots in Figure 3.15, we can see that the Poisson model underestimates the frequencies of 0 counts and the larger counts for 4-6 occurrences. The deviations for the negative binomial are small and unsystematic.

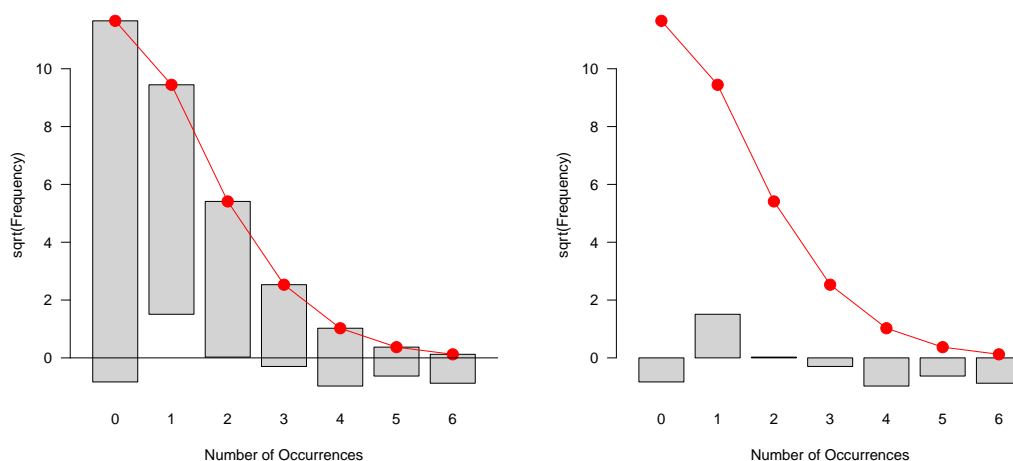


Figure 3.15: Plots for the Federalist Papers data, fitting the Poisson model. Left: hanging rootogram; right: deviation rootogram. fig: Fed0-plots2

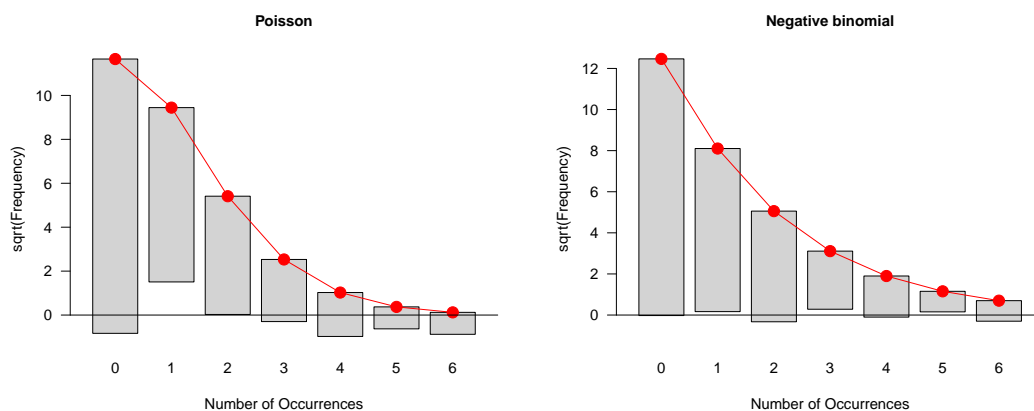


Figure 3.16: Hanging rootograms for the Federalist Papers data, comparing the Poisson and negative binomial models. fig: Fed0-Fed1

Finally, Figure 3.16 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")
plot(But.fit2, main="Negative binomial")
```

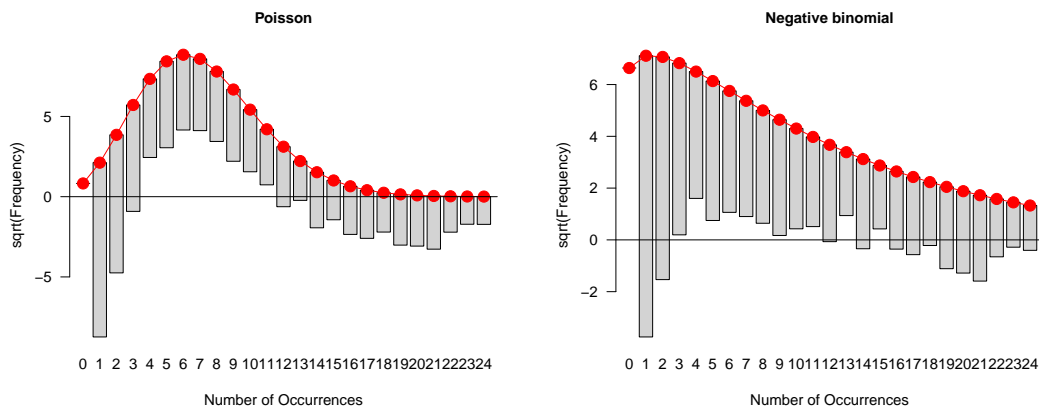


Figure 3.17: Hanging rootograms for the Butterfly data, comparing the Poisson and negative binomial models. The lack of fit for both is readily apparent.

TODO: Old sections here described the general ideas behind maximum likelihood estimation, and methods for fitting discrete distributions as loglinear models. What should be included in this revision?

3.4 Diagnosing discrete distributions: Ord plots

{sec:discrete-ord}

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,

$$\frac{k p(k)}{p(k-1)} \equiv \frac{k n_k}{n_{k-1}} = a + b k, \quad (3.12) \quad \text{{eq:ord}}$$

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

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 kn_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 kn_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 3.17. Note that the function returns (invisibly) the values of the intercept and slope in the weighted LS regression.

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

ord

## Intercept      Slope
##      -0.709      1.061
```

In this plot, the black line shows the usual 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, as regards 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 kn_k/n_{k-1} can vary enormously. The following examples illustrate some other

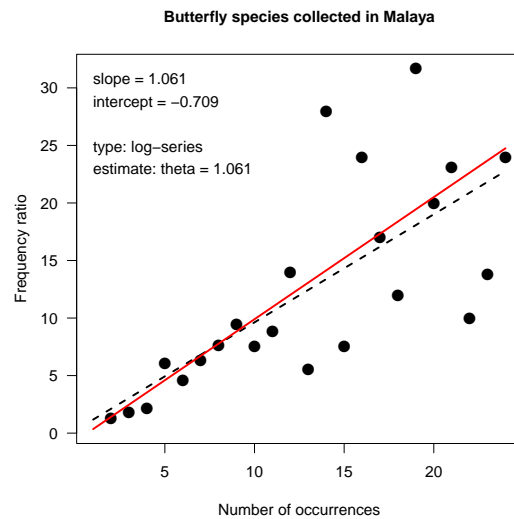


Figure 3.18: Ord plot for the Butterfly data. The slope and intercept in the plot correctly diagnoses the log-series distribution.

distributions and some of the details of the heuristics.

Done: In `vcd_1.3-2`, fixed `Ord_plot()` to provide control of `lwd`, `lty`, etc.

Ord plot examples

{ex:horskick3}

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  10.392
## 2 1  65 109 0.5963   8.000
## 3 2  22  65 0.6769   4.583
## 4 3   3  22 0.4091   1.414
## 5 4   1   3 1.3333   0.000

coef(lm(y ~ k, weights=weight, data=ord.df))

## (Intercept)          k
##    0.65602    -0.03414
```

The weighted least squares line, with weights w_k , has a slope (-0.03) close to zero, indicating

the Poisson distribution.¹³ The estimate $\lambda = a = .656$ compares favorably with the MLE, $\lambda = 0.610$ and the value from the Poissonness plot, shown in the following section. The call to `Ord_plot()` below produces Figure 3.18.

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

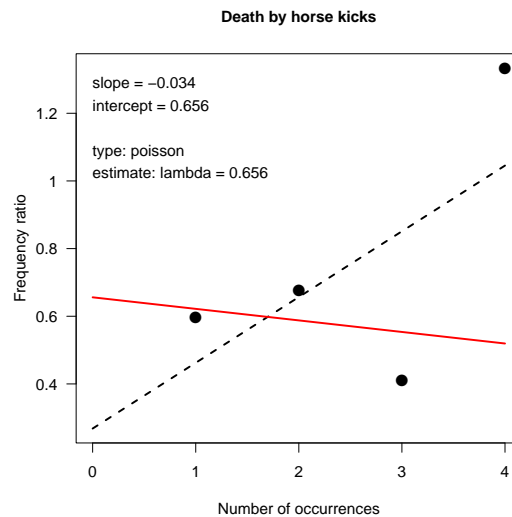


Figure 3.19: Ord plot for the HorseKicks data. The plot correctly diagnoses the Poisson distribution.

△

```
{ex:madison3}
```

EXAMPLE 3.18: Federalist papers

Figure 3.19 (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)
Ord_plot(WomenQueue, main = "Women in queues of length 10",
         gp=gpar(cex=1), pch=16)
```

```
{ex:queues}
```

EXAMPLE 3.19: Women in queues

Jinkinson and Slater (1981), 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`.

¹³The heuristic adopted in `Ord_plot()` uses a tolerance of 0.1 to decide if a coefficient is negative, zero, or positive.

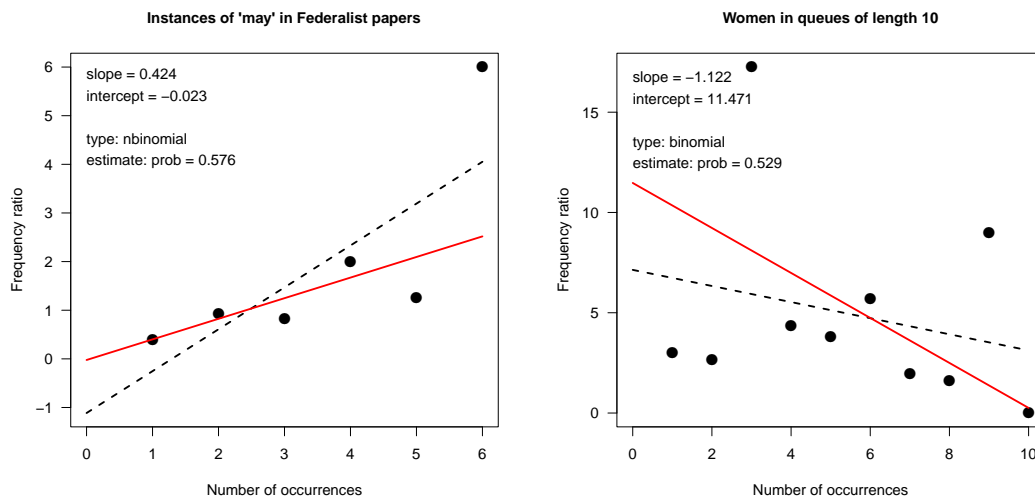


Figure 3.20: Ord plots for the Federalist (left) and WomenQueue (right) data sets. fig:ordplot3

```
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.

Figure 3.19 (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.

△

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, since 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

{sec:discre

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 count metameter 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) develop 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 and then (Section ??) the extension to other distributions.

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 which 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 ,

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

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, $\hat{\lambda} = e^b$ 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.

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

is-leveled}

$$\phi'(n_k) = \phi(n_k) + \lambda_0 - k \log \lambda_0 . \quad (3.14)$$

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.

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

$$\phi(n_k^*) \pm h_k \quad (3.15) \quad \{\text{eq:poisCI}\}$$

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,

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

and $\hat{p}_k = n_k/N$.

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 = λ_0` gives the leveled version of the plot.

{ex:horseki

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 3.20.

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

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

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

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

Using `lambda = 0.61` as below gives the leveled version shown in the right panel of Figure 3.20.

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

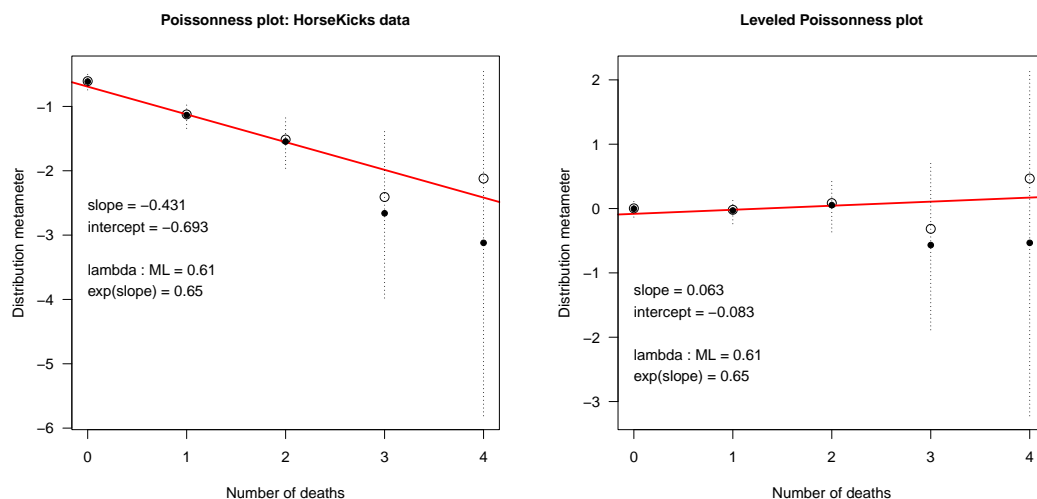


Figure 3.21: Poissonness plots for the HorseKick data. Left: standard plot; right: leveled plot. fig:distplot1

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

△

3.5.4 Leverage and influence

TODO: This subsection should be omitted, unless we extend the calculation and plotting methods.

3.5.5 Plots for other distributions

{sec:discrete-other}

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

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.

{tab:distparms}

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

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

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.

{ex:saxony-distplot}

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 3.21 (left), and even more clearly in the distribution diagnostic plot produced by `distplot()` in the right panel of Figure 3.21. For a binomial distribution, we call this distribution plot a “binomialness plot”.

```
data("Saxony", package="vcd")
plot(goodfit(Saxony, type="binomial", par=list(size=12)))
distplot(Saxony, type="binomial", size=12,
         xlab="Number of males")
```

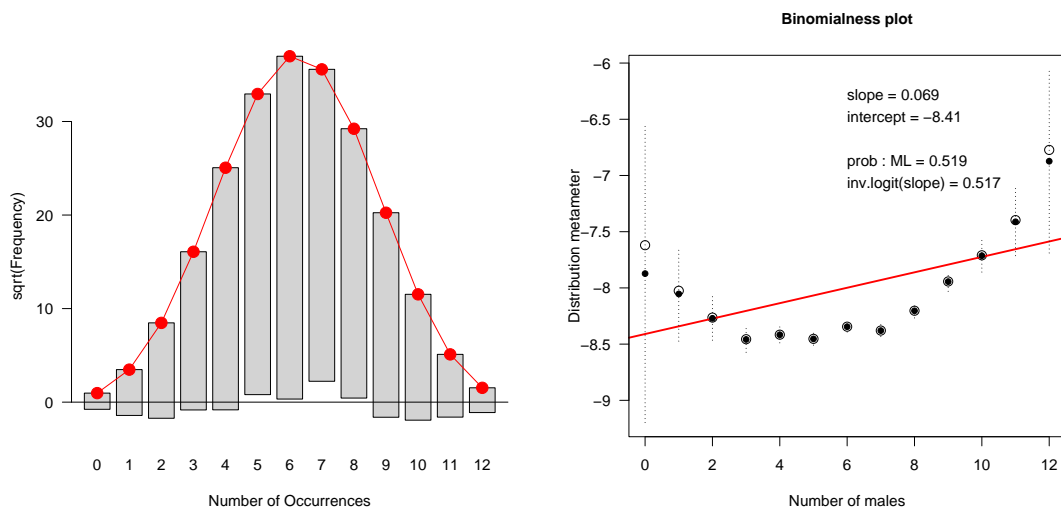


Figure 3.22: Diagnostic plots for males in Saxony families. Left: `goodfit()` plot; right: `distplot()` plot. Both plots show heavier tails than in a binomial distribution.

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.¹⁴ \triangle

federalist-distplot}

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 3.15 showed the corresponding rootogram plots. Figure 3.22 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.

¹⁴On 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.

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

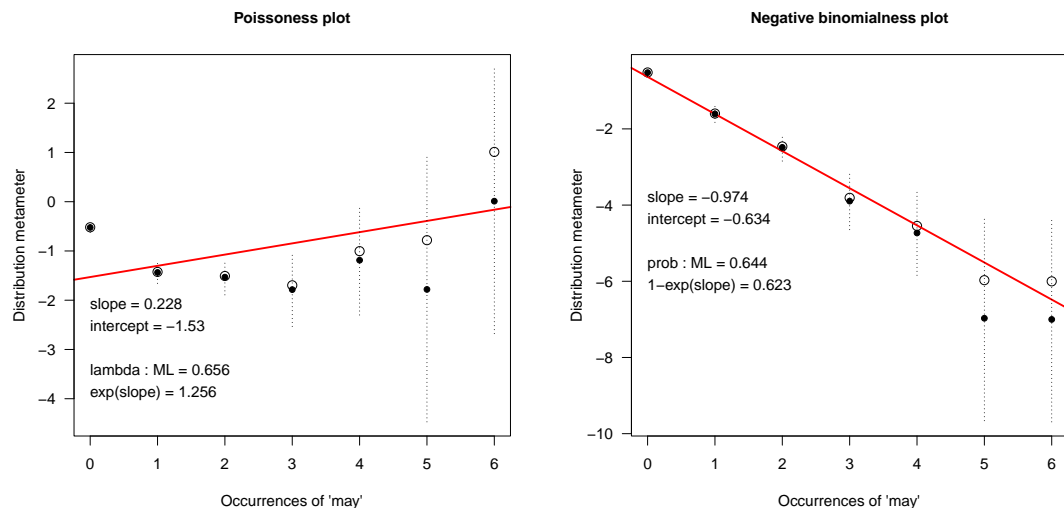


Figure 3.23: Diagnostic plots for the Federalist papers data. Left: Poissonness plot; right: negative binomialness plot.

△

3.6 Fitting discrete distributions as generalized linear models

{sec:fitglm}

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 3.5.5. 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 fit by `glm()`.

A clever approach by Lindsey and Mersch (1992), Lindsey (1995, §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

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

where N is the total frequency, π_k is the modeled probability of count k , $S(k)$ 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 which 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,¹⁵ 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 using several different models to handle overdispersion.

{tab:expfamily}

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

{ex:saxony2}

EXAMPLE 3.23: Families in Saxony

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

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

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

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

# brief model summaries
summarise(Sax.bin)

## Model Summary:
##           LR Chisq Df Pr(>Chisq) AIC BIC
## Sax.bin           97 11  6.98e-16  75 1.1

coef(Sax.bin)

## (Intercept)           Males
##    -0.06952         0.07690
```

¹⁵In 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()`).

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()`. [Done: Fixed Table 3.13 entry here](#)

```
# 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
##    -3.09692     0.06598     0.14021

summarise(glmlist(Sax.bin, Sax.dbin))

## Model Summary:
##          LR Chisq Df Pr(>Chisq)  AIC   BIC
## Sax.bin      97.0 11      0.00 75.0   1.1
## Sax.dbin     13.1 10      0.22 -6.9 -74.1
```

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
## 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 3.23 shows the rootogram for the double binomial, which can be compared with that for the binomial model shown in Figure 3.21. We can see that the fit is now quite good, particularly in the tails, where the term `YlogitY` gives additional weight.

```
with(results, rootogram(Families, fit.dbin, names=Males,
  xlab="Number of males"))
```

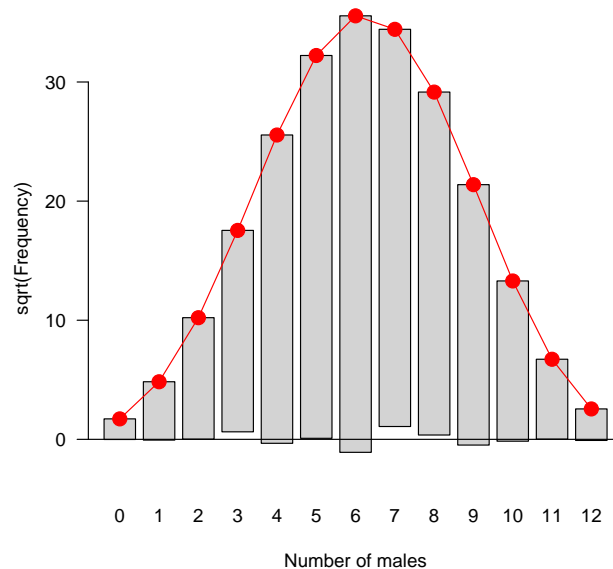



Figure 3.24: Rootogram for the double binomial model for the Saxony data.^{fig:sax-glm5}

TODO: Interpret the coefficient for $Y_{\text{logit}Y}$ in terms of dispersion??

△

3.7 Chapter summary

{sec:ch03-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 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()`.
- 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) when the data follows 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 Further reading

{sec:ch03-reading}

3.9 Lab exercises

{sec:ch03-labs}

- 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 $k = 0, 1, \dots, 10$.
 - With `xypplot()`, try the different plot formats using points connected with lines, as in Figure 3.8, or using points and lines down to the origin, as in the panels of Figure 3.9.
 - Also with `xypplot()`, 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.
 - Do the same in a multi-panel version, conditional on p .
- Use the data set `WomenQueue` to:
 - produce plots analogous to those shown in `sec:discrete-intro` (some sort of bar graph of frequencies)
 - check for goodness-of-fit to the binomial distribution using the `goodfit()` methods described in Section 3.3.2.
 -
- 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()`.]
 - Carry out the GOF test for this fixed binomial distribution. What is the ratio of χ^2/df ? What do you conclude?
 - 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.
 - Use the `plot.goodfit()` method to visualize these two models.
- 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 $n = 1$.
 - Use `goodfit()` to fit the geometric distribution. [Hint: use `type="nbinomial"`, but specify `size=1` as a parameter.]
 - Compare the negative binomial and the geometric models statistically, by a likelihood-ratio test of the difference between these two models.
 - Compare the negative binomial and the geometric models visually by hanging rootograms or other methods.
- 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?
6. 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 | 1540 | 2161 | 2310 | 1801 | 1077 | 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.
7. 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 the Poisson distribution.
- (d) What circumstances of scoring goals in soccer might cause these distributions to deviate from Poisson distributions?

8. 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)
```

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

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



10. The one-way table, `Depends` in `vcdExtra` and shown below gives the frequency distribution of the number of dependencies declared in 4983 R packages maintained on the CRAN distribution network on January 17, 2014. That is, there were 986 packages that had no dependencies, 1347 packages that depended on one other package, up to 2 packages that depended on 14 other packages.

TODO: Perhaps promote this table to an introductory example, leaving analysis to this exercise.

Depends	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Pkgs	986	1347	993	685	375	298	155	65	32	19	9	4	9	4	2

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

TODO: Cleanup local variables

```
remove(list=objects(pattern="\\.tab|\\.df|\\.fit"))
ls()

## [1] "Arbuthnot"      "Arthritis"      "BL1995"
## [4] "Bundesliga"    "Butterfly"      "chapters"
## [7] "chisq"          "Chisq"          "count"
## [10] "counts"        "CyclingDeaths"  "dat"
## [13] "DaytonSurvey"  "Depends"        "Diff"
## [16] "dp"            "education"      "exp"
## [19] "Exp"           "Families"       "Federalist"
## [22] "fm"            "Freq"           "Geissler"
## [25] "GOF"           "gplt"           "GSS"
## [28] "GSStab"        "HairEye"        "HairEyeColor"
## [31] "HEC"           "HorseKicks"     "horsetab"
## [34] "HSE"           "includeonly"    "includes"
## [37] "JobSat"        "k"              "knitrSet"
## [40] "lambda"        "logit2p"        "Males"
```

```
## [43] "mean"      "mu"        "mycol"
## [46] "n"         "nk"        "nk1"
## [49] "NP"        "op"        "ord"
## [52] "p"         "phat"      "plt"
## [55] "pred"      "predicted" "Prob"
## [58] "results"   "Sax.bin"   "Sax.dbin"
## [61] "Saxony"    "Saxony11"  "SpaceShuttle"
## [64] "spar"      "tab"       "tab1"
## [67] "tab2"      "UCB"       "UKSoccer"
## [70] "var"       "weight"    "WeldonDice"
## [73] "WomenQueue" "x"        "XL"
## [76] "XN"        "XP"        "y"
```

Chapter 4

Two-way contingency tables

{ch:twoway}

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.

4.1 Introduction

{sec:twoway-intro}

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 which 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 which 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 which 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**

plots (Section 4.6) are more general methods for depicting the pattern of associations in any two-way tables. 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 which occur in contingency table analysis, we begin with several examples.

{ex:berkeley1}

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` 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 the same, or whether the difference in proportions admitted is zero. \triangle

{tab:berk22}

Table 4.1: Admissions to Berkeley graduate programs

	Admitted	Rejected	Total	% Admit	Odds(Admit)
Males	1198	1493	2691	44.52	0.802
Females	557	1278	1835	30.35	0.437
Total	1755	2771	4526	38.78	0.633

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.

{ex:haireye1}

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 not the only categories possible.¹ 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.

¹If 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.

ab:hairdat}

Table 4.2: Hair-color eye-color data

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

△

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.

{ex:mental1}

EXAMPLE 4.3: Mental impairment and parents' SES

Srole *et al.* (1978, p. 289) gave the data in Table ?? 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, §10.5.3), Goodman (1979), and Haberman (1979, p. 375).

There are five categories of SES 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.

Table 4.3: Mental impairment and parents' SES

SES	Mental impairment			
	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

Although there may be an overall association between these two variables, more powerful and focused tests are available when we treat these variables as *ordinal*, as we will see in Section 4.2.4.

△

{ex:arthritis1}

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*.

Table 4.4: Arthritis treatment data

{tab:arthri

Treatment	Sex	Improvement			Total
		None	Some	Marked	
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

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 $N = \{n_{ij}\}$ 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 .

{tab:rbyc}

Table 4.5: The $r \times c$ contingency table

Row Category	Column category				Total
	1	2	\cdots	c	
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
##
##      Gender | 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

{sec:twoway}

The 2×2 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.

Odds and odds ratios

{sec:twoway-odds}

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** 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, §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 which varies *additively* around 0, called the **log odds** or **logit**:

{eq:logit}

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

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 ??, 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	0.05263	-2.944
##	2	0.10	0.11111	-2.197
##	3	0.25	0.33333	-1.099
##	4	0.50	1.00000	0.000
##	5	0.75	3.00000	1.099
##	6	0.90	9.00000	2.197
##	7	0.95	19.00000	2.944

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 which is symmetric about 0. The **log odds ratio**, symbolized by ψ , is just the difference between the logits for Groups 1 and 2:

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

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:

$$\hat{\theta} = \frac{p_1/(1 - p_1)}{p_2/(1 - p_2)} = \frac{n_{11}/n_{12}}{n_{21}/n_{22}} = \frac{n_{11}n_{22}}{n_{12}n_{21}} . \quad (4.2) \quad \{\text{eq:soddsratio}\}$$

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 $n \rightarrow \infty$, 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)

$$\text{ASE}_{\log(\theta)} \equiv \hat{s}(\hat{\psi}) = \left\{ \frac{1}{n_{11}} + \frac{1}{n_{12}} + \frac{1}{n_{21}} + \frac{1}{n_{22}} \right\}^{1/2} = \left\{ \sum \sum n_{ij}^{-1} \right\}^{1/2} \quad (4.3) \quad \{\text{eq:aselogtheta}\}$$

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 $\psi = \log(\theta)$,²

$$\exp\left(\hat{\psi} \pm z_{1-\alpha/2} \hat{s}(\hat{\psi})\right) .$$

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 Treatment 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 4.5: Arthritis treatment

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

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

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

round(100*prop.table(Art.tab, 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.tab)

##           X^2 df  P(> X^2)
## Likelihood Ratio 13.530  2 0.0011536
## Pearson          13.055  2 0.0014626
##
## Phi-Coefficient   : 0.394
## Contingency Coeff.: 0.367
## Cramer's V       : 0.394
```

△

²Note that $\hat{\theta}$ is 0 or ∞ if any $n_{ij} = 0$. Haldane (1955) and Gart and Zweifl (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.

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.

{ex:mental2}

EXAMPLE 4.6: 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.tab <- xtabs(Freq ~ ses + mental, data=Mental)
assocstats(mental.tab) # 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   : 0.166
## Contingency Coeff.: 0.164
## Cramer's V        : 0.096

CMHtest(mental.tab) # CMH tests

## Cochran-Mantel-Haenszel Statistics for ses by mental
##
##              AltHypothesis Chisq Df    Prob
## cor          Nonzero correlation  37.2  1 1.09e-09
## cmeans       Col mean scores differ 40.3  5 1.30e-07
## rmeans       Row 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). 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

{sec:Sample}

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.

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 are calculated by assigning integer scores, $b_i = i$ to the column categories. Figure 4.1(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

This is reflected in the `CMHtest()` output shown below. **TODO: Something wrong here: does `CMHtest()` get rows/cols mixed up? Would be nice to calculate col means also.**

```
CMHtest (cmhdemo1)
```

```
## Cochran-Mantel-Haenszel Statistics
##
##               AltHypothesis  Chisq Df    Prob
## cor             Nonzero correlation    0.0  1 1.00e+00
## cmeans   Col mean scores differ    0.0  2 1.00e+00
## rmeans   Row 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.

```
sieve (cmhdemo1, shade=TRUE, main="General association",
      gp = shading_sieve(interpolate = 0, lty = c("solid", "longdash")))
sieve (cmhdemo2, shade=TRUE, main="Linear association",
      gp = shading_sieve(interpolate = 0, lty = c("solid", "longdash")))
```

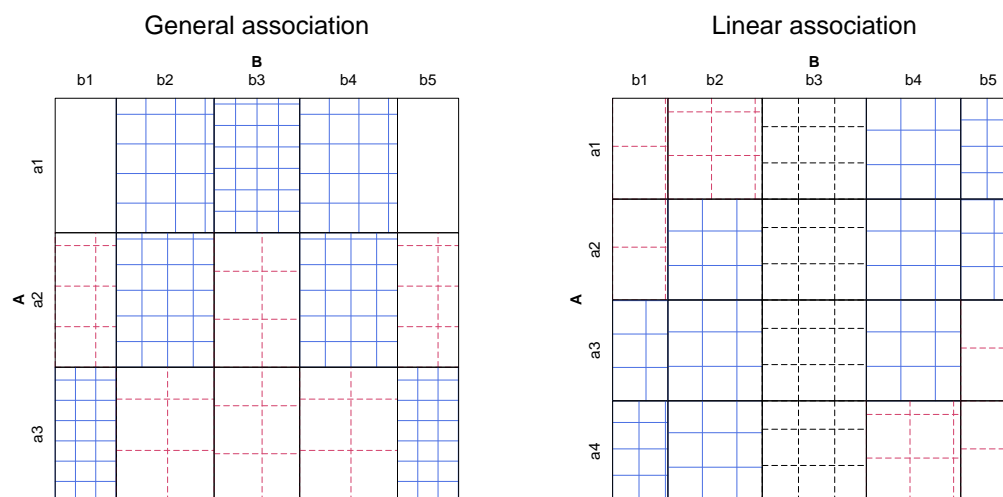


Figure 4.1: Sieve diagrams for two patterns of association: Left: General association; right: Linear association

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 4.1(right) for a visual representation of this pattern.

	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

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.

```
CMHtest(cmhdemo2)
```

```
## Cochran-Mantel-Haenszel Statistics
##
##               AltHypothesis  Chisq  Df    Prob
## cor             Nonzero correlation  10.6   1 0.00111
## cmeans   Col mean scores differ  10.7   3 0.01361
## rmeans   Row 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.

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 4.7: Arthritis treatment

The statements below request a stratified analysis of the arthritis treatment data with CMH tests, controlling for sex. Essentially, the analysis is carried out separately for males and females.

The table `Art.tab2` is constructed as a three-way table, with `sex` as the last dimension.

```
Art.tab2 <- xtabs(~Treatment + Improved + Sex, data=Arthritis)
Art.tab2

## , , 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
```

`assocstats()` only applies to two-way tables, so we use `apply()` to run it for each level of `Sex`. `CMHtest()` is designed for such stratified tables, and uses all dimensions after the first two as strata.

```
apply(Art.tab2, MARGIN=3, FUN=assocstats)

## $Female
##
##           X^2 df  P(> X^2)
## Likelihood Ratio 11.731  2 0.0028362
## Pearson          11.296  2 0.0035242
##
## Phi-Coefficient   : 0.438
## Contingency Coeff.: 0.401
## Cramer's V        : 0.438
##
```

```
## $Male
##               X^2 df P(> X^2)
## Likelihood Ratio 5.8549  2 0.053532
## Pearson          4.9067  2 0.086003
##
## Phi-Coefficient   : 0.443
## Contingency Coeff.: 0.405
## Cramer's V        : 0.443
```

Note that even though the strength of association (ϕ -coefficient) 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(Art.tab2)
```

```
## $`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
## cmeans    Col mean scores differ  10.9  1 0.000944
## rmeans    Row 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
## cmeans    Col mean scores differ   3.71  1 0.0540
## rmeans    Row mean scores differ   4.71  2 0.0949
## general      General association   4.71  2 0.0949

apply(Art.tab2, 3, sum)

## Female   Male
##      59    25
```

△

4.3.1 Assessing homogeneity of association

{sec:twoway-homog}

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 , ...

In the case of a 3-way table, this can be stated as the *loglinear model* of no three-way association, $[AB][AC][BC]$. This notation (described in Section ??) lists only the high-order association terms in a linear model for log frequency.

{ex:berkeley1a}

EXAMPLE 4.8: Berkeley admissions

Here we illustrate the use of Woolf's test for the UCBA admissions 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(UCBA admissions)

##
## Woolf-test on Homogeneity of Odds Ratios (no 3-Way
## assoc.)
##
## data: UCBA admissions
## X-squared = 17.9, df = 5, p-value = 0.003072
```

△

{ex:arthrit4}

EXAMPLE 4.9: Arthritis treatment

For the arthritis data, homogeneity means that there is no three-way Treatment * Improved * Sex association. That is, the association between treatment and outcome (improve) is the same for both men and women. This hypothesis can be stated as the loglinear model,

$$\{eq:STO2\} \quad [SexTreatment] [SexImproved] [TreatmentImproved] . \quad (4.4)$$

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=Art.tab2)

## Call:
## loglm(formula = ~(Treatment + Improved + Sex)^2, data = Art.tab2)
##
## Statistics:
##
##              X^2 df P(> X^2)
## Likelihood Ratio 1.704  2  0.4266
## Pearson          1.134  2  0.5673
```

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 `loglm()` is clearly non-significant, so we cannot reject homogeneity of association. △

4.4 Fourfold display for 2 x 2 tables

{sec:twoway-fourfold}

The *fourfold display* is a special case of a *radial diagram* (or “polar area chart”) designed for the display of 2×2 (or $2 \times 2 \times k$) 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.

{ex:berkeley2}

EXAMPLE 4.10: Berkeley admissions

Figure 4.2(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 4.8) indicating that males were nearly twice as likely to be admitted.

```
fourfold(Berkeley, std="ind.max") # unstandardized
fourfold(Berkeley, margin=1)      # equating gender
```

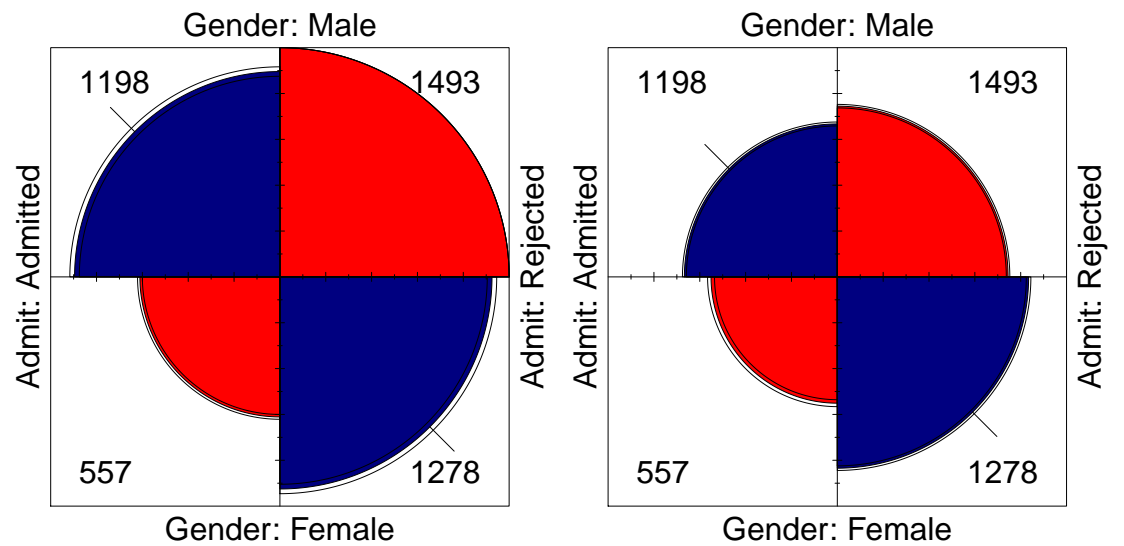


Figure 4.2: Fourfold displays for the Berkeley admission data. Left: unstandardized; right: equating the proportions of males and females

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 4.2(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

{tab:berkro

	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 4.3.

```
fourfold(Berkeley) # standardize both margins
```

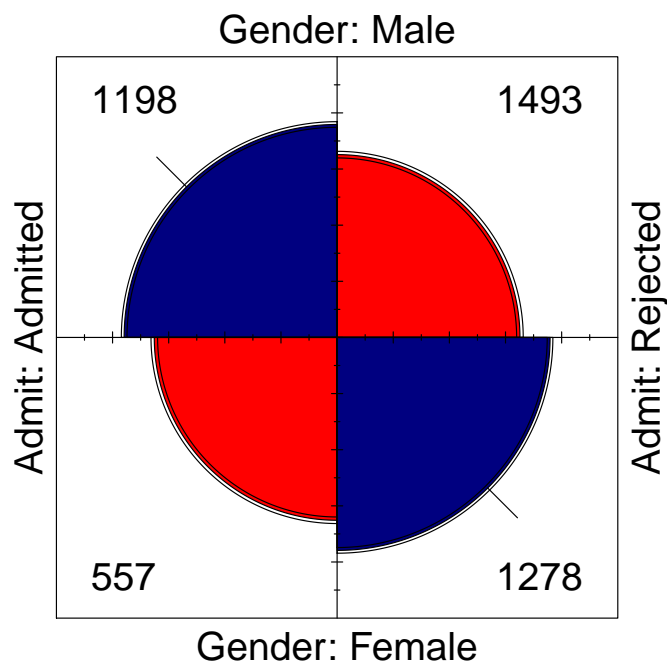


Figure 4.3: Fourfold display for Berkeley admission data with margins for gender and admission equated. The area of each quadrant shows the standardized frequency in each cell. fig:berk-fourfold3

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 4.3 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 4.3, $\hat{\psi} = \log \hat{\theta} = .6104$, and $\hat{s}(\hat{\psi}) = 0.0639$, so the 95% limits for θ are $\{1.624, 2.087\}$, as shown by the calculations below.

```
summary(oddsratio(Berkeley))

##      Log Odds Ratio Std. Error z value Pr(>|z|)
## [1,]      0.6104      0.0639      9.55  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

exp(.6103 + c(-1, 1) * qnorm(.975) * 0.06398)

## [1] 1.624 2.087
```

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 4.3 are shown in Table 4.7. **TODO: Re-calculate this table for $\alpha = 0.05$**

4.4.2 fourfold() details

TODO: Is this necessary?

Table 4.7: Odds ratios and equivalent tables for confidence rings.

{tab:berkod

	Odds Ratio	Standardized Table		Equivalent Frequencies	
Lower limit	1.562	0.555	0.445	1157.2	1533.8
		0.445	0.555	597.8	1237.2
Data	1.841	0.576	0.424	1198.0	1493.0
		0.424	0.576	557.0	1278.0
Upper limit	2.170	0.596	0.404	1237.8	1453.2
		0.404	0.596	517.2	1317.8

4.4.3 Stratified analysis for $2 \times 2 \times k$ tables

{sec:twoway

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

The admissions data shown in Figure 4.2 and Figure 4.3 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 4.4, stratified by department.

```
# fourfold display
UCB <- aperm(UCBAdmissions, c(2, 1, 3))
fourfold(UCB, mflow=c(2, 3))
```

TODO: Figure out how to trim these grid figures. Here, I’m using `out.extra='trim=...'`, but maybe `pdfcrop` is easier.

Surprisingly, Figure 4.4 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 differ 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 4.4 are joint³ 95% intervals for θ_c , $c = 1, \dots, k$.

This result, which contradicts the display for the aggregate data in Figure 4.2, is a nice example of *Simpson’s paradox*⁴, 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

³For multiple-strata plots, `fourfold()` by default adjusts the significance level for multiple testing, using Holm’s (1979) method provided by `p.adjust()`.

⁴Simpson’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 4.4) while in the aggregate data $\theta > 1$.

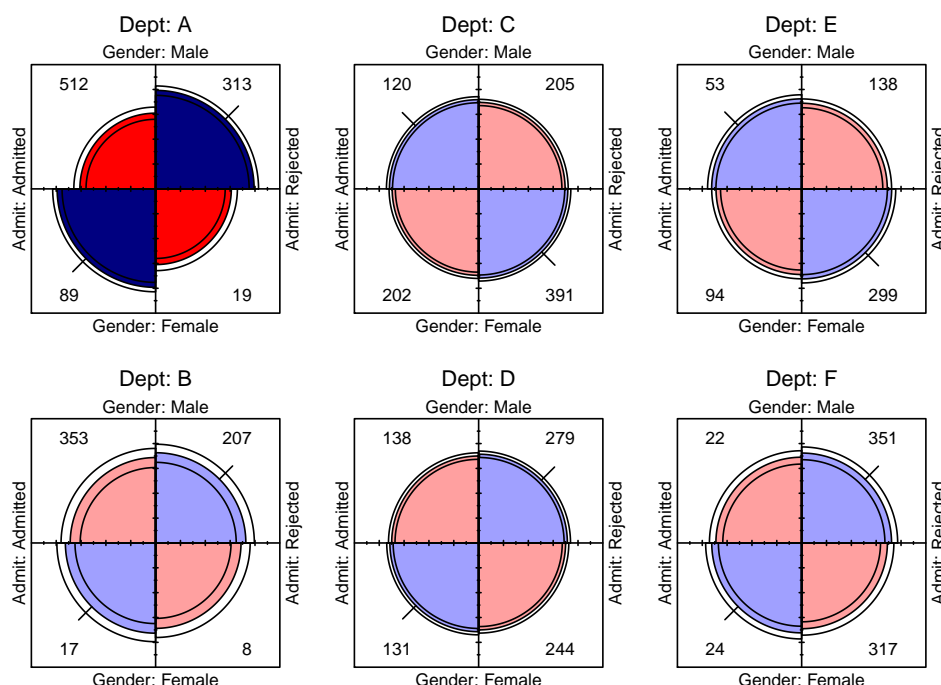


Figure 4.4: Fourfold displays for Berkeley admissions data, stratified by department. The more intense shading for Dept. A indicates a significant association. fig:berk-fourfold4

men and women are equally likely to apply in each field.⁵

Visualization principles

TODO: Move this to Ch. 1 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 4.4, 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 which also greatly facilitates controlled comparison.

Another principle is *visual impact*—we want the important features of the display to be easily distinguished from the less important (Tukey, 1993). Figure 4.4 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.

{ex:wheeze1}

⁵This explanation ignores the possibility of structural bias against women, e.g., lack of resources allocated to departments that attract women applicants.

EXAMPLE 4.11: 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 Snowden (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.⁶ These data are available in the `CoalMiners` data in `vcd`, a $2 \times 2 \times 8$ frequency table.

```
data("CoalMiners", package="vcd")
str(CoalMiners)

## table [1:2, 1:2, 1:8] 23 105 9 1654 54 ...
## - attr(*, "dimnames")=List of 3
## ..$ Wheeze      : chr [1:2] "W" "NoW"
## ..$ Breathlessness: chr [1:2] "B" "NoB"
## ..$ Age         : chr [1:8] "25-29" "30-34" "35-39" "40-44" ...
```

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 4.5. 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(CoalMiners, mfc = c(2, 4))
```

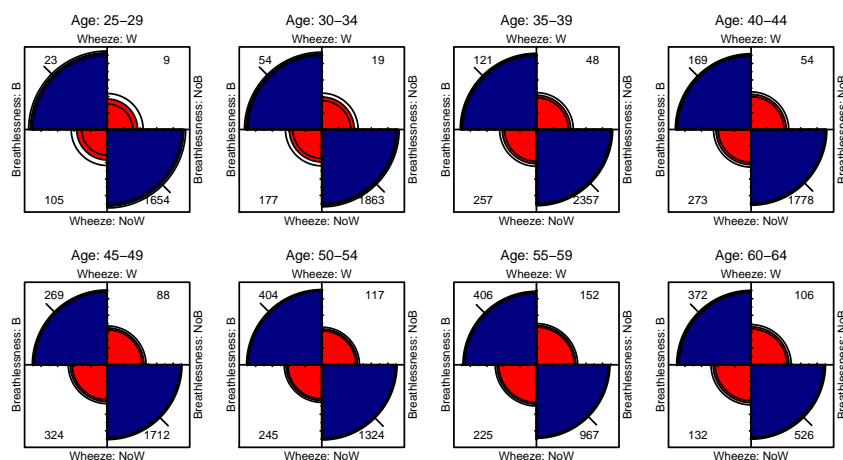


Figure 4.5: Fourfold display for `CoalMiners` data, both margins equated^{fig:coalminer1}

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.

⁶A ninth group, aged 20-24 has been omitted from these analyses.

The `oddsratio()` function in `vcd` calculates odds ratios for $2 \times 2(\times k)$ tables. 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.

```
oddsratio(CoalMiners)

## 25-29 30-34 35-39 40-44 45-49 50-54 55-59 60-64
## 3.695 3.398 3.141 3.015 2.782 2.926 2.441 2.638

oddsratio(CoalMiners, log=FALSE)

## 25-29 30-34 35-39 40-44 45-49 50-54 55-59 60-64
## 40.26 29.91 23.12 20.38 16.15 18.66 11.48 13.98
```

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.

```
lodds <- oddsratio(CoalMiners)
plot(lodds, lwd=2, cex=1.25, pch=16,
     xlab = "Age Group",
     main = "Breathlessness and Wheeze in Coal Miners")
age <- seq(25, 60, by = 5)
mod <- lm(lodds ~ poly(age, 2))
lines(fitted(mod), col = "red", lwd=2)
```

In Figure ??, 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 ...

```
# quadratic term NS
summary(mod)

##
## Call:
## lm(formula = lodds ~ poly(age, 2))
##
## Residuals:
## 25-29 30-34 35-39 40-44 45-49 50-54 55-59
## 0.0219 -0.0128 -0.0438 0.0213 -0.0558 0.2085 -0.1928
## 60-64
## 0.0534
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)    3.0045     0.0473   63.46 1.8e-08 ***
## poly(age, 2)1  -1.0080     0.1339   -7.53 0.00066 ***
## poly(age, 2)2    0.2304     0.1339    1.72 0.14594
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.134 on 5 degrees of freedom
## Multiple R-squared:  0.923, Adjusted R-squared:  0.892
## F-statistic: 29.8 on 2 and 5 DF, p-value: 0.00167
```

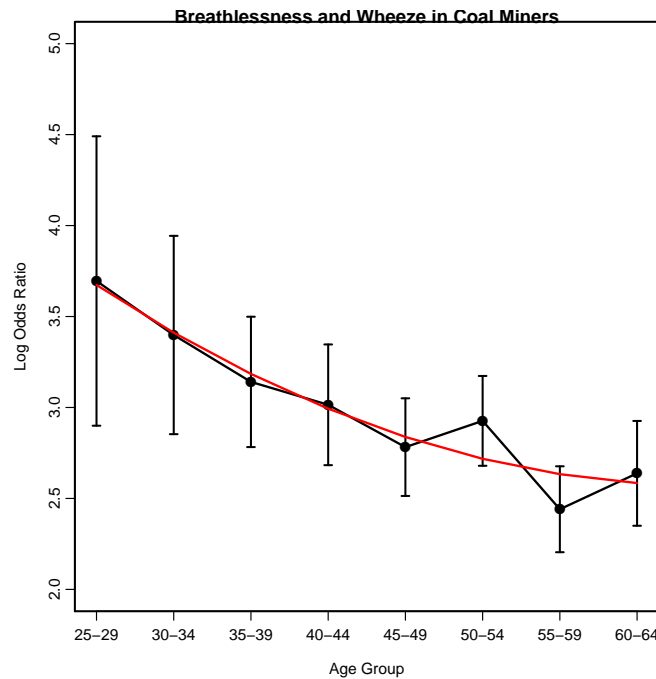


Figure 4.6: Log odds plot for the CoalMiners data. The smooth curve shows a quadratic fit to age. fig:coalminers

△

4.5 Sieve diagrams

{sec:twoway-sieve}

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.

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 4.7 (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

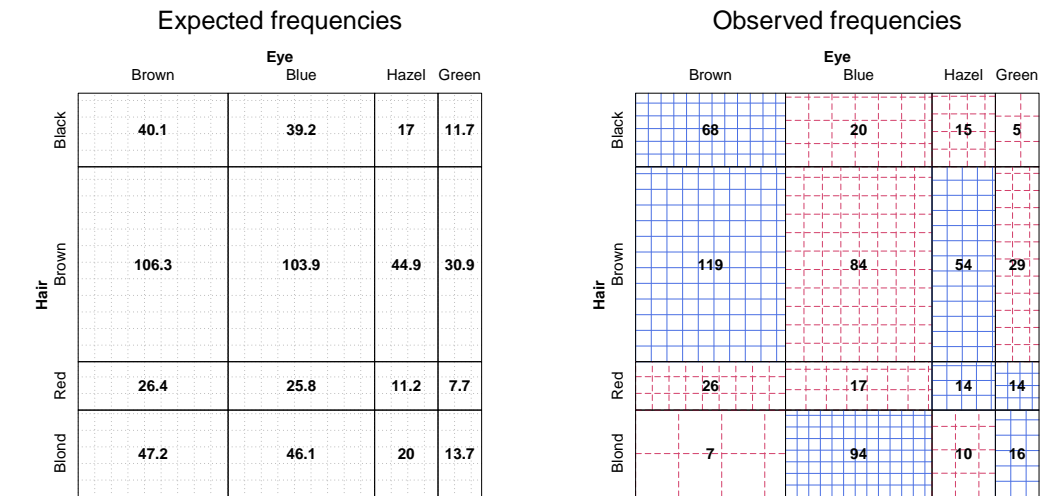


Figure 4.7: Sieve diagrams for Hair Eye color data. Left: expected frequencies shown in cells as numbers and the number of boxes; right: observed frequencies shown in cells.

Figure 4.7 (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 which implicitly compares observed and expected frequencies as shown in Figure 4.7 (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 4.7 (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.⁷

{ex:haireye2}

⁷Positive residuals are also shown by solid lines, negative residuals by broken lines, so that they may still be distinguished in monochrome versions.

EXAMPLE 4.12: Hair color and eye color

The sieve diagram for hair color and eye color shown in Figure 4.7 (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. \triangle

{ex:vision1}

EXAMPLE 4.13: Visual acuity

In World War II, all workers in the U.K. 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 (3242 men, 7477 women) aged 30-39.

Figure 4.8 shows the sieve diagram for data from the larger sample of women (Kendall and Stuart (1961, Table 33.5), Bishop *et al.* (1975, p. 284)). The `VisualAcuity` data is a frequency data frame and we first convert it to table form (`VA.tab`), a $4 \times 4 \times 2$ table to re-label the variables and levels. **TODO: Make this an exercise in Ch. 2**

```
# re-assign names/dimnames
data("VisualAcuity", package="vcd")
VA.tab <- xtabs(Freq ~ right + left + gender, data=VisualAcuity)
dimnames(VA.tab)[1:2] <- list(c("high", 2, 3, "low"))
names(dimnames(VA.tab))[1:2] <- paste(c("Right", "Left"), "eye grade")
#str(VA.tab)
```

```
sieve(VA.tab[, "female"], shade=TRUE)
```

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 $\pm 1, \pm 2, \dots$ 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 ?? in Chapter 5) 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. **TODO: Add forward references to section(s) on models for square tables.** \triangle

4.5.1 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).

{twoway-sieve-larger}

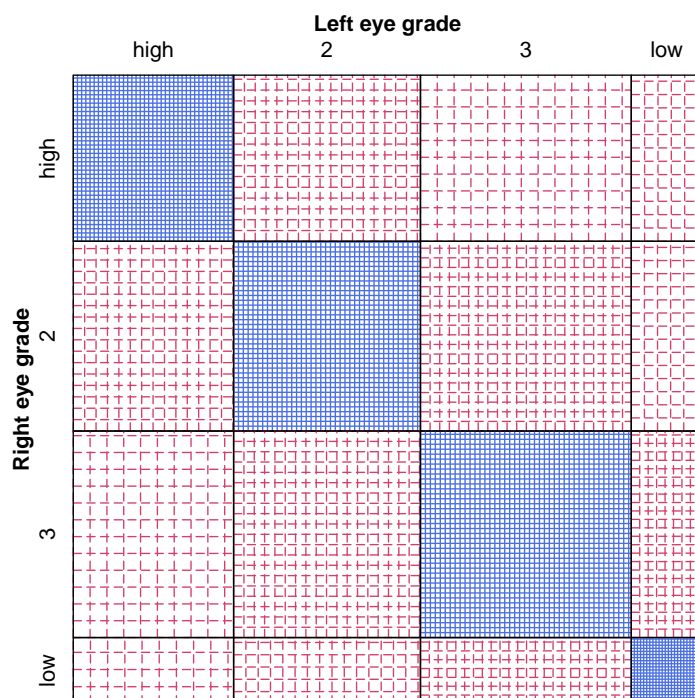


Figure 4.8: Vision classification for 7477 women in Royal Ordnance factories fig:VA-sieve2

```
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 ?? in 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*.⁸

```
sieve(Freq ~ right + left | gender, data = VisualAcuity, shade=TRUE)
```

In Figure 4.9, 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. The methods described in Section 4.3.1 can be used to test the hypothesis of homogeneity of association, and loglinear models described in Chapter ?? provide specific tests of hypotheses of *symmetry*, *quasi-independence* and other models for structured associations.

{ex:berkeley3}

EXAMPLE 4.14: Berkeley admissions

⁸An equivalent plot, but one labeled more nicely, as in Figure 4.8 can be produced from the `VA.tab` table using `sieve(VA.tab, shade=TRUE, condvar='gender')`.

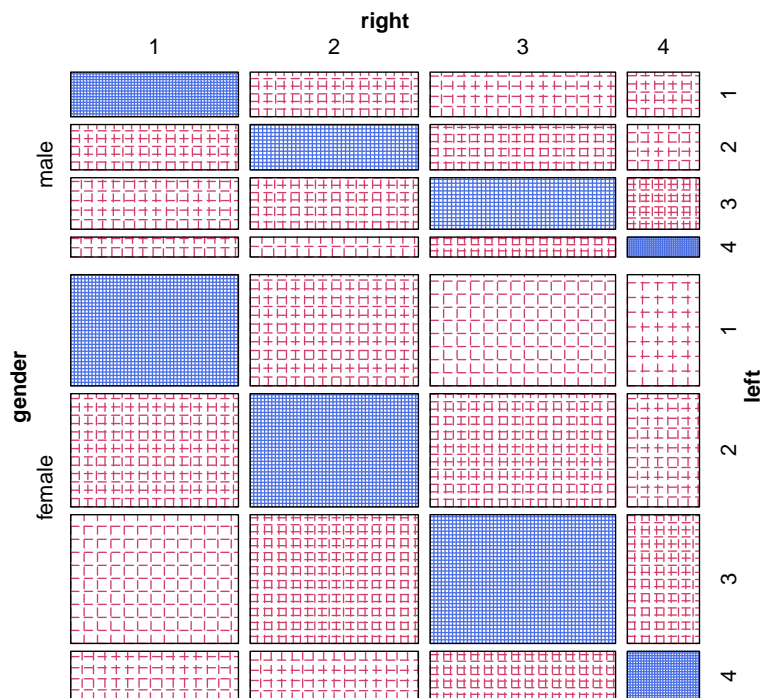


Figure 4.9: Sieve diagram for the three-way table of VisualAcuity, conditioned on gender. fig:VA-sieve3

This example illustrates some additional flexibility of sieve plots with the strucplot framework, using the Berkeley admissions data. The left panel of Figure 4.10 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
UCB <- aperm(UCBAdmissions, c(3,1,2))
dimnames(UCB)[[3]] <- c("M", "F") # abbreviate for display
sieve(UCB, shade=TRUE, pop=FALSE)
labeling_cells(text = UCB, gp_text = gpar(fontface = 2))(UCB)
```

In the right panel of Figure 4.10, the three-way table was first permuted to make *Dept* the first splitting variable. Each 2×2 table of *Admit* by *Gender* then appears, giving a sieve diagram version of what we showed earlier in fourfold displays (Figure 4.4). The function `labeling_cells()` is used here to write the cell frequency in each rectangle.

Finally, for tables of more than two dimensions, there is a variety of different models for “independence,” and the the strucplot framework allows these to 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

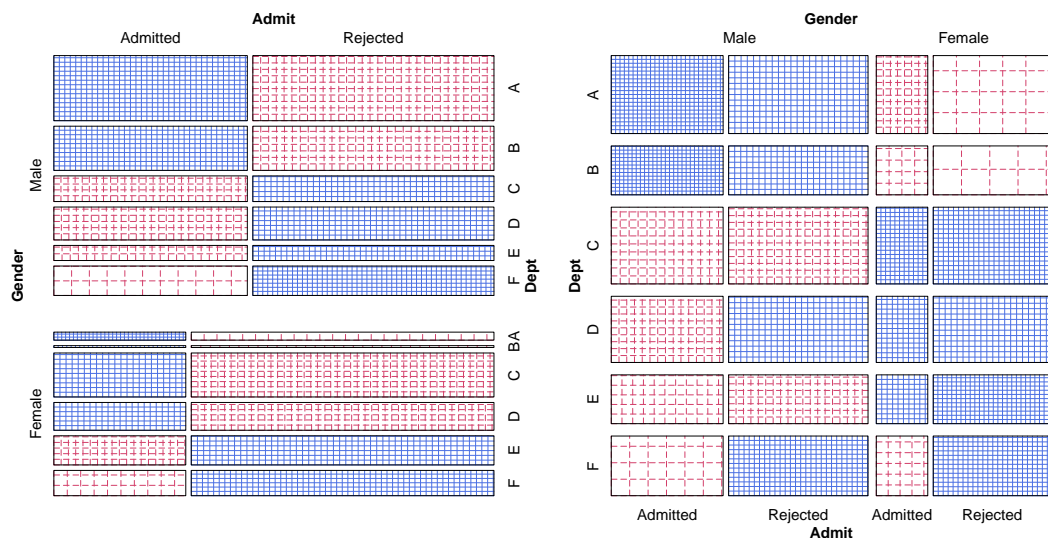


Figure 4.10: Sieve diagrams for the three-way table of the Berkeley admissions data. Left: Admit by Dept, conditioned on Gender; right: Dept re-ordered as the first splitting variable. fig:berkeley-sieve

as shown below, giving the plot in Figure 4.11.

```
UCB2 <- aperm(UCBAAdmissions, c(3,2,1))
sieve(UCB2, shade=TRUE, expected=~Admit*Gender + Dept,
       split_vertical=c(FALSE,TRUE,TRUE))
```

In terms of the loglinear models discussed in Chapter 5, this is equivalent to fitting the model of *joint independence*, $[AdmitGender][Dept]$. Figure 4.11 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

{sec:twoway-assoc}

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

$$d_{ij} = \frac{n_{ij} - m_{ij}}{\sqrt{m_{ij}}} = \text{Pearson residual}, \quad \chi^2 = \sum_{ij} (d_{ij})^2 \quad (4.5) \quad \text{{eq:Pearson-residual}}$$

In the association plot, each cell is shown by a rectangle, having:

- (signed) height $\sim d_{ij}$
- width $= \sqrt{m_{ij}}$.

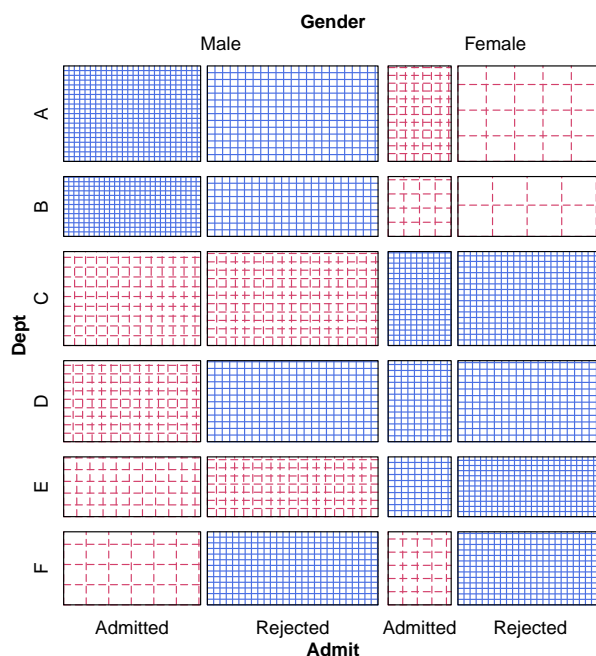


Figure 4.11: Sieve diagram for the Berkeley admissions data, fitting the model of joint independence, $\text{Admit} * \text{Gender} + \text{Dept}$

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 ($d_{ij} = 0$) 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).

```
haireye <- margin.table(HairEyeColor, 1:2)
assoc(haireye, shade=TRUE)
```

Figure 4.12 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 Eqn. (4.5), using a simple scale shown in the legend, where residuals $|d_{ij}| > 2$ are shaded blue or red depending on their sign and residuals $|d_{ij}| > 4$ 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 d_{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.

```
assoc(HairEyeColor, shade=TRUE)
```

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 4.13. **TODO: Perhaps combine these two figures into two panels, side by side.** 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.

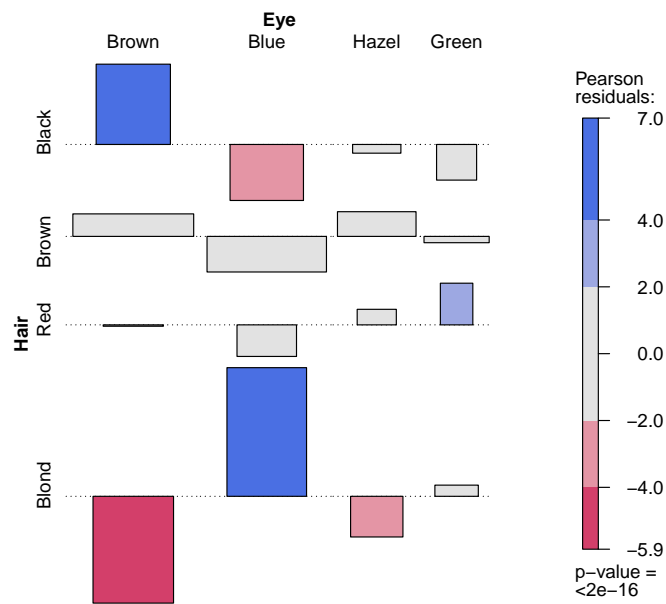


Figure 4.12: Association plot for the two-way table of hair color by eye color.^{fig:HE-assoc}

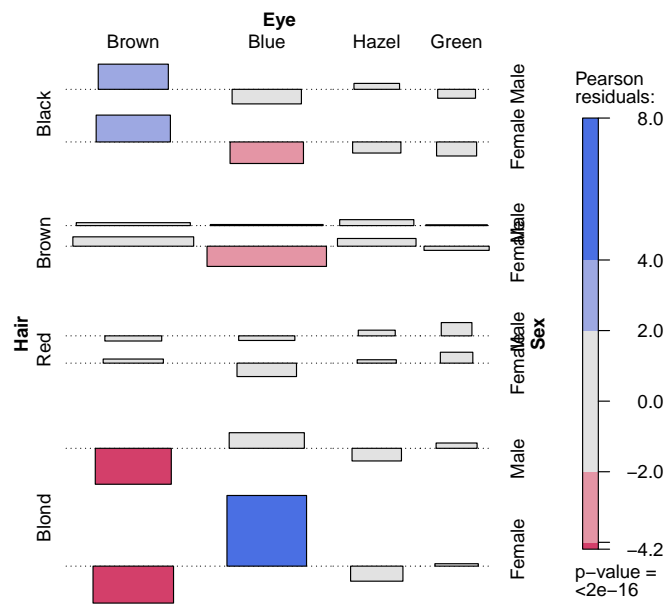


Figure 4.13: Association plot for the three-way table of hair color by eye color by sex.^{fig:HE-assoc2}

4.7 Observer agreement

{sec:twoway-agree}

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 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.

{ex:sexisfun1}

EXAMPLE 4.15: 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
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”. △

{ex:MS1}

EXAMPLE 4.16: 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

{sec:agreemeas}

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.

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` package.

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:

$$\kappa = \frac{P_o - P_c}{1 - P_c} . \quad (4.6)$$

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 .

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, one-category difference given weight 1/2. This typically makes sense only when the categories are *ordered*, as in severity of diagnosis.

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

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

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

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

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

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

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

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
## Unweighted 0.129 0.0686 1.89
## Weighted   0.237 0.0783 3.03

confint(Kappa(SexualFun))

##
## Kappa           lwr      upr
## Unweighted -0.00512 0.2638
## Weighted    0.08388 0.3909
```

4.7.2 Observer Agreement Chart

{sec:twoway:Bangdiwal

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.

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 4.14. The large rectangle shows the maximum possible agreement, given the marginal totals. Thus, a visual impression of the strength of agreement is given by

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

(4.7) {eq:bangb}

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="Unweighted")
```

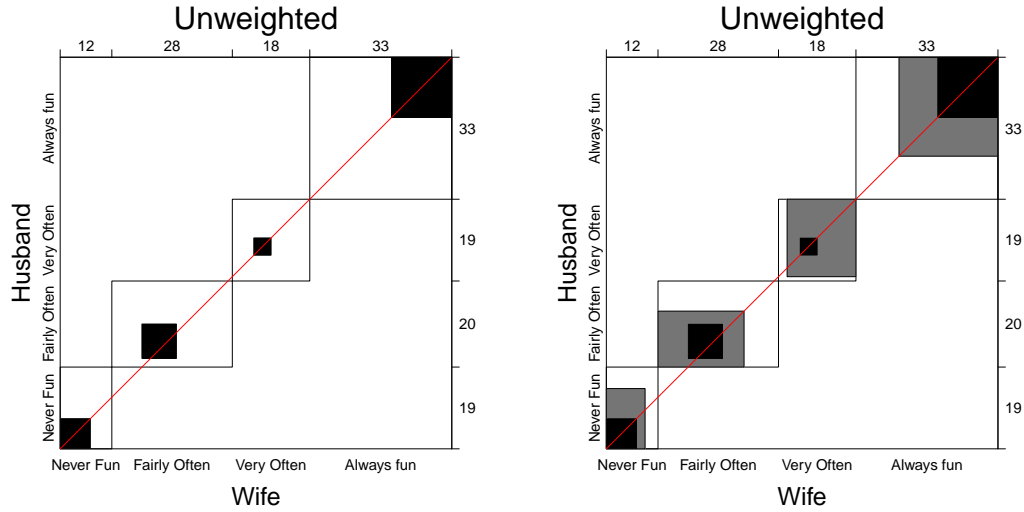


Figure 4.14: Agreement charts for husbands' and wives' sexual fun. Left: unweighted chart, showing only exact agreement; right: weighted chart, using weight $w_1 = 8/9$ for a one-step disagreement.

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, \dots, w_b is applied to the cell frequencies as shown schematically below:

$$\begin{array}{ccccccc}
 & & n_{i-b,i} & & & & w_b \\
 & & \vdots & & & & \vdots \\
 n_{i,i-b} & \cdots & n_{i,i} & \cdots & n_{i,i+b} & \Leftarrow & w_b \cdots 1 \cdots w_b \\
 & & \vdots & & & & \vdots \\
 & & n_{i-b,i} & & & & w_b
 \end{array}$$

These weights are incorporated in the agreement chart (right panel of Figure 4.14) by successively lighter shaded rectangles whose size is proportional to the sum of the cell frequencies, denoted 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
##           0.1465      0.4982
```

```
{ex:mammograms}
```

EXAMPLE 4.17: 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")
```

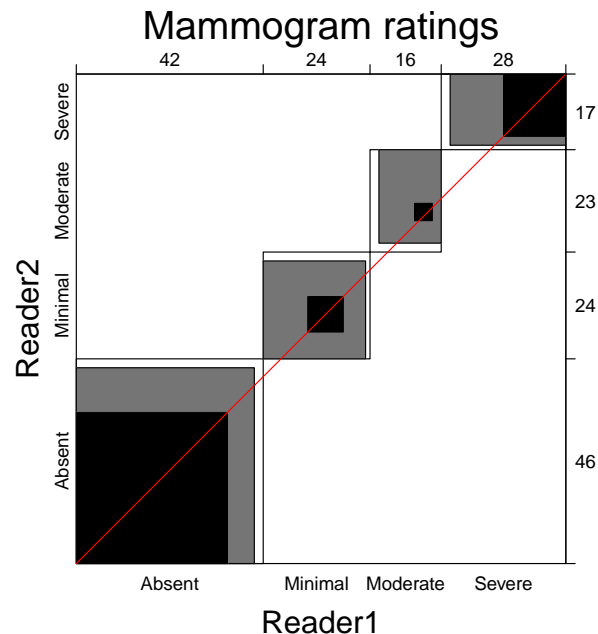


Figure 4.15: Agreement plot for the Mammograms data.^{fig:mammograms1}

The agreement plot in Figure ?? shows substantial agreement among the two raters, particularly when one-step disagreements are taken into account. Careful study of this graph shows that 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
##           0.4272      0.8366
```

△

⁹In 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.

4.7.3 Observer bias in agreement

{sec:twoway}

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.

{ex:MS2}

EXAMPLE 4.18: Diagnosis of MS patients

Agreement charts for both patient samples in the `MSPatients` data are shown in Figure 4.15. The `agreementplot()` function only handles two-way tables, so we do these separately by indexing on the last dimension (*Patients*).

```
agreementplot(MSPatients[,,"Winnipeg"], main="Winnipeg patients")
agreementplot(MSPatients[,,"New Orleans"], main="New Orleans patients")
```

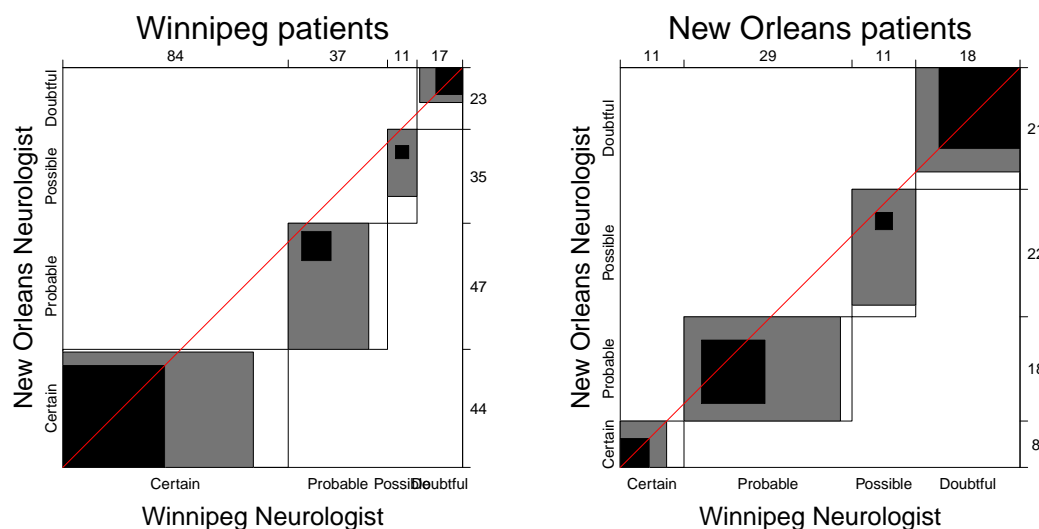


Figure 4.16: Weighted agreement charts for both patient samples in the `MSPatients` data. Departure of the middle rectangles from the diagonal indicates lack of marginal homogeneity.

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 `fig:MS-agree` 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      0.2721           0.7381
## NewOrleans    0.2854           0.8223

```

△

4.8 Trilinear plots

{sec:twoway-trilinear

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 coordinate corresponds 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. For instance, Figure 4.16 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"))
ggtern(data = DATA,
  mapping = aes(x=C, y=A, z=B,
    label=id, colour=id)) +
  geom_point(aes(size=2)) +
  geom_text(vjust=-.5, size=8) +
  theme_tern_rgbw() +
  theme(plot.margin=unit(c(0,0,0,0),"mm")) +
  guides(size = "none")

```

Note that each apex corresponds to 100% of the labeled variable, and the percentage of this variable decrease 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/wiki/Ternary_plot. Briefly, let $P(a,b,c)$ represent the three components normalized so

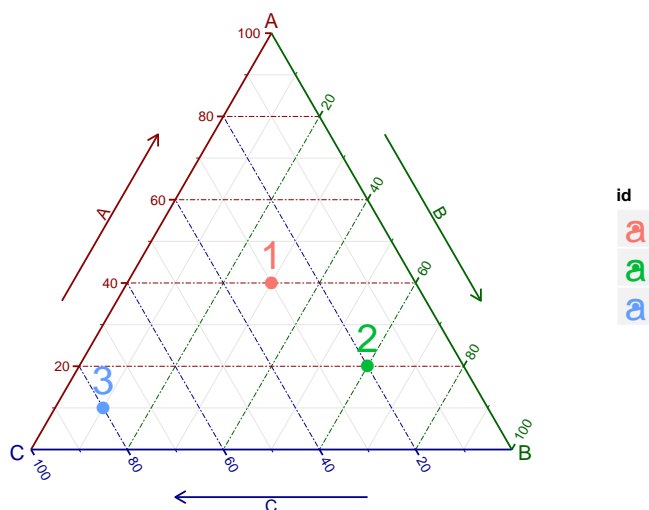


Figure 4.17: A trilinear plot showing three points, for variables A, B, C.^{fig:tripdemo2}

that $a+b+c = 1.0$. If the apex corresponding to Point A in Figure 4.16 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 `tripplot()` function in the `TeachingDemos` package, and also in the `ggtern` 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()`.

For example, the following code¹⁰ creates a data frame `DATA` containing 100 uniformly distributed random points. It uses `stat_density2d()` to draw contours of the densities of the points in the trilinear space.

```
set.seed(1)
DATA <- data.frame(x = runif(100),
                  y = runif(100),
                  z = runif(100))
plot <- ggtern(data = DATA,
              aes(x, y, z))
```

¹⁰This example was taken from the `ggtern` web site, <http://ggtern.com/2013/12/12/patched-density-functions-2/>.

```
plot + stat_density2d(method = "lm", fullrange = T,
  n = 200, geom = "polygon",
  aes(fill = ..level..,
    alpha = ..level..)) +
  geom_point() +
  theme_tern_rgbw() +
  labs(title = "Uniform data with density contours") +
  scale_fill_gradient(low = "blue", high = "red") +
  guides(color = "none", fill = "none", alpha = "none")
```

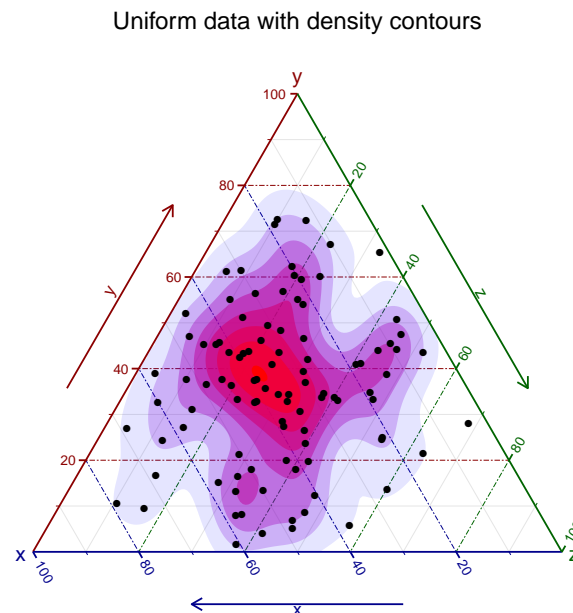


Figure 4.18: A trilinear plot with density contours^{fig:ggterm-demo}

{ex:lifeboat1}

EXAMPLE 4.19: Lifeboats on the *Titanic*

We examine the question of who survived and why in the sinking of the *RMS Titanic* in Section 5.4 (Example ??), where we analyze a four-way table, `Titanic`, of the 2201 people on board (1316 passengers and 885 crew), classified by Class, Sex, Age, and Survival. A related data set, `Lifeboats` in `vcd` tabulates the survivors according to the life boats 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).¹¹ The data frame `Lifeboats` in `vcd` contains the data listed on

¹¹The *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 1178 (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.

p. 38 of that report.¹²

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 4.18, has most of the points near the top, 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), "")
ggtern(data = Lifeboats,
        mapping = aes(x = women, y = men, z = crew,
                      colour=side, shape=side, label=id)) +
  theme_tern_rgbw() +
  theme(plot.margin=unit(c(0,0,0,0), "mm")) +
  geom_point(aes(size=2)) +
  labs(title = "Lifeboats on the Titanic") +
  labs(T="Women and children") +
  guides(size = "none") +
  geom_smooth(method="lm", size=1.5, aes(fill=side)) +
  geom_text(vjust=1, color="black")
```

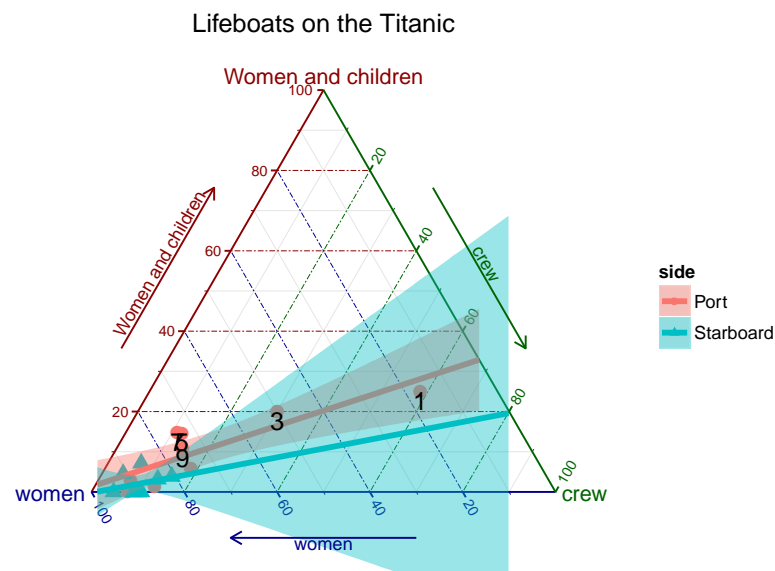


Figure 4.19: Lifeboats on the *Titanic*, showing the composition of each boat. Boats with more than 10% male passengers are labeled.

¹²The “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”.

The resulting plot in Figure 4.18, makes it immediately apparent that many of the boats launched from the port side differ substantially from the remaining 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 a linear regression smooth for the relation $\text{men} \sim \text{women}$.

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 4.18. The total number reported loaded is plotted against launch time in Figure 4.19, with a separate regression line and loess smooth fit to the data for the port and starboard sides.

```
ggplot(data = Lifeboats,
       aes(x=launch, y=total, colour=side, label=boat)) +
  geom_smooth(method="lm", aes(fill=side), size=1.5) +
  geom_smooth(method="loess", aes(fill=side), se=FALSE, size=1.2) +
  geom_point() + ylim(c(0,100)) +
  geom_text(vjust=-.5, color="black") +
  labs(y="Total loaded", x="Launch time")
```

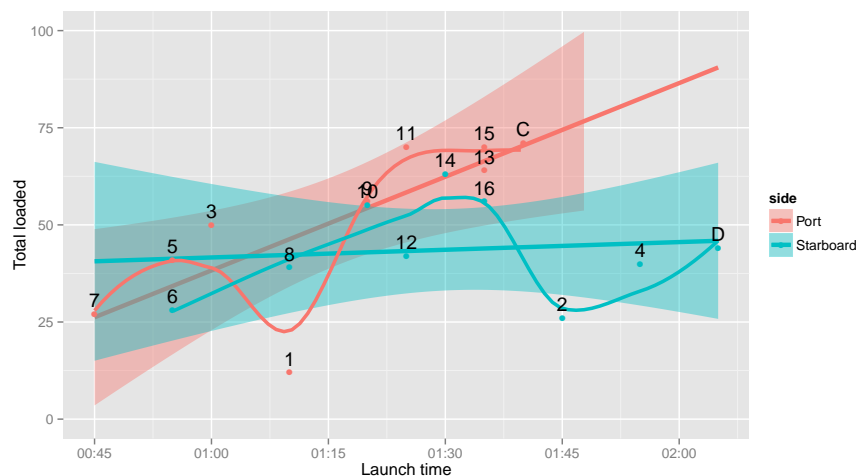


Figure 4.20: Number of people loaded on lifeboats on the Titanic vs. time of launch, by side of boat. The plot annotations show the linear regression and loess smooth.

From the linear regression lines in Figure 4.19, it seems that the rescue effort began in panic on the port side, with relatively small numbers loaded, and (from Figure 4.18), 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.

△

4.9 Chapter summary

{sec:twoway-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 asso-

ciations 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-square 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 Further reading

{sec:twoway-reading}

4.11 Lab exercises

{sec:twoway-lab}

1. The `Hospital` 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?
2. 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 `loglm()` method described in Section 4.3.1 to test whether the association between left and right eye acuity can be considered the same for men and women.
3. The graph in Figure 4.19 may be misleading, in that it doesn't take 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 4.19, showing the changes over time in this measure.

Chapter 5

Mosaic displays for n-way tables

{ch:mosaic}

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

{sec:mosaic-intro}

Little boxes, little boxes, little boxes made of ticky-tacky;
Little boxes, little boxes, little boxes all the same.
There are red ones, and blue ones, and green ones, and yellow ones;
Little boxes, little boxes, and they all look just the same.

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 (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 nonlinear 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 which are fitted by various loglinear models.

5.2 Two-way tables

{sec:mosaic

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.

{ex:haireye2a}

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.

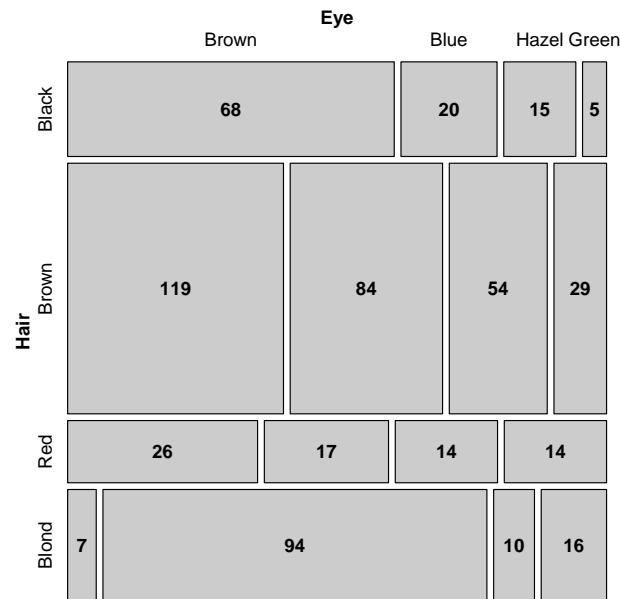


Figure 5.1: Basic mosaic display for hair color and eye color data. The area of each rectangle is proportional to the observed frequency in that cell.

fig:haireye-mosi

```
data(HairEyeColor, pkg="datasets")
haireye <- margin.table(HairEyeColor, 1:2)
mosaic(haireye)
```

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 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
## 0.1824 0.4831 0.1199 0.2145
```

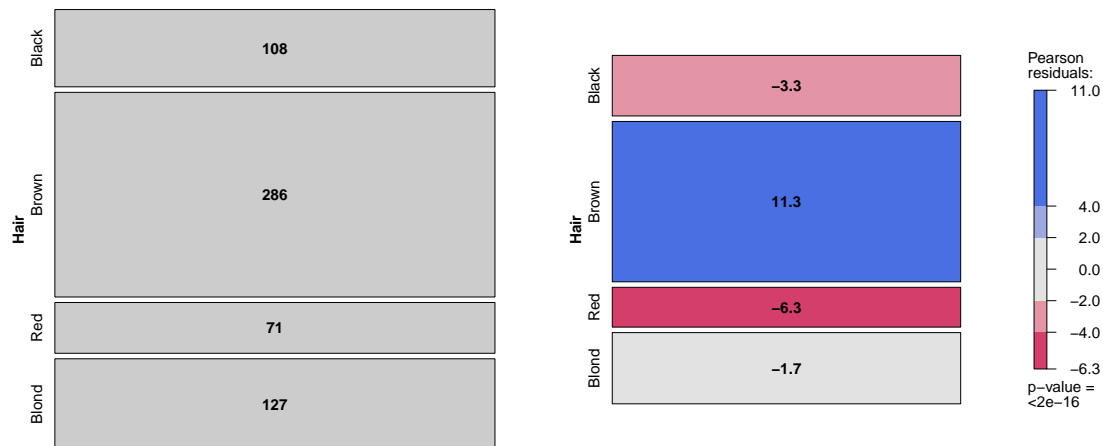


Figure 5.2: First step in constructing a mosaic display. Left: splitting the unit square according to frequencies of hair color; right: shading the tiles according to residuals from a model of equal marginal probabilities.

TODO: Resize these figures so they are both of the same height. Maybe have to manually do the figure environment here.

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, $d_i = (n_i - m_i)/\sqrt{m_i}$, are:

```
(residuals <- (hair - expected) / sqrt(expected))

## Hair
## Black Brown Red Blond
## -3.288 11.344 -6.329 -1.726
```

and these values are shown by color and shading as shown in the legend. 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 red heads 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.

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 heights of the tiles in the second mosaic display in Figure 5.3.

```
mosaic(haireye, shade=TRUE, suppress=0,
       labeling=labeling_residuals, gp_text=gpar(fontface=2))
```

- 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 `loglm()` to fit the independence model and `residuals()`.

```
HE.mod <- loglm(~ Hair + Eye, data=haireye)
round(resids <- residuals(HE.mod, type="pearson"), 2)

## Re-fitting to get frequencies and fitted values
##      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 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.

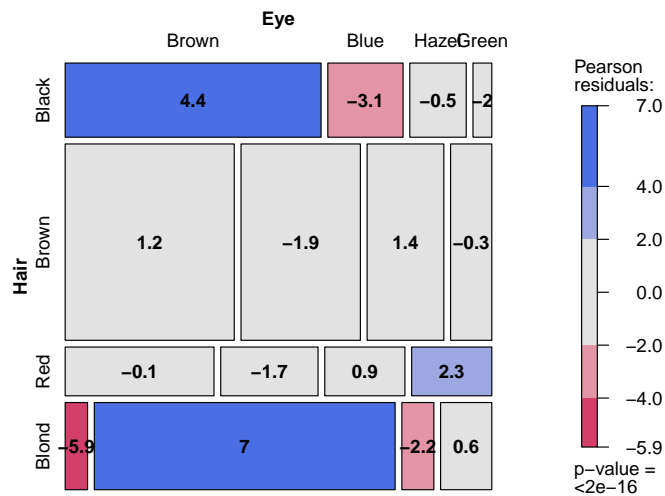


Figure 5.3: Second step in constructing the mosaic display. Each rectangle for hair color is subdivided in proportion to the relative frequencies of eye color, and the tiles are shaded in relation to residuals from the model of independence.

- 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] 138.3

(df <- prod(dim(haireye)-1))

## [1] 9

chisq.test(haireye)

##
## Pearson's Chi-squared test
##
## data:  haireye
## X-squared = 138.3, df = 9, p-value < 2.2e-16
```

△

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.

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 which 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 <- haireye[, c("Brown", "Hazel", "Green", "Blue")]
mosaic(haireye2, shade=TRUE)
```

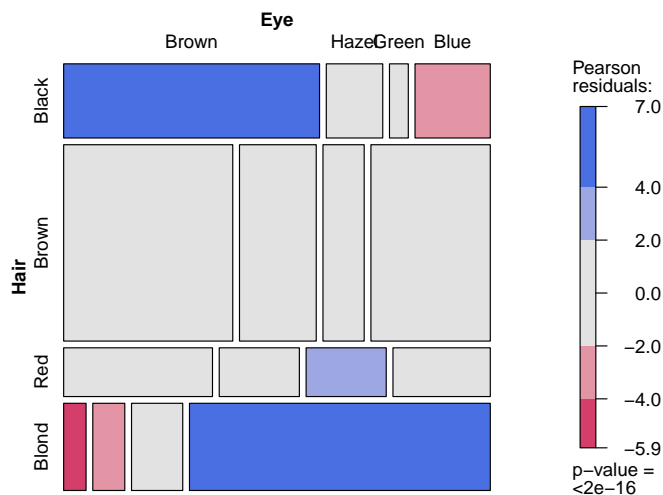


Figure 5.4: Two-way mosaic for Hair color and Eye color, reordered. The Eye colors were reordered from dark to light, enhancing the interpretation. Fig:haireye-mos9

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 hazel eyes do not quite fit this pattern

5.3 The strucplot framework

{sec:mosaic-strucplot}

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 `strutable()` 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

The strucplot framework is highly modularized: Figure 5.5 shows the hierarchical relationship between the various components. For the most part, you will use directly the convenience and related 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).

Group	Grapcon generator	Description
strucplot core	<code>struc_assoc()</code> <code>struc_mosaic()</code> <code>struc_sieve()</code>	core function for association plots core function for mosaic plots core function for sieve plots
labeling	<code>labeling_border()</code> <code>labeling_cboxed()</code> <code>labeling_cells()</code> <code>labeling_conditional()</code> <code>labeling_doubledecker()</code> <code>labeling_lboxed()</code> <code>labeling_left()</code> <code>labeling_left2()</code> <code>labeling_list()</code> <code>labeling_residuals()</code> <code>labeling_value()</code>	border labels centered labels with boxes, all labels clipped, and on top and left border cell labels border labels for conditioning variables and cell labels for conditioned variables draws labels for doubledecker plot left-aligned labels with boxes left-aligned border labels left-aligned border labels, all labels on top and left border draws a list of labels under the plot show residuals in cells show values (observed, expected) in cells
shading	<code>shading_binary()</code> <code>shading_Friendly()</code> <code>shading_hcl()</code> <code>shading_hsv()</code> <code>shading_max()</code> <code>shading_sieve()</code>	visualizes the sign of the residuals implements Friendly shading (based on HSV colors) shading based on HCL colors shading based on HSV colors shading visualizing the maximum test statistic (based on HCL colors) implements Friendly shading customized for sieve plots (based on HCL colors)
spacing	<code>spacing_conditional()</code> <code>spacing_dimequal()</code> <code>spacing_equal()</code> <code>spacing_highlighting()</code> <code>spacing_increase()</code>	increasing spacing for conditioning variables, equal spacing for conditioned variables equal spacing for each dimension equal spacing for all dimensions increasing spacing, last dimension set to zero increasing spacing
legend	<code>legend_fixed()</code> <code>legend_resbased()</code>	creates a fixed number of bins (similar to <code>mosaicplot()</code>) suitable for an arbitrary number of bins (also for continuous shadings)

Table 5.1: Available graphical appearance control (grapcon) generators in the strucplot frame-
work

{tab:grapcon}

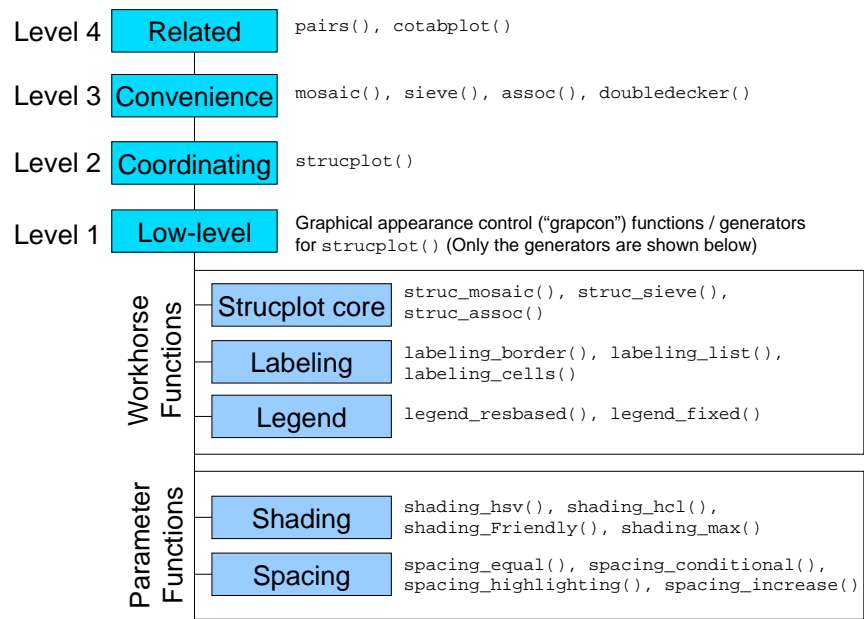


Figure 5.5: Components of the strucplot framework. High level functions use those at lower levels to provide a general system for tile-based plots of frequency tables.

{fig:struc}

- 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()`, 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 `pairs()` methods for table objects) arranging collections of plots of the `strucplot` framework into more complex displays (e.g., by means of panel functions).

5.3.1 Shading schemes

{sec:mosaic-shading}

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 and HCL colors, and the visualization of significance tests. **TODO:** This points to the need for a section, probably in Chapter 1, on color spaces and color schemes for categorical data graphics.

Specifying graphical parameters for `strucplot` displays

`Strucplot` displays in `vcd` are built using the `grid` graphics package. 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:

<code>col</code>	Color for lines and borders.
<code>fill</code>	Color for filling rectangles, polygons, ...
<code>alpha</code>	Alpha channel for transparency of fill color.
<code>lty</code>	Line type for lines and borders.
<code>lwd</code>	Line width for lines and borders.

In addition, a number of parameters control the display of text labels in these displays:

<code>fontsize</code>	The size of text (in points)
<code>cex</code>	Multiplier applied to <code>fontsize</code>
<code>fontfamily</code>	The font family
<code>fontface</code>	The font face (bold , <i>italic</i> , ...)

See `help(gpar)` for a complete list and 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.

```
# color by hair color
fill_colors <- c("brown4", "#acba72", "green", "lightblue")
(fill_colors <- 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, col=0))
```

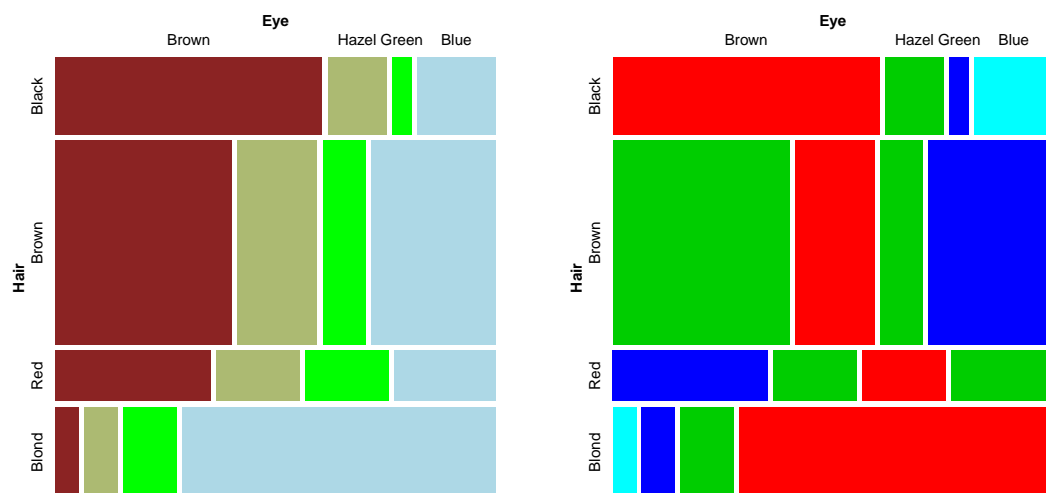


Figure 5.6: Mosaic displays for the `haireye2` data, using custom colors to fill the tiles. Left: Marimekko chart, using colors to reflect the eye colors; right: Toeplitz-based colors, reflecting the diagonal strips in a square table.

{fig:HE-fill}

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,¹ we might like to use color to highlight the pattern of diagonal 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 `palette()` function. The code below produces the right panel in Figure 5.6.

```
# toeplitz designs
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 <- palette()[1+toeplitz(1:4)]
mosaic(haireye2, gp=gpar(fill=fill_colors, col=0))
```

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 the association between the table variables.

¹For example, this would be appropriate for a square table, showing agreement between row and column categories, as in Section 4.7.

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`.

```
{ex:interp}
```

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))
```

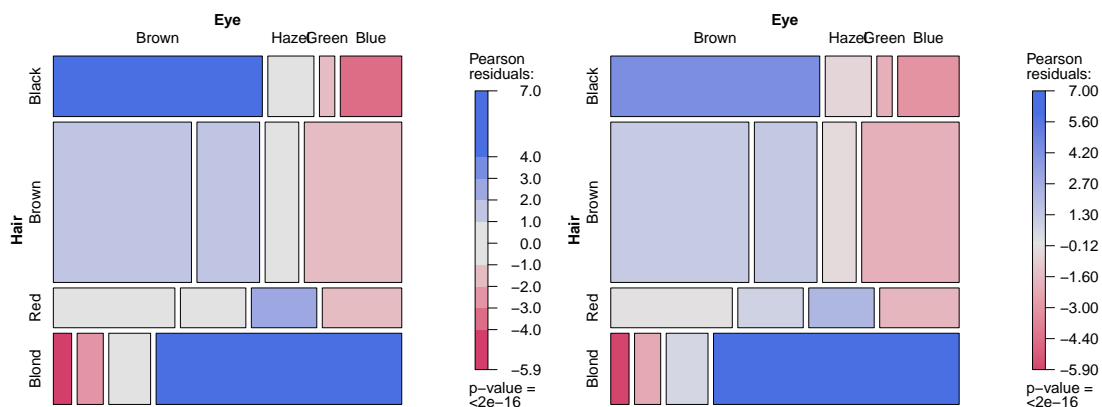


Figure 5.7: Interpolation options for shading levels in mosaic displays. Left: four shading levels; right: continuous shading.

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 ± 2 , ± 4 usually works quite well. \triangle

```
{ex:shading}
```

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 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 cut-offs 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 .² 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.³

```
mosaic(haireye2, gp=shading_Friendly, legend=legend_fixed)
set.seed(1234)
mosaic(haireye2, gp=shading_max)
```

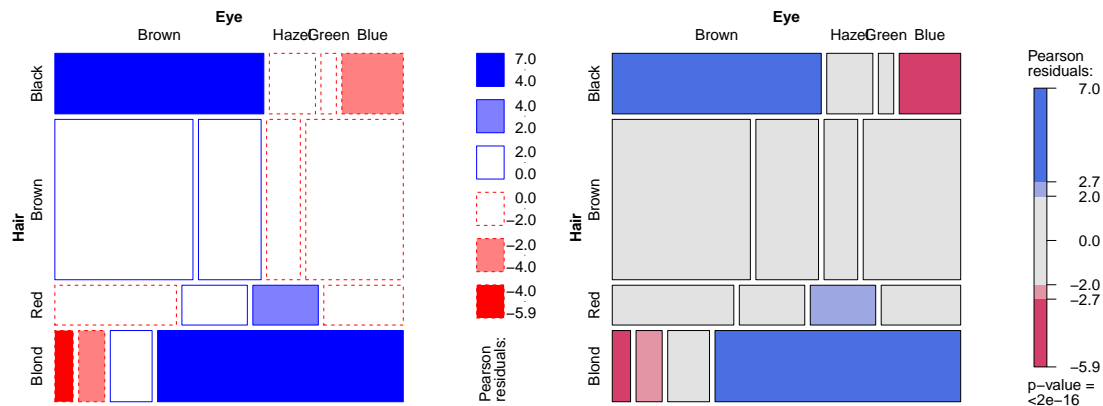


Figure 5.8: Shading functions for mosaic displays. Left: `shading_Friendly` using fixed cutoffs and the “Friendly” color scheme; right: `shading_max`, using a permutation-based test to determine significance of residuals.

²These 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 ($0.01 \leq p \leq 0.1$).

³This computation uses the `vcd` function `coinddep_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.

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. \triangle

```
{ex:arth-mosaic}
```

EXAMPLE 5.4: Arthritis treatment data

This example uses the Arthritis data, illustrated earlier (Example ??), on the relation between treatment and 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))
```

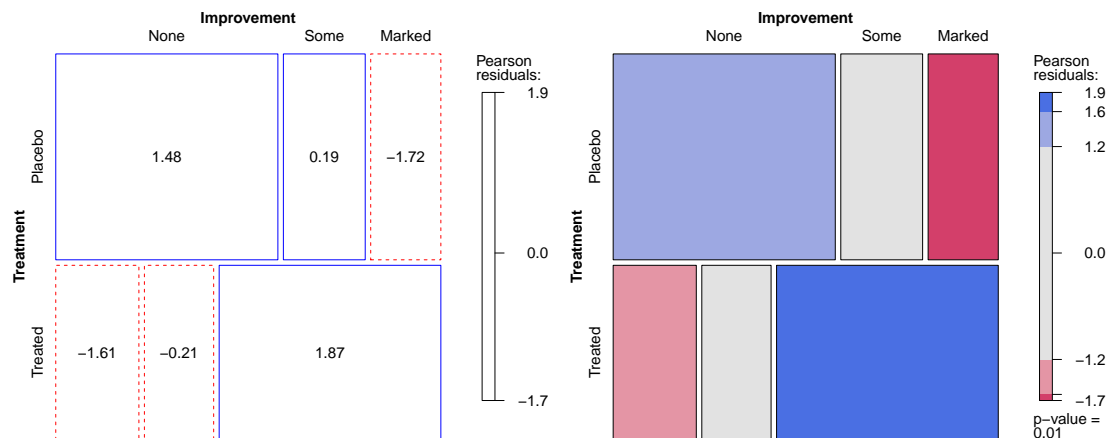


Figure 5.9: Mosaic plots for the female patients in the Arthritis data. Left: Fixed shading levels via `shading_Friendly`; right: shading levels determined by significant maximum residuals via `shading_max`. Fig:arth-mosaic

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(loglm(~Improvement + Treatment, data=art), type="pearson")

## Re-fitting to get frequencies and fitted values
##           Improvement
## Treatment    None    Some Marked
##   Placebo  1.478  0.1927 -1.717
##   Treated -1.609 -0.2097  1.870
```

The `shading_max()` function then calls `coinddep_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 <- coinddep_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%
## 1.239 1.917
```

△

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.

{ex:soccer2}

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
## cmeans   Col mean scores differ  5.63  4 0.229
## rmeans   Row 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.

```
set.seed(1234)
mosaic(UKSoccer, gp=shading_max, labeling=labeling_residuals, digits=2)
```

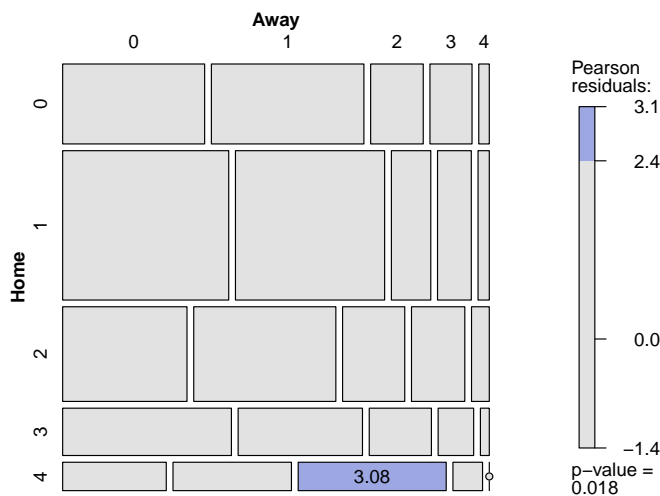


Figure 5.10: Mosaic display for UK soccer scores, highlighting one cell that stands out for further attention

This occurrence may or may not turn out to have some explanation, but at least the mosaic plot draws it to our attention. \triangle

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 multi-way 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.

{ex:HEC1}

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 5.4.1) there are different models for “independence” we could display. Here, we show residuals for the model of joint independence, $[\text{HairEye}][\text{Sex}]$, which asserts that the combinations of Hair color and Eye color are independent of Sex. This model, and the corresponding mosaic plot does *not* show the (overall) association between Hair color and Eye color we explored in earlier examples (see Figure 5.3). It merely shows how where the Hair color–Eye color combinations might differ by Sex.

In the call to `mosaic()` below, the model of joint independence is specified as the argument `expected = ~ Hair*Eye + Sex`. The `strucplot` labeling function `labeling_residuals` is used to display the residuals in the highlighted cells.

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

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—except among blue-eyed blonds, where there are an overabundance of females.

The model of joint independence has a non-significant Pearson $\chi^2(15) = 19.567, p = 0.189$. Yet, 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. \triangle

5.4.1 Fitting models

{sec:mosaic-fitting}

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.

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

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

$$\log m_{ijk} = \mu + \lambda_i^A + \lambda_j^B + \lambda_k^C + \lambda_{ij}^{AB} + \lambda_{ik}^{AC}, \quad (5.1) \quad \{\text{eq:AB-AC}\}$$

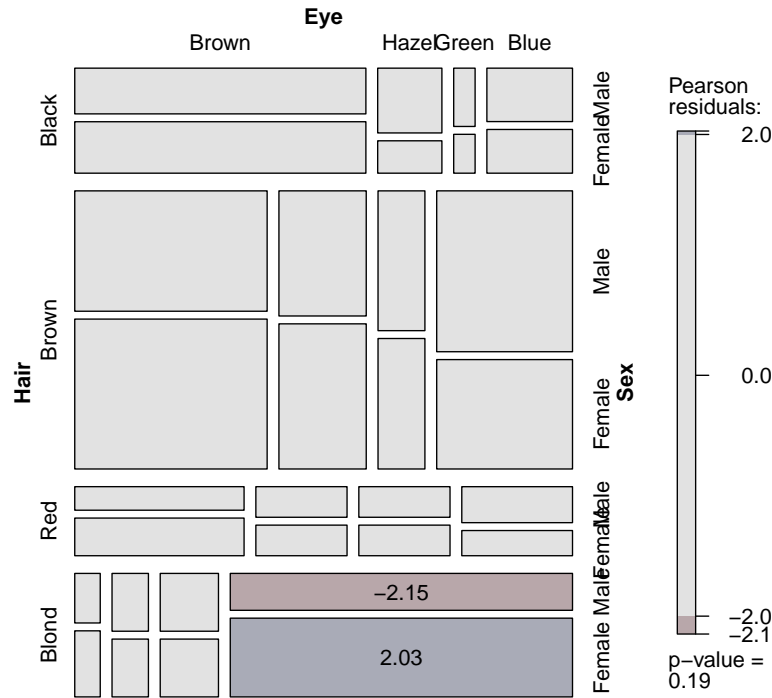


Figure 5.11: Three-way mosaic for Hair color, Eye color and Sex. Residuals from the model of joint independence, $[HE][S]$ are shown by shading. fig:HEC-mosi

(as described in Section ??) and reproduces the $\{AB\}$ and $\{AC\}$ marginal subtables.⁴ 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. (5.1) 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.⁵ Not included here is the **saturated model**, $[ABC]$, which fits the observed data exactly.

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

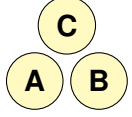
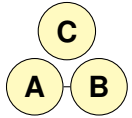
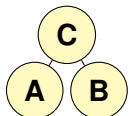
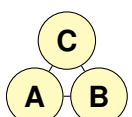
$$\pi_{ijk} = \pi_{i++} \pi_{+j+} \pi_{++k} ,$$

for all i, j, k in a three-way table. This corresponds to the log-linear model $[A][B][C]$.

⁴The notation here uses curly braces, $\{\bullet\}$ to indicate a marginal subtable summed over all other variables.

⁵For 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	

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 $\sim A*B + C$, 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 $\sim A*C + B*C$. This means that conditional probabilities, $\pi_{ij|k}$ obey

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

where $\pi_{ij|k} = \pi_{ijk} / \pi_{ij+}$, $\pi_{i+|k} = \pi_{i+k} / \pi_{++k}$, and $\pi_{+j|k} = \pi_{+jk} / \pi_{++k}$. 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.

TODO: Add a textbox or text describing the general scheme for translating among loglinear shorthand, R model formulas and independence interpretations.

{ex:HEC2}

EXAMPLE 5.7: Hair color, eye color and sex

We continue with the analysis of the `HairEyeColor` data from Example 5.6. Figure 5.11 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 5.12 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, $[Hair-Sex][EyeSex]$ testing whether, given sex, hair color and eye color are independent. 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")
```

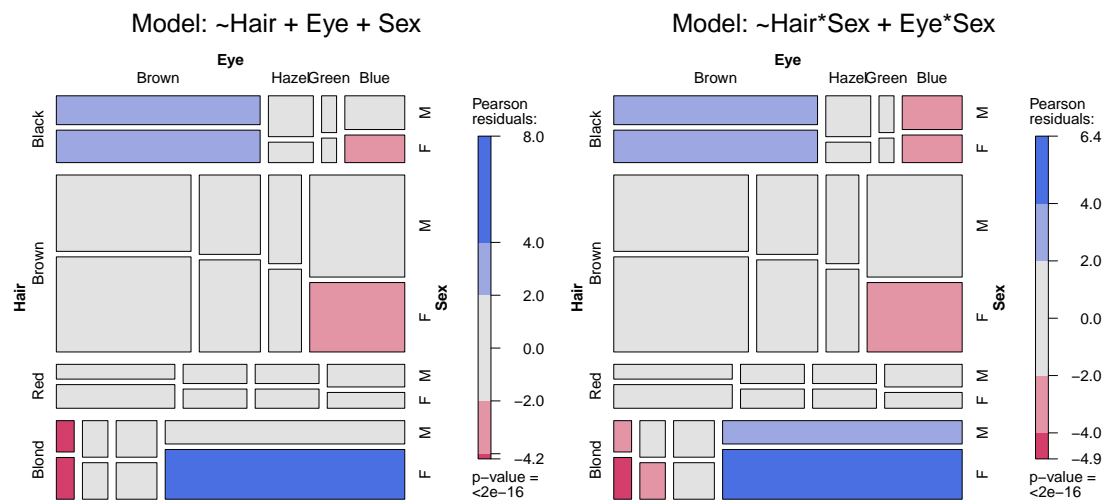


Figure 5.12: Mosaic displays for other models fit to the data on Hair Color, Eye color and Sex. Left: Mutual independence model; right: Conditional independence of Hair color and Eye color given Sex.

Compared with Figure 5.11 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 ??, 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 `summarise()` in **vcdExtra**.

```
library(MASS)
mod1 <- loglm(~ Hair + Eye + Sex, data=HEC) # mutual independence
mod2 <- loglm(~ Hair*Sex + Eye*Sex, data=HEC) # conditional independence
mod3 <- loglm(~ Hair*Eye + Sex, data=HEC) # joint independence
vcdExtra::summarise(loglmlist(Mutual=mod1, Condit=mod2, Joint=mod3))

## Re-fitting to get fitted values
## Re-fitting to get fitted values
## Re-fitting to get fitted values
## Model Summary:
##      LR Chisq Df Pr(>Chisq)    AIC    BIC
## Mutual   166.3 24      0.000 118.3  13.1
## Condit   156.7 18      0.000 120.7  41.8
## Joint    19.9 15      0.177 -10.1 -75.9
```

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 166.3 24      0
## Pearson          164.9 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    166.30 24
## Model 2    156.68 18      9.622        6      0.1415
## Model 3     19.86 15    136.821        3      0.0000
## Saturated     0.00  0     19.857       15      0.1775
```

5.4.2 Sequential plots and models

{sec:mosaic}

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,

$$\{eq:seqprod\} \quad p_{ijkl\dots} = \underbrace{p_i \times p_{j|i}}_{\{v_1 v_2 v_3\}} \times p_{k|ij} \times p_{l|ijk} \times \dots \times p_{n|ijk\dots} \quad (5.2)$$

In Eqn. (5.2), 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:

$$\{eq:seqgsq\} \quad G^2_{[v_1][v_2]\dots[v_n]} = G^2_{[v_1][v_2]} + G^2_{[v_1 v_2][v_3]} + G^2_{[v_1 v_2 v_3][v_4]} + \dots + G^2_{[v_1 \dots v_{n-1}][v_n]} \quad (5.3)$$

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 5.11) 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. (5.3) is shown as a visual equation in Figure 5.13. 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, \dots, E . In all cases, the natural model for the one-way margin is the equiprobability model, and that for the two-way margin is $[A][B]$.

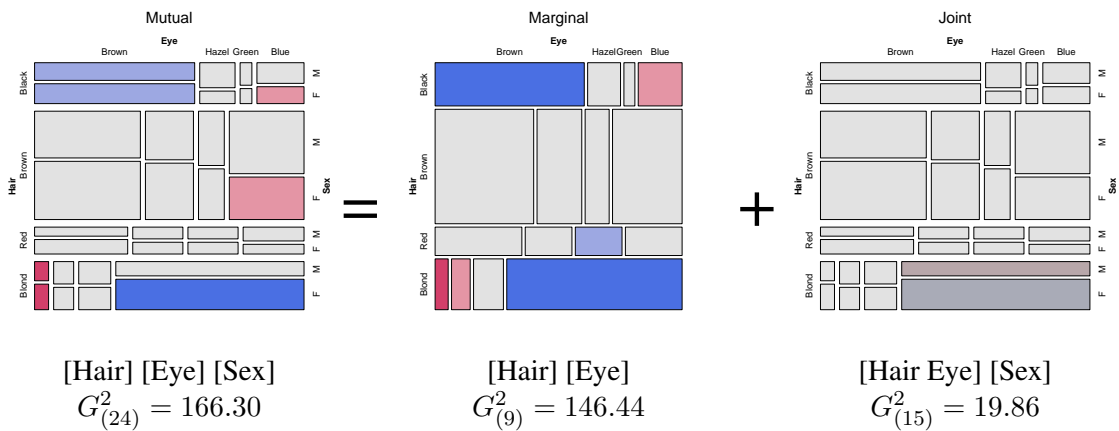


Figure 5.13: Visual representation of the decomposition of the G^2 for mutual independence (total) as the sum of marginal and joint independence.

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]

The `vcdExtra` package provides a collection of convenience functions that generate the log-linear 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.

```
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")
summarise(HEC.mods)

## Model Summary:
##      LR Chisq Df Pr(>Chisq)    AIC    BIC
## model.1    165.6  3    0.000  159.6  146.4
## model.2    146.4  9    0.000  128.4   89.0
## model.3     19.9 15    0.177  -10.1 -75.9
```

In this section we have described a variety of models which can be fit to higher-way tables, some relations among those models, and the aspects of lack-of-fit which are revealed in the mosaic displays. The following examples illustrate 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.⁶ 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.”

5.4.3 Causal models

{sec:causal}

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, §7.2)). Suppose, for example, that the causal ordering of four

⁶When one variable, R is a response, this normally is the model of joint independence, $[E_1 E_2 \dots] [R]$, where E_1, E_2, \dots are the explanatory variables. Better-fitting models will often include associations of the form $[E_i R]$, $[E_i E_j R] \dots$

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 which treat (a) B as a response to A , (b) C as a response to A and B jointly, (c) and D as a response to A , B and C . These are equivalent to the loglinear models which 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. (5.2). In application, residuals from each submodel show the associations that remain unexplained.

{ex:marital1}

EXAMPLE 5.8: 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 2⁴ table shown below, and included in `vcd` as `PreSex`.

```
data("PreSex", package="vcd")
strutable(Gender+PremaritalSex+ExtramaritalSex ~ MaritalStatus, PreSex)
```

	Gender	Women				Men			
	PremaritalSex	Yes	No	Yes	No	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 analysed by ?, §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 5.14.

```
# (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

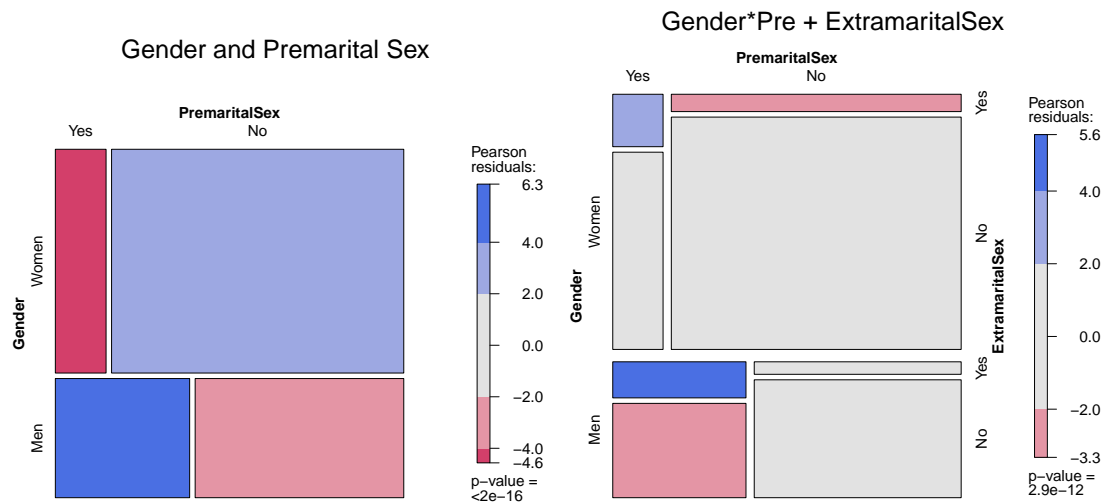


Figure 5.14: Mosaic displays for the first two marginal tables in the PreSex data. Left: Gender and premarital sex; right: fitting the model of joint independence with extramatrital sex, $[GP][E]$ ^{fig:presex2}

are equivalent to the recursive logit models whose path diagram is $G \rightarrow P \rightarrow E \rightarrow M$.⁷ 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 5.14 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 5.14 shows that extramarital sex depends on gender and premarital sex jointly. From the pattern of residuals in Figure 5.14 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.

```
oddsratio(margin.table(PreSex, 1:3), stratum=1, log=FALSE)

## Women    Men
## 3.562    3.605
```

⁷?, 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.

```
## (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")
```

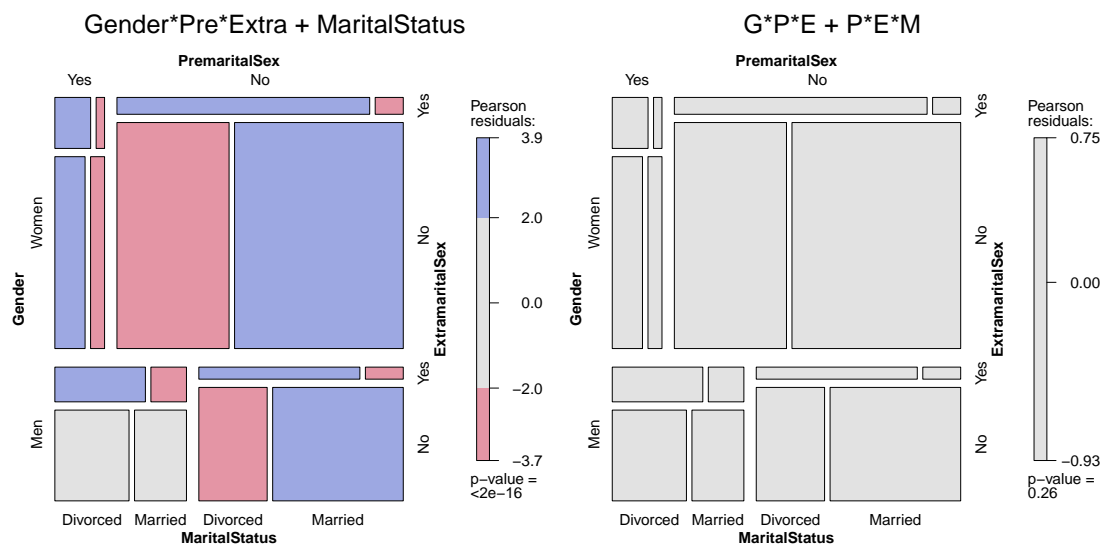


Figure 5.15: Four-way mosaics for the PreSex data. The left panel fits the model $[GPE][M]$. The pattern of residuals suggests other associations with marital status. The right panel fits the model $[GPE][PEM]$.

△

TODO: Complete this section with other examples: Titanic

5.4.4 Partial association

{sec:mospart}

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 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.⁸ 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.

⁸This 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.

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 experiences 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 the education, age categories.

A simpler version of this idea is considered first below (Example 5.9): 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 $A \perp B | C(k)$, $k = 1, \dots, K$. Then one can show (Andersen, 1991) that

$$\{eq:partial1\} \quad G^2_{A \perp B | C} = \sum_k^K G^2_{A \perp B | C(k)} \quad (5.4)$$

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

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

Mosaic and other displays in the *strucplot* framework for partial association can be produced in several different ways. One way is to use a model formula in the call to `mosaic()` which lists the conditioning variables after the “|” (given) symbol, as in `~ Memory + Attitude | Age + Education`. Another way is to use `cotabplot()`. This takes the same kind of conditioning model formula, but presents each panel for the conditioning variables in a separate frame within a trellis-like grid.⁹

`{ex:employ}`

EXAMPLE 5.9: Employment status data

Data from a 1974 Danish study of 1314 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”).

⁹Depending on your perspective, this has the advantage of adjusting for the total frequency in each conditional panel, or the disadvantage of ignoring these differences.

```
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 172.3 11      0
## Pearson          165.7 11      0
```

The residuals, shown in Figure 5.16, 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")
```

Beyond this baseline model, it is substantively more meaningful to consider the conditional independence model, $A \perp B | C$, (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 24.63 10 0.006093
## Pearson          26.07 10 0.003644
```

This model fits far better ($G^2(10) = 24.63$), but the lack of fit is still significant. The residuals, shown in Figure 5.17, 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.

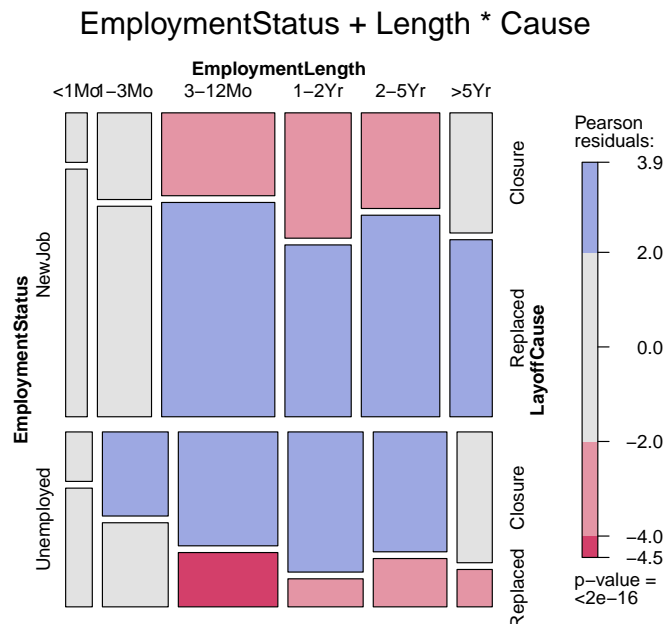


Figure 5.16: Mosaic display for the employment status data, fitting the baseline model of joint independence.

```
mosaic(Employment, shade=TRUE, gp_args=list(interpolate=1:4),
       expected = ~ EmploymentStatus*LayoffCause + EmploymentLength*LayoffCause,
       main = "EmploymentStatus * Cause + Length * Cause")
```

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 1.479  5  0.9155
## Pearson          1.483  5  0.9150
##
## $Replaced
## Call:
## loglm(formula = ~EmploymentStatus + EmploymentLength, data = x)
##
## Statistics:
##              X^2 df P(> X^2)
## Likelihood Ratio 23.15  5 0.0003158
## Pearson          24.59  5 0.0001673
```

EmploymentStatus * Cause + Length * Cause

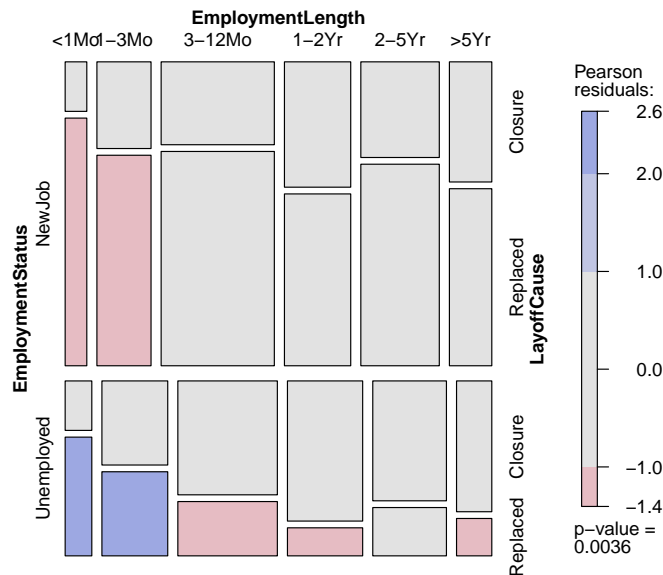


Figure 5.17: Mosaic display for the employment status data, fitting the model of conditional independence, $[AC][BC]$.

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 , (Eqn. (5.4)) and χ^2 .

Model	df	G^2	χ^2
$A \perp B C_1$	5	1.49	1.48
$A \perp B C_2$	5	23.15	24.59
$A \perp B C$	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 5.18.

```
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")
```

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. \triangle

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.

{ex:punish}

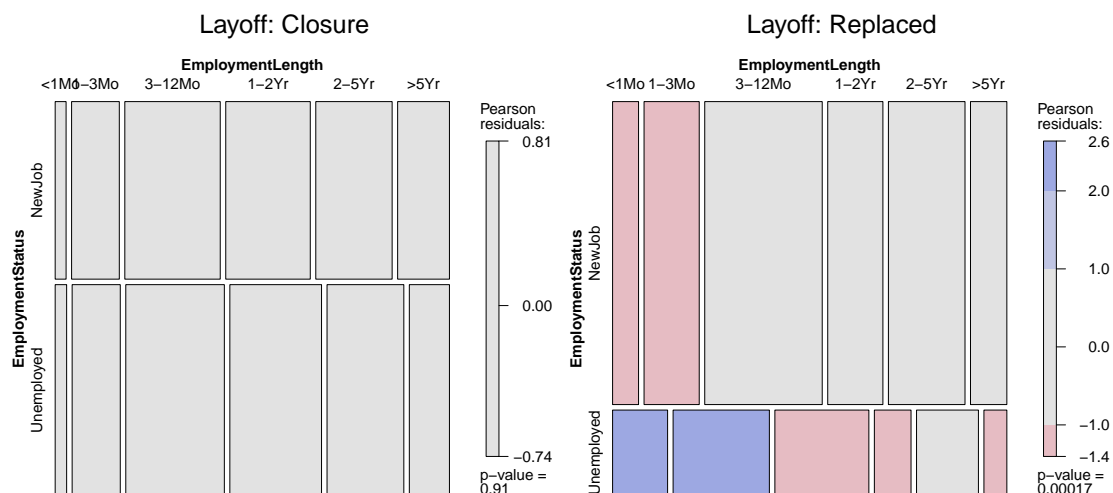


Figure 5.18: Mosaic displays for the employment status data, with separate panels for cause of layoff. `fig:employ-mos3`

EXAMPLE 5.10: 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); (d) age category of respondent.

```
data("Punishment", package = "vcd")
str(Punishment)

## 'data.frame': 36 obs. of 5 variables:
## $ Freq : num 1 3 20 2 8 4 2 6 1 26 ...
## $ attitude : Factor w/ 2 levels "no","moderate": 1 1 1 1 1 1 1 1 1 1 ...
## $ memory : Factor w/ 2 levels "yes","no": 1 1 1 1 1 1 1 1 1 2 ...
## $ education: Factor w/ 3 levels "elementary","secondary",...: 1 1 1 2 2 2 3 3 3 1 ...
## $ age : Factor w/ 3 levels "15-24","25-39",...: 1 2 3 1 2 3 1 2 3 1 ...
```

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

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

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

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

```
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 39.68   9 8.685e-06
## Pearson          34.60   9 6.996e-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 < 2.2e-16
```

These tests all show substantial association between attitude and memory of corporal punishment. How can we understand and explain this?

As in Example 5.9, 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 returns a 3×3 matrix, each of whose elements is the list of results returned by `loglm()`. The Pearson χ^2 statistics for each subtable can be extracted using `sapply()` as shown below.

```
mods.list <- apply(pun, c("Age", "Education"),
  function(x) loglm(~Memory + Attitude, data=x))
XSQ <- matrix(sapply(mods.list, function(x) x$pearson), 3, 3)
dimnames(XSQ) <- dimnames(mods.list)
addmargins(XSQ)

##           Education
## Age      Elementary Secondary   High   Sum
## 15-24         3.591    0.07878 0.09143  3.761
## 25-39         8.584    0.93467 0.48000  9.999
## 40+          11.626    6.09485 3.12371 20.844
## Sum          23.801    7.10830 3.69514 34.604
```

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 5.19. `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, $\sum_{k,\ell} \chi_{k,\ell}^2$.

```
set.seed(1071)
pun_cotab <- cotab_coinddep(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)
```

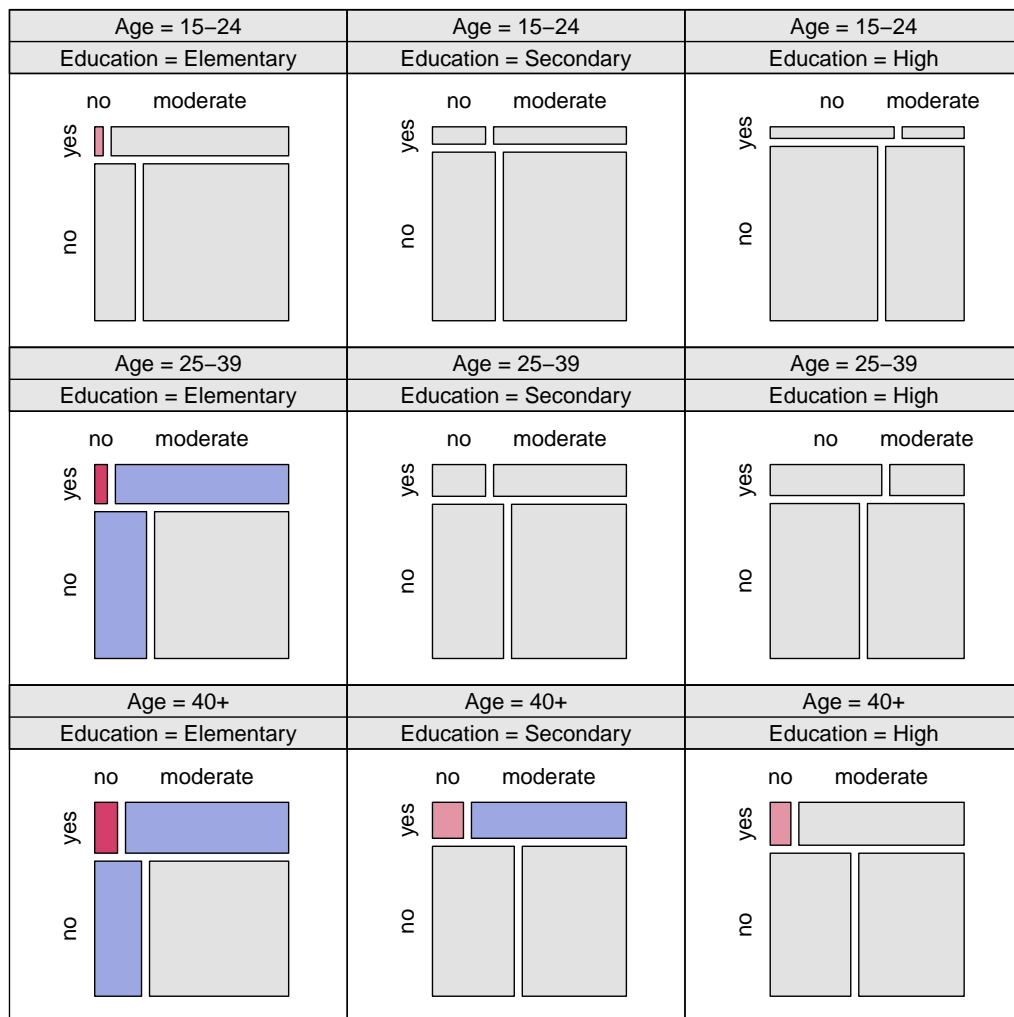


Figure 5.19: Conditional mosaic plot of the Punishment data for the model of conditional independence of attitude and memory, given age and education. Shading of tiles is based on the sum of squares statistic.

Alternatively, the pattern of conditional association can be shown somewhat more directly in a conditional mosaic plot (Figure 5.20), 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))
```

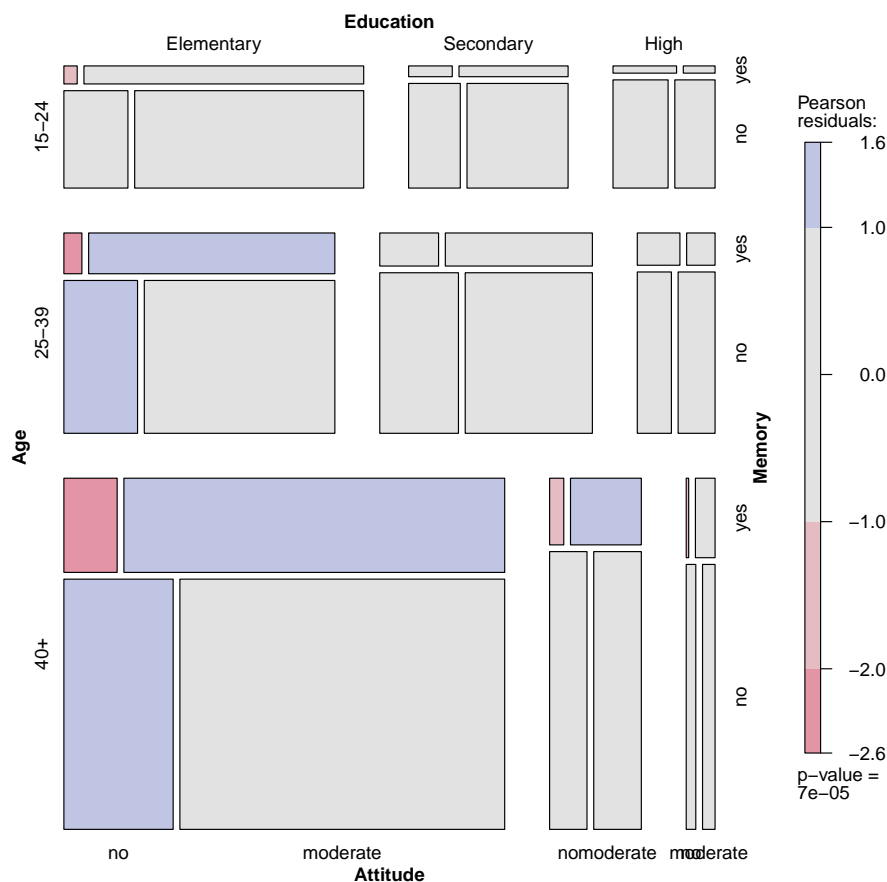


Figure 5.20: Conditional mosaic plot of the Punishment data for the model of conditional independence of attitude and memory, given age and education. This plot explicitly shows the total frequencies in the cells of age and education by the areas of the main blocks for these variables. Fig:punish-cond2

Both Figure 5.19 and Figure 5.20 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 5.19 the marginal frequencies of age and education are not shown, whereas in the `mosaic()` of Figure 5.20 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. △

5.5 Mosaic matrices for categorical data

{sec:mosmat}

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 set. 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 which 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, \dots , contains the interdependence among the collection of variables as a whole. The saturated loglinear model, $[ABC \dots]$ 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.

```
{ex:bartlett}
```

EXAMPLE 5.11: 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_Friendly)
```

The mosaic matrix for these data, showing all twoway marginal relations, is shown in Figure 5.21. 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 those cut long (vs. short), and the degree of association is stronger for planting time than for cutting length. △

```
{ex:marital2}
```

EXAMPLE 5.12: Marital status and pre- and extramarital sex

In Example 5.8 we examined a series of models relating marital status to reported premarital and extramarital sexual activity and gender in the `PreSex` data. Figure 5.22 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.

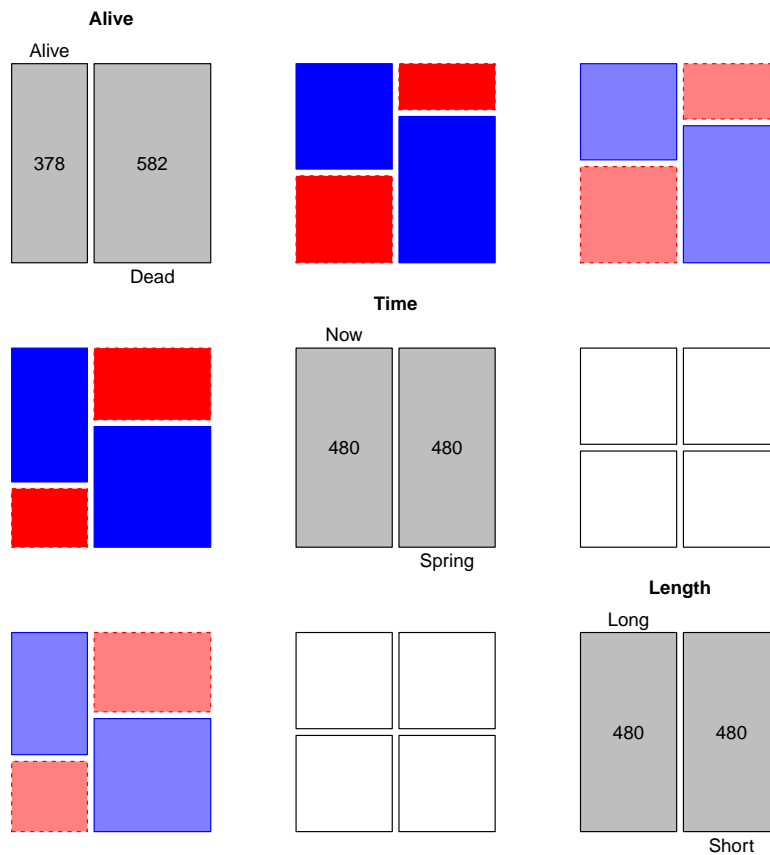


Figure 5.21: Mosaic pairs plot for the Bartlett data. Each panel shows the bivariate marginal relation between the row and column variables.

```
pairs(PreSex, shade=TRUE, gp_args=list(interpolate=1:4), space=0.25)
```

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)¹⁰ shows how marital status depends on each predictor marginally. The remaining panels show the relations within the set of explanatory variables.

Thus, we see in row 1, column 4, that marital status is independent of gender (all residuals equal zero, here), by design of the data collection. In the (1, 3) panel, we see that reported premarital sex is more often followed by divorce, while non-report is more prevalent among those still married. The (1, 2) panel shows a similar, but stronger relation between extramarital sex and marriage stability. These effects pertain to the associations of P and E with marital status (M)—the terms [PM] and [EM] in the loglinear model. We saw earlier that an interaction of P and E (the term [PEM]) is required to fully account for these data. This effect is not displayed in Figure 5.22.

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

¹⁰Rows 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.

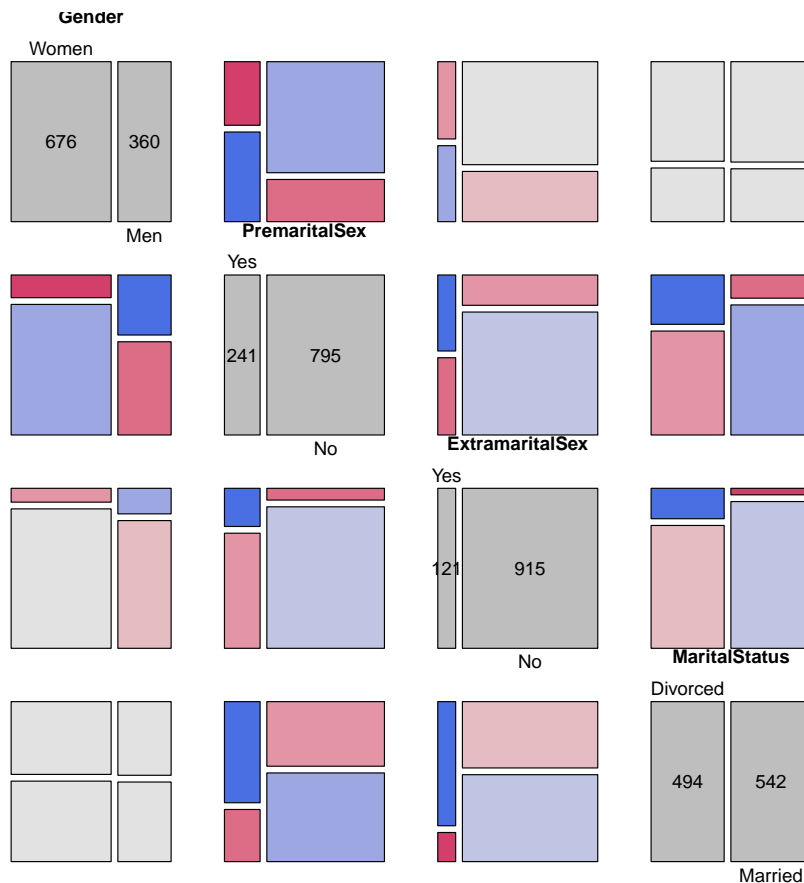


Figure 5.22: Mosaic pairs plot for the PreSex data. Each panel shows the bivariate marginal relation between the row and column variables. fig:marital-pairs

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.

△

```
{ex:berkeley4}
```

EXAMPLE 5.13: 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 4.10, Example 4.14) using fourfold displays (Figure 4.3 and Figure 4.4) and sieve diagrams (Figure 4.10). These displays showed either a marginal relation (e.g., Admit, Gender) or the full three-way table.

In contrast, Figure 5.23 shows all pairwise marginal relations among these variables, produced using `pairs()`:

```
pairs(UCBAdmissions, shade = TRUE, space=0.25)
```

The panel in row 2, column 1 shows that Admission and Gender are strongly associated

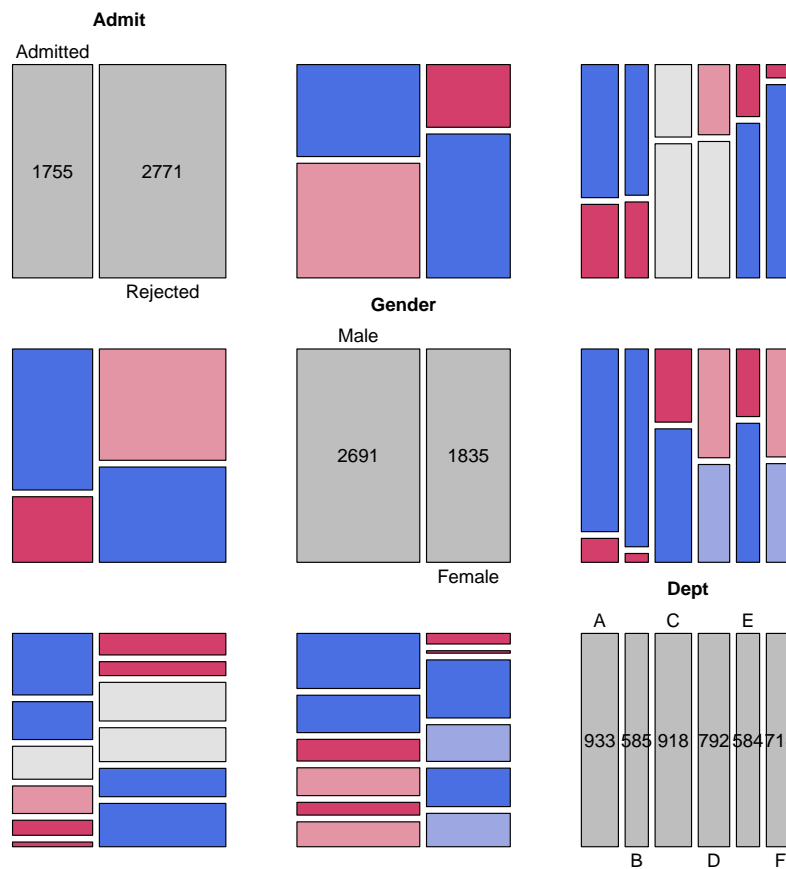


Figure 5.23: Mosaic matrix of the UCBA admissions data showing bivariate marginal relations^{fig:berk-pairs1}

marginally, as we saw in Figure 4.3, and overall, males are more often admitted. The diagonally-opposite panel (row 1, column 2) shows the same relation, splitting first by gender.¹¹

The panels in the third column (and third row) provide the explanation for the paradoxical result (see Figure 4.4) 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 ??). The (1,3) 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.

△

¹¹Note 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.

5.5.1 Generalized mosaic matrices and pairs plots

{sec:condma

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.

{ex:berkeley4b}

EXAMPLE 5.14: Berkeley admissions

Figure 5.24 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,
      lower_panel = pairs_mosaic(type = "joint", shade=TRUE),
      upper_panel = pairs_mosaic(type = "total", shade=TRUE),
      space=0.2)
```

TODO: Replace this with a figure using `type = "conditional"`, once we can get this to work.

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 5.25.

```
pairs(UCBAdmissions,
      lower_panel = pairs_mosaic(type = "conditional", shade=TRUE),
      upper_panel = pairs_mosaic(type = "conditional", shade=TRUE),
      space=0.2)
```

Thus, the shading in the (1,2) and (2,1) panels show 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. **TODO: This isn't quite right!**

△

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), 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:

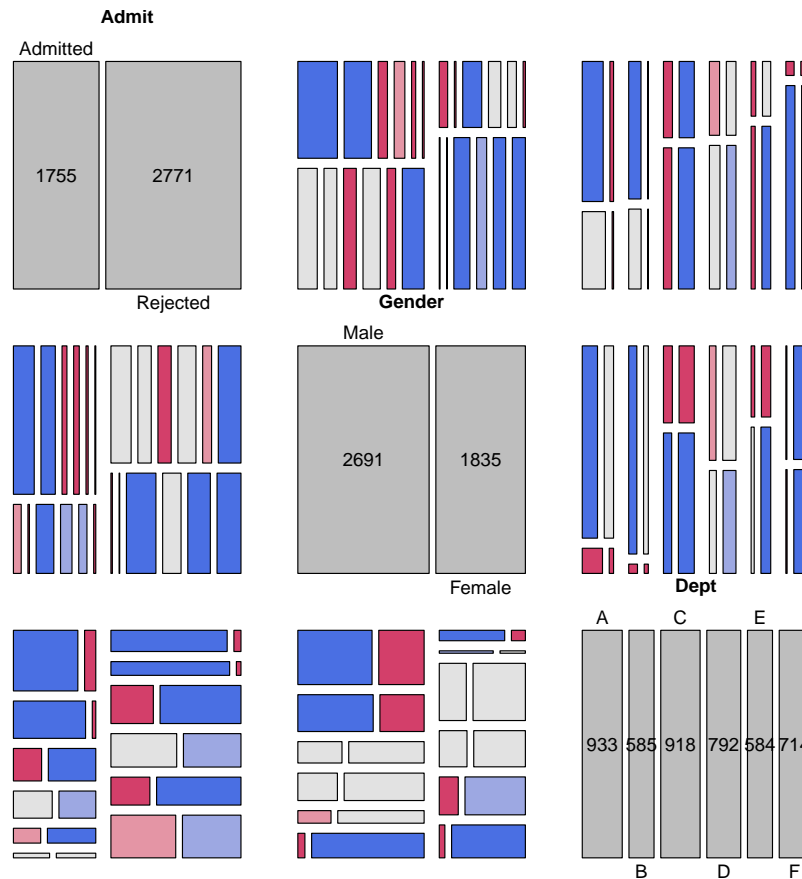


Figure 5.24: Generalized mosaic matrix of the UCBA admissions data. The above-diagonal plots fit models of joint independence; below-diagonal plots fit models of mutual independence. fig:berk-pairs2

CC: mosaic display, sieve diagram, doubledecker plot, faceted or divided bar chart;

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** package (an extension to **ggplot2**).

{ex:arthritis-gpairs}

EXAMPLE 5.15: Arthritis treatment

We illustrate these ideas with the *Arthritis* data using the **gpairs** package in Figure 5.26. 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.

```
library(gpairs)
data("Arthritis", package="vcd")
gpairs(Arthritis[,c(5,2,3,4)],
       diag.pars=list(fontsize = 16),
       mosaic.pars=list(shade=TRUE))
```

TODO: `shade=TRUE` doesn't have any effect here because the default `interpolate=` option can't be overridden. See if we can get Jay Emerson to fix this.

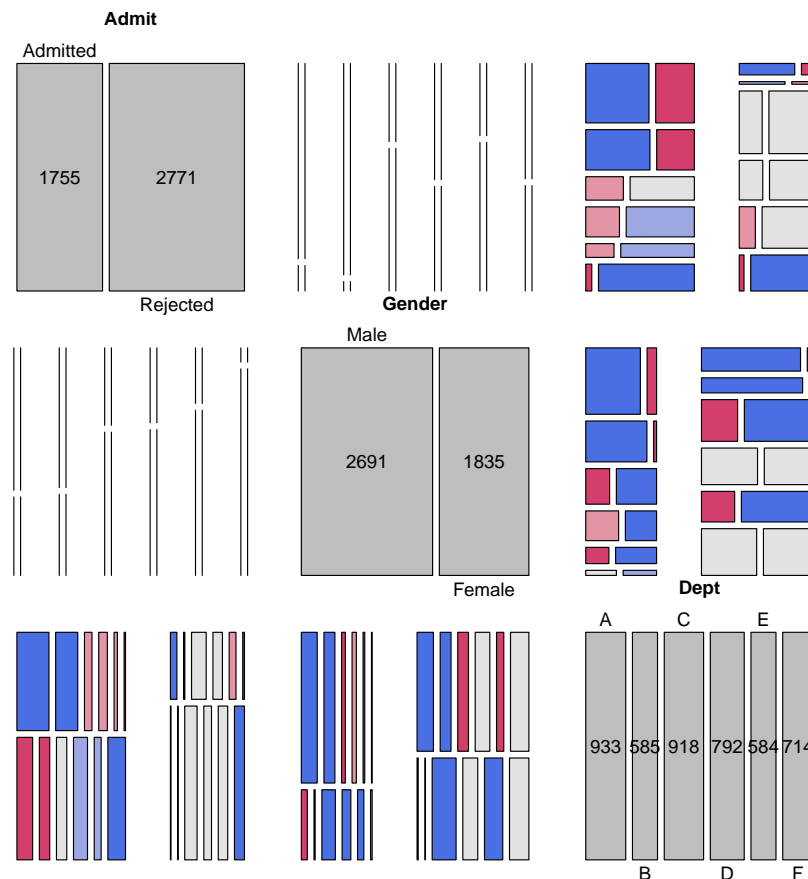


Figure 5.25: Generalized mosaic matrix of the UCBA admissions data. The above-diagonal plots fit models of joint independence; below-diagonal plots fit models of mutual independence. fig:berk-pairs3

`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 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. △

5.6 3D mosaics

{sec:3D}

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)

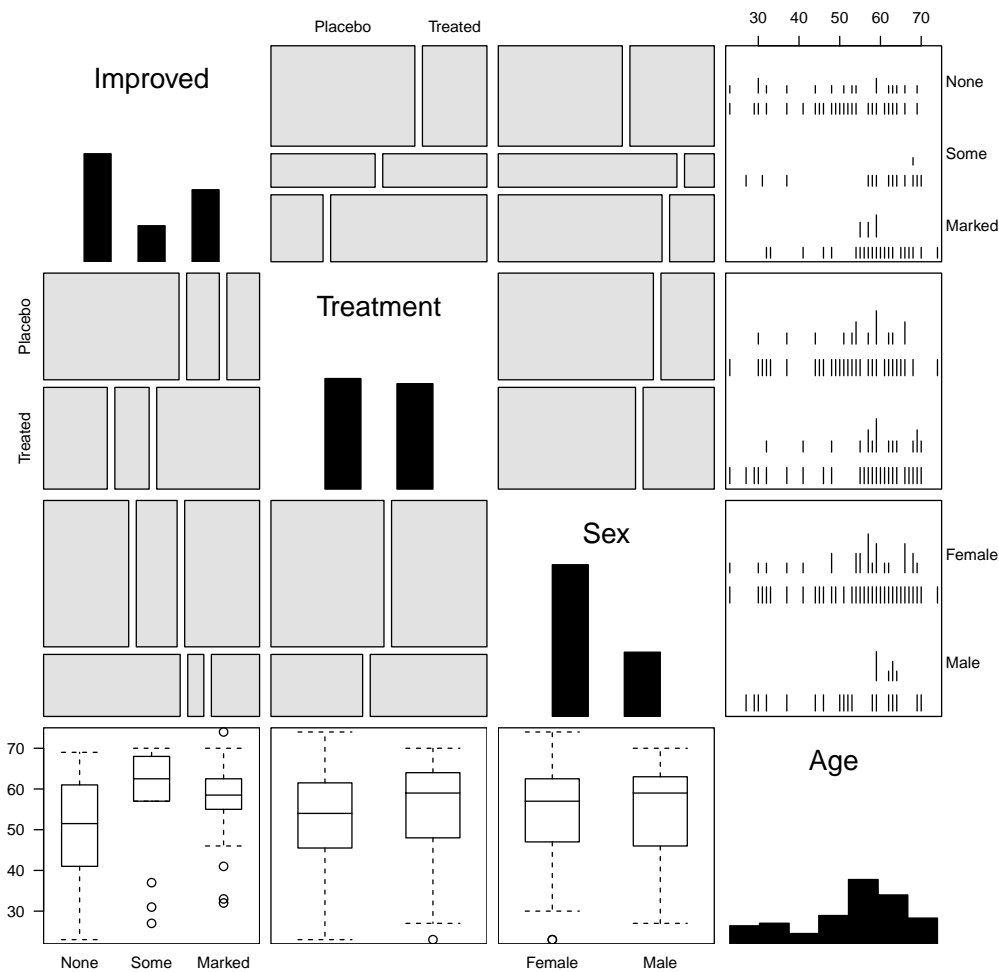


Figure 5.26: Generalized pairs plot of the Arthritis data. Combinations of categorical and quantitative variables can be rendered in various ways.

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` package, that mitigates 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.

{ex:bartlett-3d}

EXAMPLE 5.16: Bartlett data on plum root cuttings

In Example 5.11 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 5.27 shows the 3D version, produced using `mosaic3d()`:

```
mosaic3d(Bartlett)
```

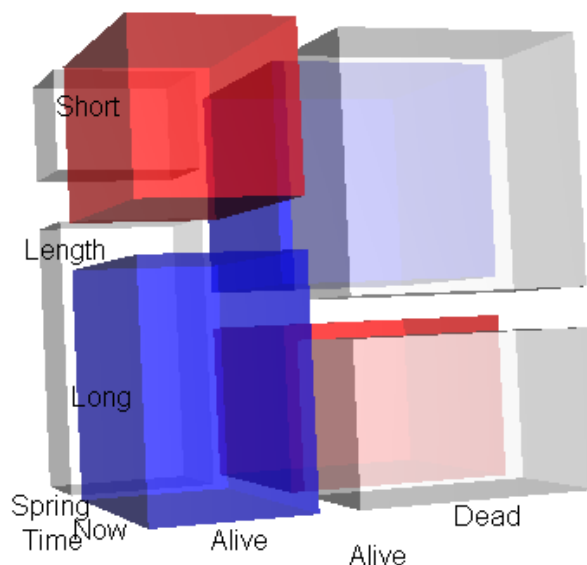


Figure 5.27: 3D mosaic plot of the Bartlett data, according to the model of mutual independence

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, though these relations can more easily be appreciated by rotating the 3D display. \triangle

5.7 Parallel coordinate plots for categorical data

Beyond pairwise plots, parallel coordinate plots (Inselberg, 1985, 1989, Wegman, 1990) provide another means to extend visualization methods beyond 2/3D. With the cartesian coordinate system, we run out of axes in 3D. Parallel coordinates overcome this limitation by plotting multiple axes in parallel. The geometry of parallel coordinates is the dual of cartesian geometry in standard plots: points in cartesian space appear as lines in parallel coordinates and vice versa. We first illustrate this visual framework for quantitative data, and then describe extensions to categorical data.

EXAMPLE 5.17: 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*.

Standard parallel coordinate plots are easily constructed in R using `parallelplot()` from the `lattice` package.

```
library(lattice)
data("iris", package="datasets")
vnames <- gsub("\\.", "\\n", names(iris))
key = list(
  columns = 3,
  lines = list(col=c("red", "blue", "green3"), lwd=4),
  col=c("red", "blue", "green3"),
  text = list(c("Setosa", "Versicolor", "Virginica")))
parallelplot(~iris[1:4], iris, groups = Species,
  varnames = vnames[1:4], key=key,
  horizontal.axis = FALSE, lwd=2,
  col=c("red", "blue", "green3"))
```

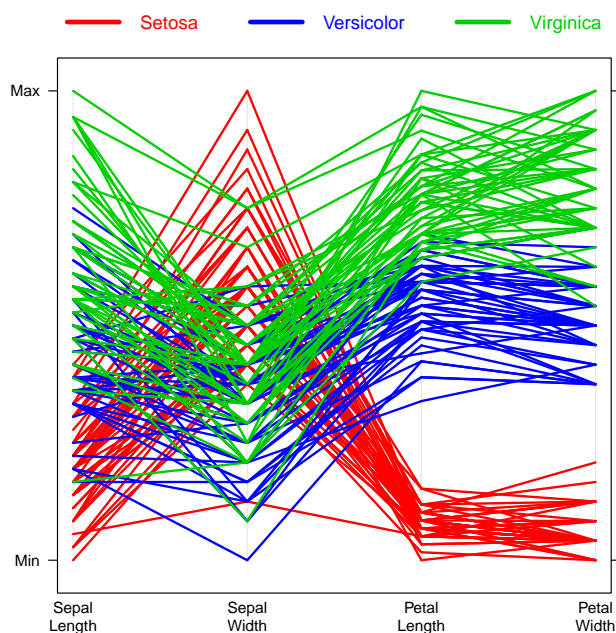


Figure 5.28: Parallel coordinates plot of the Iris data ^{fig:iris1}

Figure 5.28 shows the typical parallel coordinate plot for quantitative variables, where each variable axis represents the range of the corresponding variable, and observation values for the iris flowers are connected by lines. It can readily be seen that the flowers within each species vary systematically on the four variables, with the *setosa* flowers smaller on all except sepal width. Moreover, the patterns of the cases are positively correlated on all except sepal width, which is negatively related to the other three variables.

The transition to such plots for categorical data can be illustrated as shown in Figure 5.29. In the left panel, we have tried to show the *density* of a discrete variable *visually*, by (a) making the connecting lines thicker, but coloring them using transparent colors, so that more data “ink” corresponds to increasing frequency. (b) showing the categorical variable *Species* itself.

```
# alpha-blending, and showing species
parallelplot(~iris[1:5], data=iris, groups = Species,
  varnames = vnames, key = key,
  horizontal.axis = FALSE, lwd=8,
  col=c(rgb(1,0,0,.2), rgb(0,0,1,.2), rgb(0,205/255,0,.2)) )
```

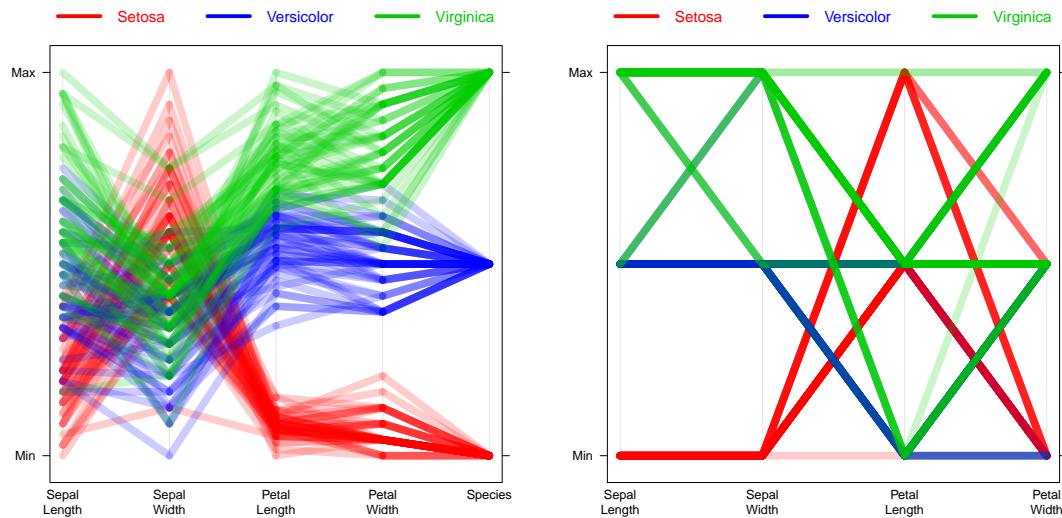


Figure 5.29: Discretized versions of parallel coordinate plots. Left: frequency shown by intensity of shading, along with a categorical variable; right: A less useful parallel coordinate plot for all categorical variables.

{fig:iris2}

)

Alternatively, we can `cut()` the quantitative variables into (ordered) categorical variables as shown in Figure 5.29 (right panel). However, although `parallelplot()` does handle discrete variables, the resulting plot is relatively uninformative.

```
# make ordered factors
iris2 <- within(iris, {
  sepalL <- cut(Sepal.Length, 3)
  sepalW <- cut(Sepal.Width, 3)
  petalL <- cut(Petal.Length, 3)
  petalW <- cut(Petal.Width, 3)
})
parallelplot(~iris2[6:9], data=iris2, groups = Species,
  varnames = vnames[1:4],
  horizontal.axis = FALSE, lwd=8, key=key,
  col=c(rgb(1,0,0,.2), rgb(0,0,1,.2), rgb(0,205/255,0,.2))
)
```

Note that, just as in the mosaic display, parallel coordinate plots are influenced by the order of the variable axes, because only adjacent pairs of variables are connected by lines. You can see this yourself by reordering the `iris` variables to place sepal width last (or first), as in this call. The result (another example of effect ordering for data displays) is not shown here, but is actually a more coherent display than Figure 5.28.

```
# effect of order of variables
parallelplot(~iris[c(1,3,4,2)], data=iris, groups = Species,
  varnames = vnames[c(1,3,4,2)], key=key,
  horizontal.axis = FALSE, lwd=8,
  col=c(rgb(1,0,0,.2), rgb(0,0,1,.2), rgb(0,205/255,0,.2))
)
```

TODO: Use this as an example of effect ordering in Chapter 1.



5.7.1 Parallel set plots: Hammock plots and common angle plots

The right panel of Figure 5.29 is unsuccessful in showing the relations among the four categorical iris measures because the individual observations are shown; their discrete nature results on much overplotting, obscuring the visual interpretation of frequency, and making unusual points (outliers) more dominant. A simple way to circumvent this is to use *parallel sets* (Kosara *et al.*, 2006) representation, that shows data *frequencies* instead of the individual data points. The method is based on the same axis layout of parallel coordinates, but with boxes representing the categories and parallelograms between the axes showing the relations between categories.

In implementations, the sizes of the boxes typically represent the frequencies of the categories, and the total length of each axis is subdivided according to their relative frequencies. We illustrate these methods below using the `ggparallel` package, which provides two other varieties: *hammock plots* (Schonlau, 2003) and *common angle plots* (Hofmann and Vendettuoli, 2013). These have better perceptual properties, as we describe below.

{ex:titanic-par1}

EXAMPLE 5.18: Titanic data

In this example, we use the `Titanic`, converted to a frequency data frame. In the call to `ggparallel()`, we use the `Freq` variable to weight the categories. `order=0` says to keep the order of the factor levels unchanged (rather than sorting them by frequency); `method="parset"` gives the basic parallel sets version.

```
library(RColorBrewer)
library(ggparallel)

## Error: there is no package called 'ggparallel'

titanic <- as.data.frame(Titanic)
vars <- names(titanic)[c(1, 4, 2)]
ggparallel(vars, titanic, order=0, weight="Freq", method="parset") +
  scale_fill_brewer(palette="Paired", guide="none") +
  scale_colour_brewer(palette="Paired", guide="none")

## Error: could not find function "ggparallel"
```

In this example, we positioned the variable `Survived` between `Class` and `Sex` to focus attention on the relation of each of these with survival. The bands between the first two axes show, for each class, the number who lived and the number who died. Yet, it is hard to accurately compare the relative frequencies of the various bands, because a perceptual illusion called the *line width illusion* makes the less slanted bands appear wider than more slanted bands that represent equal frequencies.¹² The main reason for this is that there is a strong visual bias toward evaluating the width of lines *orthogonal* to their slopes as opposed to vertically, which is the representation of frequency in this plot.



¹²You can see this for yourself with the following task: From Figure ??, write down the *order* of the classes according to the number who survived. Don't be too cocky: in one experimental study (Hofmann and Vendettuoli, 2013), only 6% of respondents could do so correctly. You can find the correct answer by running `sort(margin.table(Titanic[,,"Yes"], 1), dec=TRUE)`.

One solution to this problem is the *hammock plot* (Schonlau, 2003), which adjusts the width of the line by a factor ($\sin \theta$) to make the *perceived* orthogonal line proportional in width to the number of observations it represents. This works, but may overcorrect, because it assumes that everyone is governed 100% by the line width illusion.

In contrast, the *common angle plot* (Hofmann and Vendettuoli, 2013) tries to draw all lines in a plot with the same angle. To achieve this, instead of drawing straight lines between variables, it uses ribbons composed of connected line segments where at least one line segment is drawn with the same angle. In

{ex:titanic-par2}

EXAMPLE 5.19: Titanic data

With `ggparallel()`, hammock plots are obtained using `method="hammock"` and common angle plots using `method="angle"` (the default). Figure ?? shows a hammock plot designed to explore relations of Sex and Survived with Class, by plotting Class twice, on outside axes.

The labels for variables and factor levels are easier to read when the parallel axes are horizontal rather than vertical; in the `ggplot2` framework, this is done with `coord_flip()`.

```
# define colors for factor levels
cols <- c(brewer.pal(name="Blues", 6)[-c(1,2)],
          rev(brewer.pal(name="Reds", 3)[-1]),
          rev(brewer.pal(name="Greens", 3)[-1]))
# hammock plot
vars <- names(titanic)[c(1, 4, 2, 1)]
ggparallel(vars, data=titanic, weight="Freq", method="hammock",
           order=c(0,1,1,0), ratio=.25, text.angle=0) +
  scale_fill_manual(values=cols, guide="none") +
  scale_colour_manual(values=cols, guide="none") + coord_flip()

## Error: could not find function "ggparallel"
```

```
# angle plot
ggparallel(vars, data=titanic, weight="Freq",
           order=c(0,1,1,0), text.angle=0) +
  scale_fill_manual(values=cols, guide="none") +
  scale_colour_manual(values=cols, guide="none") + coord_flip()

## Error: could not find function "ggparallel"
```

△

5.8 Visualizing the structure of loglinear models

{sec:mosaic-struct}

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 non-linear 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 which 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(Sex+Age ~ Disease, struc)

##           Sex      F           M
##           Age Young Old Young Old
## Disease
## No           6    10     312   44
## Yes          37    31     192   76
```

First, note that there are substantial associations in this table, as shown in Figure 5.30, fitting the (default) mutual independence model.

```
mosaic(struc, shade=TRUE)
```

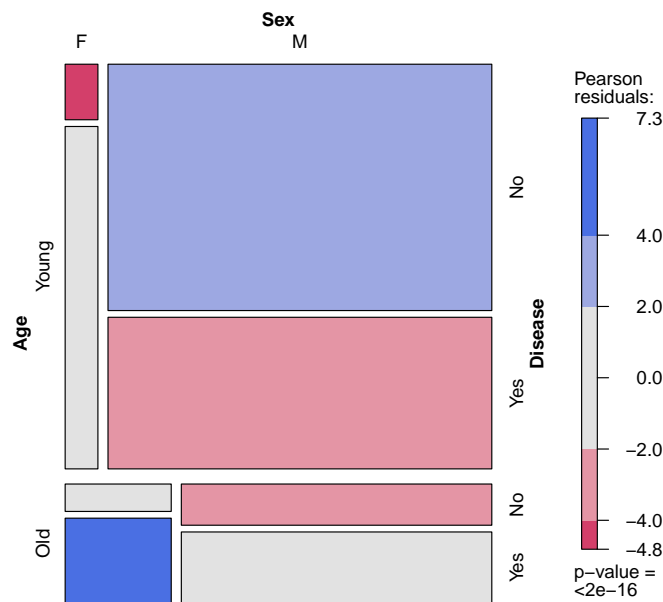


Figure 5.30: Mosaic display for the data on age, sex and disease. Observed frequencies are shown in the plot, and residuals reflect departure from the model of mutual independence. fig: struc-mos1

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 5.31, 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")
```

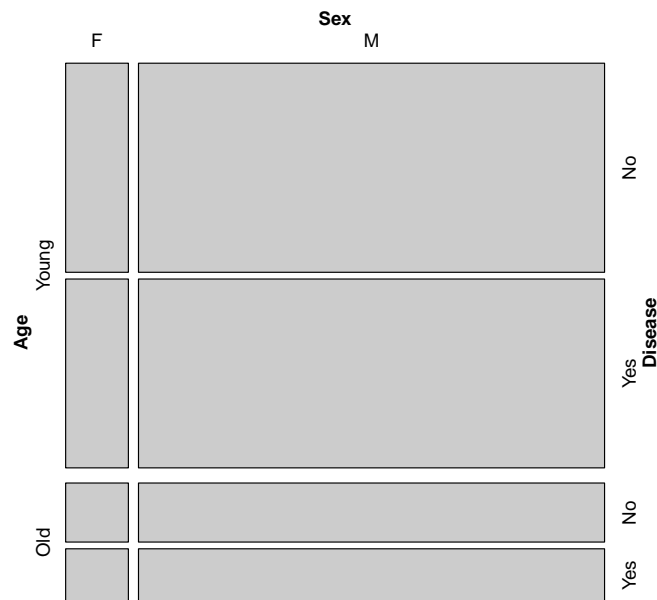


Figure 5.31: Mosaic display for the data on age, sex and disease, using expected frequencies under mutual independence. Left: `fig:struc-mos2`

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 which 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.

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

```
##           Sex      F      Old      M      Old
##           Age  Young      Old  Young      Old
```

```
## Disease
## No      34.099  10.036 253.308  74.557
## Yes     30.799   9.065 228.794  67.342
```

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_Friendly, type="total")
```

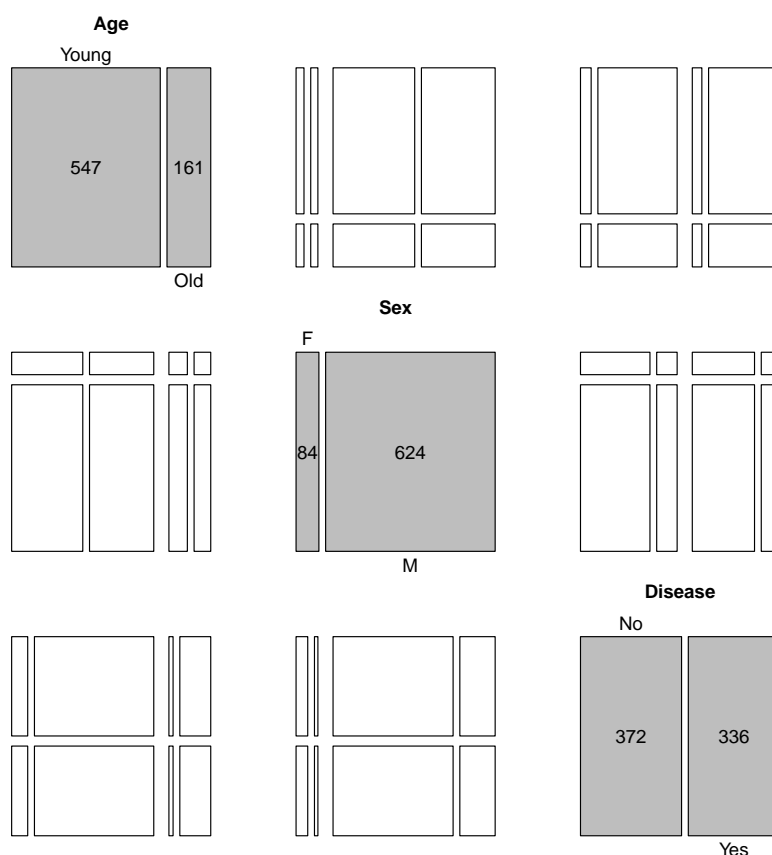


Figure 5.32: Mosaic matrix for fitted values under mutual independence. In all panels the joint frequencies conform to the one-way margins. fig:struc-mos3

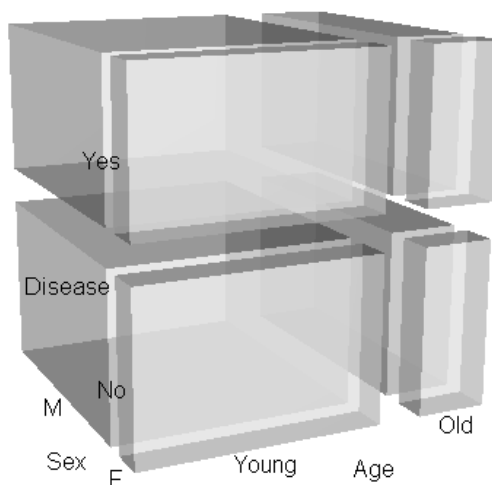
In this figure the same data are shown in all the off-diagonal panels and the mutual independence model was fitted in each case, but with the table variables permuted. All residuals are exactly zero in all cells, by construction. We see that in each view, the four large tiles, corresponding to the first two variables align, indicating that these two variable 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 ($\text{age} \perp \text{sex} \mid \text{disease}$).

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, $(X \perp Y | Z)$ 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 5.33, all of the 3D tiles are unshaded and you can see that the 3D unit cube has been sliced according to the marginal frequencies.

```
mosaic3d(fit)
```



```
{fig:struct-mos3d1}
```

Figure 5.33: 3D mosaic plot of frequencies according to the model of mutual independence

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(Sex+Age ~ Disease, fit)
```

##	Sex	F	M		
##	Age	Young	Old	Young	Old
##	Disease				
##	No	22.59	21.54	264.81	63.05
##	Yes	20.41	19.46	239.19	56.95

The `pairs.table()` plot, now using simpler pairwise plots (`type="pairwise"`), is shown in Figure 5.34.

```
pairs(fit, gp=shading_Friendly)
```

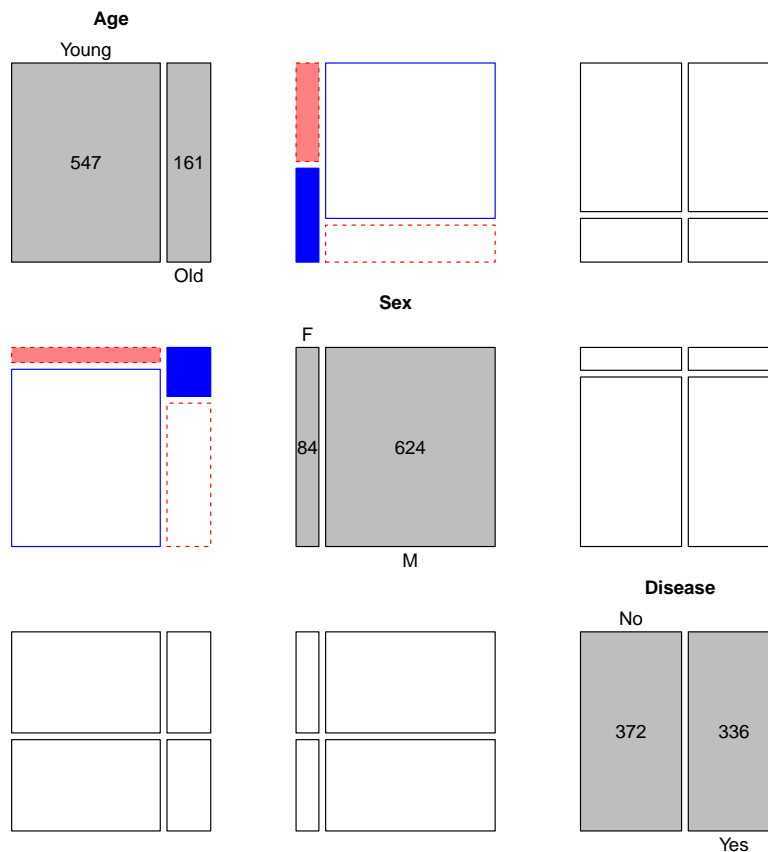


Figure 5.34: Mosaic matrix for fitted values under joint independence. ^{fig:struc-mos4}

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 Chapter summary

{sec:mosaic-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 which 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. Parallel set plots provide another method 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.

5.10 Further reading

{sec:mosaic-reading}

5.11 Lab exercises

{sec:mosaic-lab}

- 1.
2. For the `Bartlett` data described in Example 5.11, fit the model of no three-way association, H_4 in Table 5.2.
 - (a)
 - (b) Use a mosaic-like display to show the lack of fit for this model.
- 3.

Chapter 6

Correspondence analysis

{ch:corresp}

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.

6.1 Introduction

Whenever a large sample of chaotic elements are taken in hand and marshaled in the order of their magnitude, an unsuspected and most beautiful form of regularity proves to have been latent all along.

Sir Francis Galton (1822–1911)

Correspondence analysis (CA) is an exploratory technique which 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 components analysis*.

This technique finds scores for the row and column categories on a small number of dimensions which 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), Lebart *et al.* (1984) for a detailed treatment of the method and its applications from the French and Dutch 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 and the canonical correlation model, with close links to Correspondence analysis.

One simple development of CA is as follows: For a two-way table the scores for the row categories, namely $\mathbf{X} = \{x_{im}\}$, and column categories, $\mathbf{Y} = \{y_{jm}\}$, on dimension $m = 1, \dots, M$ 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

$$\frac{d_{ij}}{\sqrt{n}} = \frac{n_{ij} - m_{ij}}{\sqrt{n m_{ij}}} = \mathbf{X} \mathbf{D}_\lambda \mathbf{Y}^\top = \sum_{m=1}^M \lambda_m x_{im} y_{jm} , \quad (6.1)$$

where \mathbf{D}_λ is a diagonal matrix with elements $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_M$, 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

$$n \sum_{m=1}^d \lambda_m^2 / \chi^2 .$$

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 which 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:

- $\mathbf{N} = \{n_{ij}\}$ 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.
- $\mathbf{P} = \{p_{ij}\} = \mathbf{N}/n$ is the matrix of joint cell probabilities, called the **correspondence matrix**.

- $\mathbf{r} = \sum_j p_{ij} = \mathbf{P}\mathbf{1}$ is the row margin of \mathbf{P} ; $\mathbf{c} = \sum_i p_{ij} = \mathbf{P}^\top \mathbf{1}$ is the column margin. \mathbf{r} and \mathbf{c} are called the *row masses* and *column masses*.
- \mathbf{D}_r and \mathbf{D}_c are diagonal matrices with \mathbf{r} and \mathbf{c} on their diagonals, used as weights.
- $\mathbf{R} = \mathbf{D}_r^{-1} \mathbf{P} = \{n_{ij}/n_{+j}\}$ is the matrix of row conditional probabilities, called *row profiles*. Similarly, $\mathbf{C} = \mathbf{D}_c^{-1} \mathbf{P}^\top = \{n_{ij}/n_{i+}\}$ is the matrix of column conditional probabilities or *column profiles*.

Two types of coordinates, \mathbf{X} , \mathbf{Y} for the row and column categories are defined, based on the generalized singular value decomposition of \mathbf{P} ,

$$\mathbf{P} = \mathbf{A} \mathbf{D}_\lambda \mathbf{B}^\top$$

where \mathbf{D}_λ is the diagonal matrix of singular values $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_M$, \mathbf{A} is the $I \times M$ matrix of left singular vectors, normalized so that $\mathbf{A} \mathbf{D}_r^{-1} \mathbf{A}^\top = \mathbf{I}$, and \mathbf{B} is the $J \times M$ matrix of right singular vectors, normalized so that $\mathbf{B} \mathbf{D}_c^{-1} \mathbf{B}^\top = \mathbf{I}$. Thus the columns of \mathbf{A} and \mathbf{B} are orthogonal in the weighted metrics defined by the row and column margins, \mathbf{D}_r^{-1} and \mathbf{D}_c^{-1} , respectively.

principal coordinates: The coordinates of the row (\mathbf{F}) and column (\mathbf{G}) profiles with respect to their own principal axes are defined so that the inertia along each axis is the corresponding singular value, λ_i ,

$$\mathbf{F} = \mathbf{D}_r^{-1} \mathbf{A} \mathbf{D}_\lambda \quad \text{so that} \quad \mathbf{F}^\top \mathbf{D}_r \mathbf{F} = \mathbf{D}_\lambda \quad (6.2) \quad \{\text{eq:pcoord1}\}$$

$$\mathbf{G} = \mathbf{D}_c^{-1} \mathbf{B} \mathbf{D}_\lambda \quad \text{so that} \quad \mathbf{G}^\top \mathbf{D}_c \mathbf{G} = \mathbf{D}_\lambda \quad (6.3) \quad \{\text{eq:pcoord2}\}$$

The plot in principal coordinates, \mathbf{F} and \mathbf{G} is called the *symmetric map*.

standard coordinates: The standard coordinates (Φ, Γ) are a rescaling of the principal coordinates to unit inertia along each axis,

$$\Phi = \mathbf{D}_r^{-1} \mathbf{A} \quad \text{so that} \quad \Phi^\top \mathbf{D}_r \Phi = \mathbf{I} \quad (6.4) \quad \{\text{eq:scoord1}\}$$

$$\Gamma = \mathbf{D}_c^{-1} \mathbf{B} \quad \text{so that} \quad \Gamma^\top \mathbf{D}_c \Gamma = \mathbf{I} \quad (6.5) \quad \{\text{eq:scoord2}\}$$

These differ from the principal coordinates in Eqn. (6.2) and Eqn. (6.3) simply by the absence of the scaling factors, \mathbf{D}_λ .

Thus, the weighted average of the squared principal coordinates for the rows or columns on a principal axis equals the squared singular value, λ 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

{sec:ca-properties}

We summarize here some geometric and statistical properties of the Correspondence analysis solutions which 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: 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 $D_r^{-1}P$, rescaled inversely by the square-root of the column masses, $D_c^{-1/2}$. Distances between two row profiles, R_i and $R_{i'}$ is most sensibly defined as χ^2 distances, where the squared difference $[R_{ij} - R_{i'j}]^2$ is inversely weighted by the column frequency, to account for the different relative frequency of the column categories. The rescaling by $D_c^{-1/2}$ 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 near a column point when $n_{ij} - m_{ij} > 0$, and away 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 widely 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

{sec:ca-R}

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 the symmetric factorization. There is also a `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, and also provides `mjca()` for multiple and joint correspondence analysis of higher-way tables. `mjca()` (multiple and joint correspondence analysis)

FactoMineR: `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::ca` handles two-way tables and matrices, and require other formats to be converted to these forms. In the following, we largely use the `ca` package.

{ex:haireye3}

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 2-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 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 -1.1043   1.4409  -1.0889
## Brown -0.3245  -0.2191   0.9574
## Red   -0.2835  -2.1440  -1.6312
## Blond  1.8282   0.4667  -0.3181

(Gamma <- haireye.ca$colcoord)

##          Dim1      Dim2      Dim3
## Brown -1.0771   0.5924  -0.42396
## Blue   1.1981   0.5564   0.09239
## Hazel  -0.4653  -1.1228   1.97192
## Green   0.3540  -2.2741  -1.71844

# 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.

```
op <- par(cex=1.4, mar=c(5,4,2,1)+.1)
res <- plot(haireye.ca, xlab="Dimension 1 (89.4%)", ylab="Dimension 2 (9.5%)")
par(op)
```

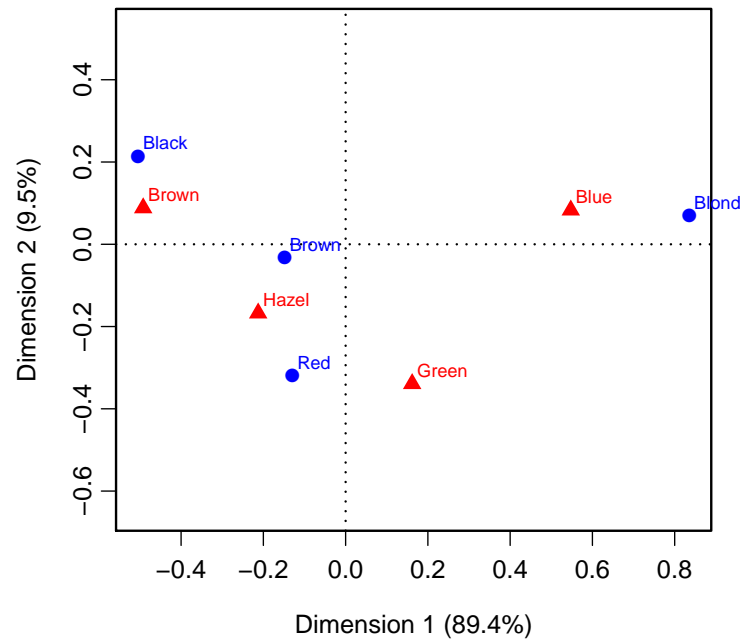


Figure 6.1: Correspondence analysis solution for the Hair color and Eye color data^{fig:ca-haireye-plot}

For use in further customizing such plots (as we will see in the next example), the function `plot.ca()` returns (invisibly)¹ the coordinates for the row and column points actually plotted, which we saved above as `res`:

```
res

## $rows
##      Dim1      Dim2
## Black -0.5046  0.21482
## Brown -0.1483 -0.03267
## Red   -0.1295 -0.31964
## Blond  0.8353  0.06958
##
## $cols
##      Dim1      Dim2
## Brown -0.4922  0.08832
## Blue   0.5474  0.08295
## Hazel  -0.2126 -0.16739
## Green  0.1618 -0.33904
```

It is important to understand that in CA plots (and related biplots, Section 6.7), the interpretation of distances between points (and angles between vectors) is meaningful. In order to achieve

¹This uses features incorporated in the `ca` package, version 0.54+.

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* = 1.²

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 column points are meaningful. Along Dimension 1, the eye colors could be considered roughly equally spaced, but for the hair colors, Blond is quite different in terms of its frequency profile.

△

{ex:mental3}

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. **TODO: Want a sieve diagram or mosaic plot in Chapter 5 for comparison here.** As in Example 4.6, 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 ?? . 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.

²In base R graphics, this is achieved with the `plot()` option `asp=1`.

```

op <- par(cex=1.3, mar=c(5,4,1,1)+.1)
res <- plot(mental.ca, ylim=c(-.2, .2),
            xlab="Dimension 1 (93.9%)", ylab="Dimension 2 (0.5%)")
lines(res$rows, col="blue", lty=3)
lines(res$cols, col="red", lty=4)
par(op)

```

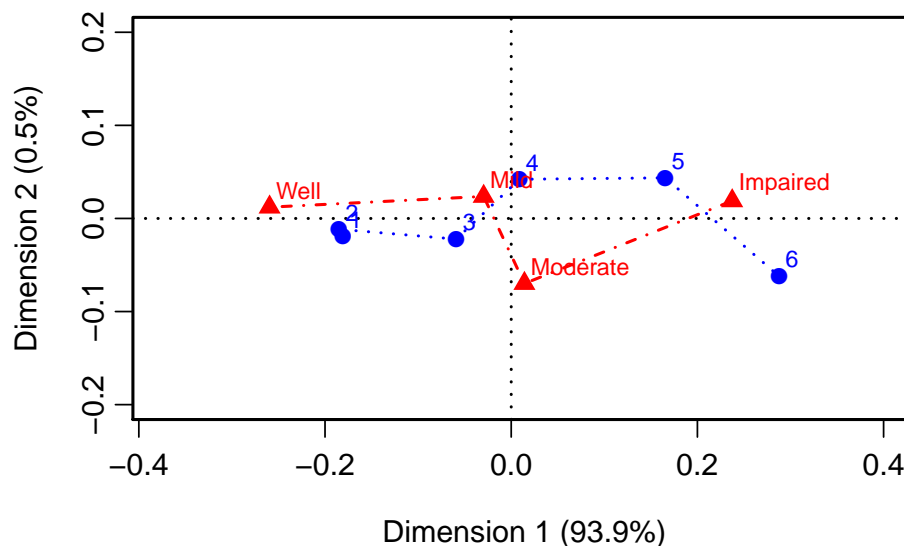


Figure 6.2: Correspondence analysis solution for the Mental health data fig:ca-mental-plot

The plot of the CA scores in Figure ?? shows that diagnostic mental health categories are well-aligned with Dimension 1. The and 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; perhaps 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.

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 ??.

△

{ex:victims2}

EXAMPLE 6.3: Repeat victimization

Example ?? presented mosaic displays for the data on repeat victimization, RepVict.

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 = 11131, df = 49, p-value < 2.2e-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 6.2. The addition of a `legend()` makes the plot more easily readable.

```
op <- par(cex=1.3, mar=c(4,4,1,1)+.1)
res <- plot(victim.ca, labels=c(2,0))
segments(res$rows[,1], res$rows[,2], res$cols[,1], res$cols[,2])
legend(-0.35, 0.45, c("First", "Second"), title="Occurrence",
       col=c("blue", "red"), pch=16:17, bg="gray90")
par(op)

```

In Figure 6.2 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 Pick Pocket as exceptions.

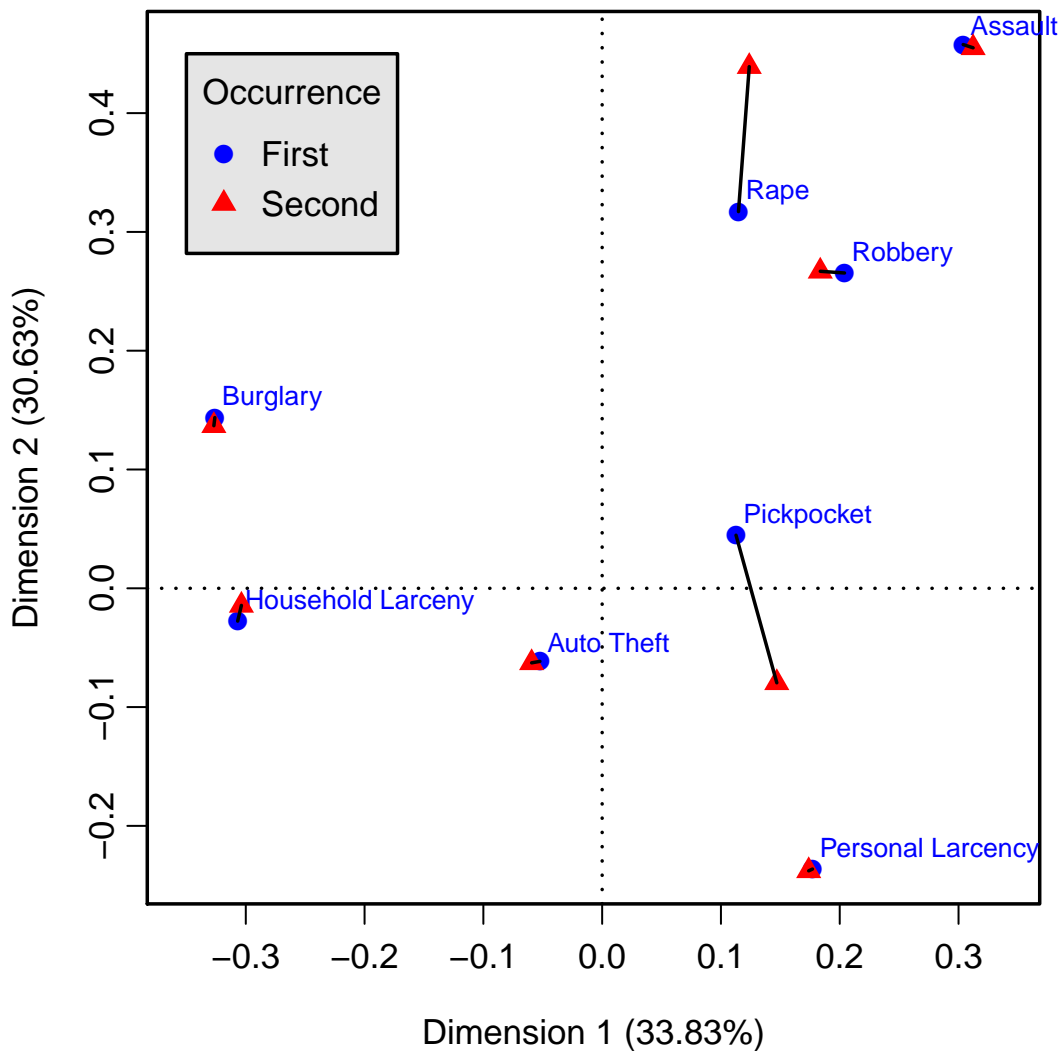


Figure 6.3: 2D CA solution for the repeat victimization data fig:ca-victims-plot

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.

△

6.3 Properties of category scores

{sec:ca-scores}

This section illustrates several properties of the correspondence analysis scores through calculation and visualization.

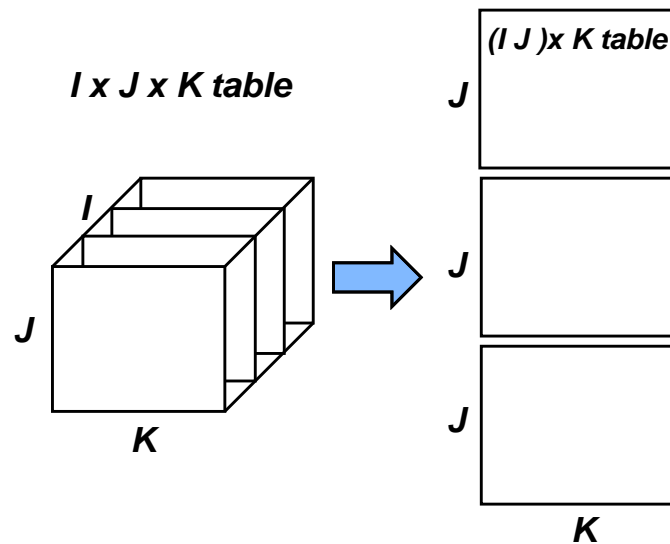


Figure 6.4: Stacking approach for a three-way table. Two of the table variables are combined interactively to form the rows of a two-way table.

{fig:stacking}

TODO: This section requires a lot of custom programming. Maybe useful, but for now, I'm leaving this until later, or just delete this section.

6.3.1 Optimal category scores

{sec:ca-optimal-scores}

6.3.2 Simultaneous linear regressions

{sec:ca-linreg}

6.4 Multi-way tables: Stacking and other tricks

{sec:ca-multiway}

A three- or higher-way table can be analyzed by correspondence analysis in several ways. Multiple correspondence analysis (MCA), described in Section 6.5, is an extension of simple correspondence analysis which 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 a useful one.

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 6.3. 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

$$m_{[ij]k} = \frac{n_{ij+} n_{++k}}{n} \quad (6.6)$$

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 analysed 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.

Table 6.1: Each way of stacking a three-way table corresponds to a loglinear model

Stacking structure	Loglinear model
$(I \times J) \times K$	$[AB][C]$
$I \times (J \times K)$	$[A][BC]$
$J \times (I \times K)$	$[B][AC]$

Interactive coding in R

In the general case of an n -way table, the stacking approach is similar to that used by `fstack()` and `structable()` 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. However, those functions don't create a suitable matrix or table that can be used as input to `ca()`.

TODO: The rest here depends on writing general functions for this: `as.matrix.fstack()`, or using `reshape2::acast()`.

EXAMPLE 6.4: 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 age, sex, 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 age 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: simply paste the level values together to form a new, composite factor, called `age_sex` here.

```
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 is 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 6.4.

```
op <- par(cex=1.3, mar=c(4, 4, 1, 1)+.1)
plot(suicide.ca)
par(op)
```

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

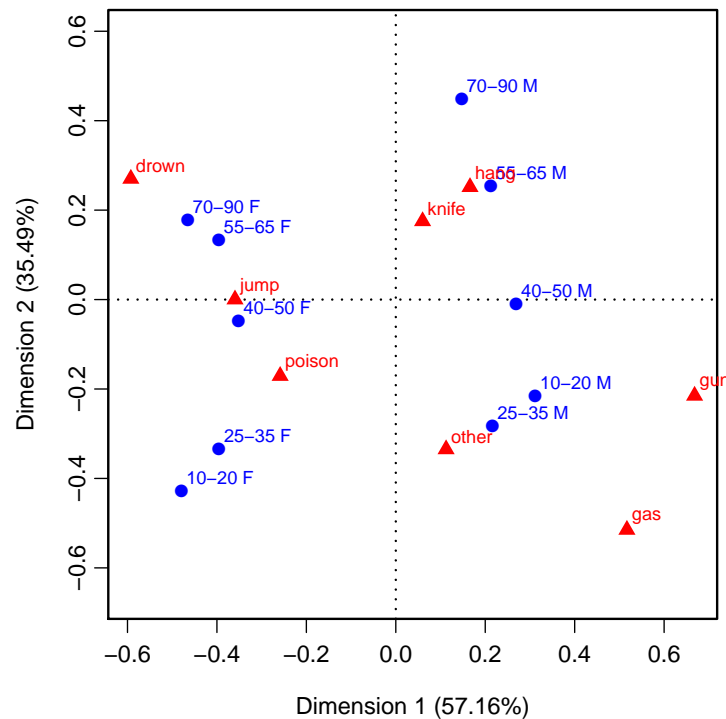


Figure 6.5: 2D CA solution for the stacked [AgeSex][Method] table of the suicide data fig:ca-suicide-plot

dimension is mostly ordered by age with younger groups at the top and older groups at the bottom. 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. △

{ex:suicide2}

EXAMPLE 6.5: 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 6.4 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 analysed 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.

```
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 6.5) 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 for females the excess prevalence of methods varies somewhat less with age than it does for males.

△

6.4.1 Marginal tables and supplementary variables

```
{ca:marginal}
```

An n -way table in frequency form or case form is automatically collapsed over factors which 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 which are listed.

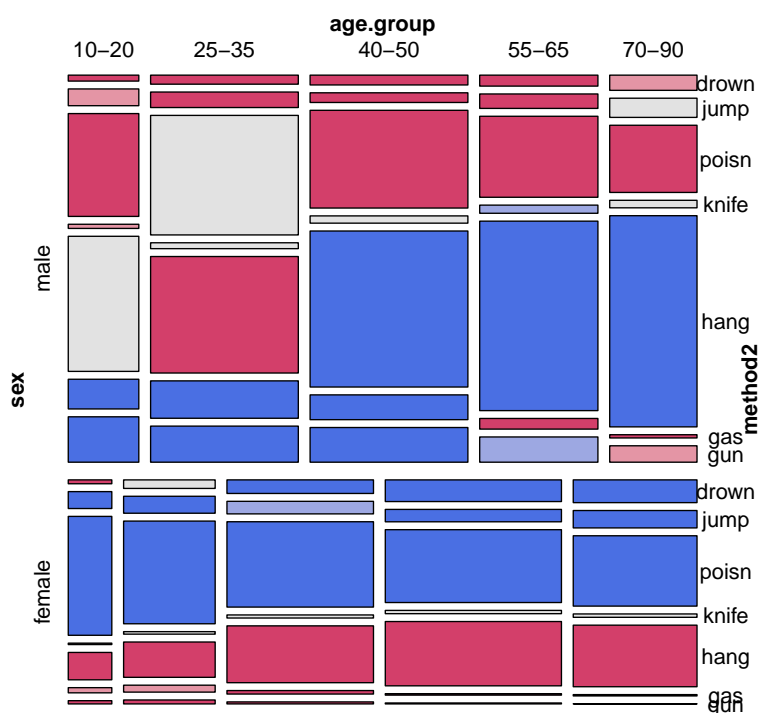


Figure 6.6: Mosaic display showing deviations from the model [AgeSex][Method] for the suicide data

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 trick to extend traditional CA to higher-way tables.

To illustrate, the code below list 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 metho is essentially one-dimensional. We plot the CA results as shown below (see Figure 6.6), and add a line connecting the supplementary points for sex.

```
op <- par(cex=1.3, mar=c(4, 4, 1, 1)+.1)
res <- plot(suicide.ca2s, pch=c(16, 15, 17, 24))
lines(res$rows[6:7,])
par(op)
```

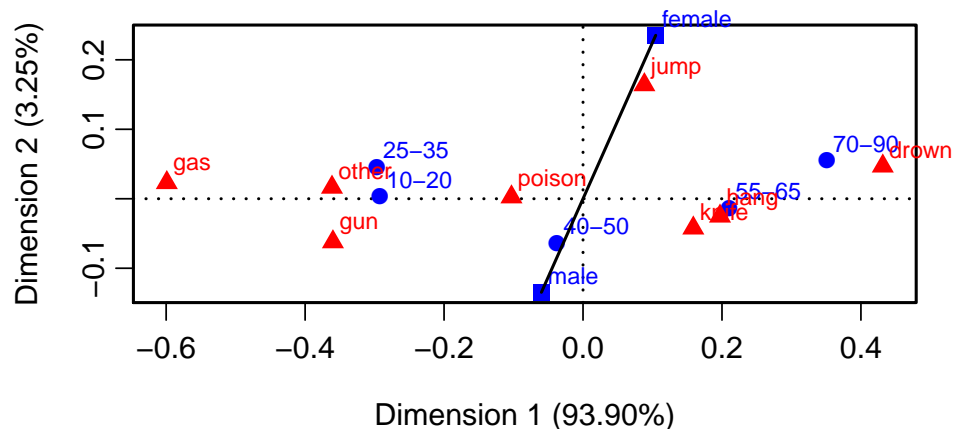


Figure 6.7: 2D CA solution for the [Age] [Method] marginal table. Category points for Sex are shown as supplementary points

Comparing this graph with Figure 6.4, 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 6.6 is the Dimension 1 ordering of both age and method.

However, as in Figure 6.4, the supplementary points for sex point toward the methods that are more prevalent for females and males.

6.5 Multiple correspondence analysis

{sec:mca}

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 6.4 don't 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, \mathbf{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 5.5). 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.

TODO: I've run into a wall here. The `ca` functions `mjca()` and `plot.mjca()` are too limited for the plots I want to do here.

6.5.1 Bivariate MCA

{sec:mca-bi}

For the hair color, eye color data, the indicator matrix \mathbf{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 `outer()` to compute the dummy (0/1) variables for the levels of hair color (h1–h4) and eye color (e1–e4).

```
haireye <- margin.table(HairEyeColor, 1:2)

haireye.df <- as.data.frame(haireye)
dummy.hair <- 0+outer(haireye.df$Hair, levels(haireye.df$Hair), `==`)
colnames(dummy.hair) <- paste0('h', 1:4)
dummy.eye <- 0+outer(haireye.df$Eye, levels(haireye.df$Eye), `==`)
colnames(dummy.eye) <- paste0('e', 1:4)

haireye.df <- data.frame(haireye.df, dummy.hair, dummy.eye)
haireye.df

##      Hair   Eye Freq h1 h2 h3 h4 e1 e2 e3 e4
## 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
```

6.6 Extended MCA: Showing interactions in 2^Q tables

```
{sec:ca-mcainter}
```

6.7 Biplots for contingency tables

```
{sec:biplot}
```

6.8 Chapter summary

```
{sec:ca-summary}
```

6.9 Further reading

```
{sec:ca-reading}
```

6.10 Lab exercises

```
{sec:ca-lab}
```

References

- Aberdein, J. and Spiegelhalter, D. (2013). Have London's roads become more dangerous for cyclists? *Significance*, 10(6), 46–48.
- Agresti, A. (1990). *Categorical Data Analysis*. New York: Wiley-Interscience.
- Agresti, A. (1996). *An Introduction to Categorical Data Analysis*. New York: Wiley Interscience.
- Agresti, A. (2002). *Categorical Data Analysis*. Wiley Series in Probability and Statistics. New York: Wiley-Interscience [John Wiley & Sons], 2nd edn.
- Agresti, A. (2013). *Categorical Data Analysis*. Wiley Series in Probability and Statistics. New York: Wiley-Interscience [John Wiley & Sons], 3rd edn.
- Aitchison, J. (1986). *The Statistical Analysis of Compositional Data*. London: Chapman and Hall.
- Andersen, E. B. (1991). *Statistical Analysis of Categorical Data*. Berlin: Springer-Verlag, 2nd edn.
- Anderson, E. (1935). The irises of the Gaspé peninsula. *Bulletin of the American Iris Society*, 35, 2–5.
- Andrews, D. F. and Herzberg, A. M. (1985). *Data: A Collection of Problems from Many Fields for the Student and Research Worker*. New York, NY: Springer-Verlag.
- Antonio, A. L. M. and Crespi, C. M. (2010). Predictors of interobserver agreement in breast imaging using the breast imaging reporting and data system. *Breast Cancer Research and Treatment*, 120(3), 539–546.
- Arbuthnot, J. (1710). An argument for devine providence, taken from the constant regularity observ'd in the births of both sexes. *Philosophical Transactions*, 27, 186–190. Published in 1711.
- Ashford, J. R. and Snowden, R. D. (1970). Multivariate probit analysis. *Biometrics*, 26, 535–546.
- Bangdiwala, S. I. (1985). A graphical test for observer agreement. In *Proceeding of the International Statistics Institute*, vol. 1, (pp. 307–308). Amsterdam: ISI.
- Bangdiwala, S. I. (1987). Using SAS software graphical procedures for the observer agreement chart. *Proceedings of the SAS User's Group International Conference*, 12, 1083–1088.
- Bartlett, M. S. (1935). Contingency table interactions. *Journal of the Royal Statistical Society, Supplement*, 2, 248–252.

- Bertin, J. (1981). *Graphics and Graphic Information-processing*. New York: de Gruyter. (trans. W. Berg and P. Scott).
- Bertin, J. (1983). *Semiology of Graphics*. Madison, WI: University of Wisconsin Press. (trans. W. Berg).
- Bickel, P. J., Hammel, J. W., and O'Connell, J. W. (1975). Sex bias in graduate admissions: Data from Berkeley. *Science*, 187, 398–403.
- Birch, M. W. (1963). An algorithm for the logarithmic series distributions. *Biometrics*, 19, 651–652.
- Bishop, Y. M. M., Fienberg, S. E., and Holland, P. W. (1975). *Discrete Multivariate Analysis: Theory and Practice*. Cambridge, MA: MIT Press.
- Böhning, D. (1983). Maximum likelihood estimation of the logarithmic series distribution. *Statistische Hefte (Statistical Papers)*, 24(1), 121–140.
- Carlyle, T. (1840). *Chartism*. London: J. Fraser.
- Chambers, J. M., Cleveland, W. S., Kleiner, B., and Tukey, P. A. (1983). *Graphical Methods for Data Analysis*. Belmont, CA: Wadsworth.
- Cicchetti, D. V. and Allison, T. (1971). A new procedure for assessing reliability of scoring EEG sleep recordings. *American Journal of EEG Technology*, 11, 101–109.
- Cleveland, W. S. (1993a). A model for studying display methods of statistical graphics. *Journal of Computational and Graphical Statistics*, 2, 323–343.
- Cleveland, W. S. (1993b). *Visualizing Data*. Summit, NJ: Hobart Press.
- Cleveland, W. S. and McGill, R. (1984). Graphical perception: Theory, experimentation and application to the development of graphical methods. *Journal of the American Statistical Association*, 79, 531–554.
- Cleveland, W. S. and McGill, R. (1985). Graphical perception and graphical methods for analyzing scientific data. *Science*, 229, 828–833.
- Cohen, A. (1980). On the graphical display of the significant components in a two-way contingency table. *Communications in Statistics—Theory and Methods*, A9, 1025–1041.
- Cohen, J. (1960). A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20, 37–46.
- Cohen, J. (1968). Weighted kappa: Nominal scale agreement with provision for scaled disagreement or partial credit. *Psychological Bulletin*, 70, 213–220.
- Edwards, A. W. F. (1958). An analysis of geissler's data on the human sex ratio. *Annals of Human Genetics*, 23(1), 6–15.
- Emerson, J. W., Green, W. A., Schloerke, B., Crowley, J., Cook, D., Hofmann, H., and Wickham, H. (2013). The generalized pairs plot. *Journal of Computational and Graphical Statistics*, 22(1), 79–91.

- Fienberg, S. E. (1975). Perspective Canada as a social report. *Social Indicators Research*, 2, 153–174.
- Fienberg, S. E. (1980). *The Analysis of Cross-Classified Categorical Data*. Cambridge, MA: MIT Press, 2nd edn.
- Fisher, R. A. (1925). *Statistical Methods for Research Workers*. London: Oliver & Boyd.
- Fisher, R. A. (1936a). Has Mendel's work been rediscovered? *Annals of Science*, 1, 115–137.
- Fisher, R. A. (1936b). The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 8, 379–388.
- Fisher, R. A., Corbet, A. S., and Williams, C. B. (1943). The relation between the number of species and the number of individuals. *Journal of Animal Ecology*, 12, 42.
- Fleiss, J. L. (1973). *Statistical Methods for Rates and Proportions*. New York: John Wiley and Sons.
- Fleiss, J. L. and Cohen, J. (1972). The equivalence of weighted kappa and the intraclass correlation coefficient as measures of reliability. *Educational and Psychological Measurement*, 33, 613–619.
- Fleiss, J. L., Cohen, J., and Everitt, B. S. (1969). Large sample standard errors of kappa and weighted kappa. *Psychological Bulletin*, 72, 332–327.
- Fox, J. and Weisberg, S. (2011). *An R Companion to Applied Regression*. Thousand Oaks CA: Sage, 2nd edn.
- Friendly, M. (1991). *SAS System for Statistical Graphics*. Cary, NC: SAS Institute, 1st edn.
- Friendly, M. (1992). Mosaic displays for loglinear models. In ASA, *Proceedings of the Statistical Graphics Section*, (pp. 61–68). Alexandria, VA.
- Friendly, M. (1994a). A fourfold display for 2 by 2 by K tables. Tech. Rep. 217, York University, Psychology Dept.
- Friendly, M. (1994b). Mosaic displays for multi-way contingency tables. *Journal of the American Statistical Association*, 89, 190–200.
- Friendly, M. (1994c). SAS/IML graphics for fourfold displays. *Observations*, 3(4), 47–56.
- Friendly, M. (1997). Conceptual models for visualizing contingency table data. In M. Greenacre and J. Blasius, eds., *Visualization of Categorical Data*, chap. 2, (pp. 17–35). San Diego, CA: Academic Press.
- Friendly, M. (1999a). Extending mosaic displays: Marginal, conditional, and partial views of categorical data. *Journal of Computational and Graphical Statistics*, 8(3), 373–395.
- Friendly, M. (1999b). Extending mosaic displays: Marginal, conditional, and partial views of categorical data. *Journal of Computational and Graphical Statistics*, 8(3), 373–395.
- Friendly, M. (2000). *Visualizing Categorical Data*. Cary, NC: SAS Institute.

- Friendly, M. (2003). Visions of the past, present and future of statistical graphics: An ideographic view. American Psychological Association. Toronto, ON, URL: <http://datavis.ca/papers/apa-2x2.pdf>.
- Friendly, M. (2013). Comment on the generalized pairs plot. *Journal of Computational and Graphical Statistics*, 22(1), 290–291.
- Friendly, M. and Kwan, E. (2003). Effect ordering for data displays. *Computational Statistics and Data Analysis*, 43(4), 509–539.
- Friendly, M. and Kwan, E. (2011). Comment (graph people versus table people). *Journal of Computational and Graphical Statistics*, 20(1), 18–27.
- Gart, J. J. and Zweifl, J. R. (1967). On the bias of various estimators of the logit and its variance with applications to quantal bioassay. *Biometrika*, 54, 181–187.
- Geissler, A. (1889). Beitrage zur frage des geschlechts verhaltnisses der geborenen. *Z. K. Sachsischen Statistischen Bureaus*, 35(1), n.p.
- Gifi, A. (1981). *Nonlinear Multivariate Analysis*. The Netherlands: Department of Data Theory, University of Leiden.
- Goodman, L. A. (1973). The analysis of multidimensional contingency tables when some variables are posterior to others: A modified path analysis approach. *Biometrika*, 60, 179–192.
- Goodman, L. A. (1979). Simple models for the analysis of association in cross-classifications having ordered categories. *Journal of the American Statistical Association*, 74, 537–552.
- Goodman, L. A. (1981). Association models and canonical correlation in the analysis of cross-classifications having ordered categories. *Journal of the American Statistical Association*, 76(374), 320–334.
- Goodman, L. A. (1985). The analysis of cross-classified data having ordered and/or unordered categories: Association models, correlation models, and asymmetry models for contingency tables with or without missing entries. *Annals of Statistics*, 13(1), 10–69.
- Goodman, L. A. (1986). Some useful extensions of the usual correspondence analysis approach and the usual log-linear models approach in the analysis of contingency tables. *International Statistical Review*, 54(3), 243–309. With a discussion and reply by the author.
- Greenacre, M. (1984). *Theory and Applications of Correspondence Analysis*. London: Academic Press.
- Greenacre, M. (1989). The Carroll-Green-Schaffer scaling in correspondence analysis: A theoretical and empirical appraisal. *Journal of Marketing Research*, 26, 358–365.
- Greenacre, M. (1997). Diagnostics for joint displays in correspondence analysis. In J. Blasius and M. Greenacre, eds., *Visualization of Categorical Data*, (pp. 221–238). Academic Press.
- Greenacre, M. and Hastie, T. (1987). The geometric interpretation of correspondence analysis. *Journal of the American Statistical Association*, 82, 437–447.
- Greenacre, M. J. (2007). *Correspondence analysis in practice*. Boca Raton: Chapman Hall/CRC.

- Greenwood, M. and Yule, G. U. (1920). An inquiry into the nature of frequency distributions of multiple happenings, with particular reference to the occurrence of multiple attacks of disease or repeated accidents. *Journal of the Royal Statistical Society, Series A*, 83, 255–279.
- Haberman, S. J. (1979). *The Analysis of Qualitative Data: New Developments*, vol. II. New York: Academic Press.
- Haldane, J. B. S. (1955). The estimation and significance of the logarithm of a ratio of frequencies. *Annals of Human Genetics*, 20, 309–311.
- Hartigan, J. A. and Kleiner, B. (1981). Mosaics for contingency tables. In W. F. Eddy, ed., *Computer Science and Statistics: Proceedings of the 13th Symposium on the Interface*, (pp. 268–273). New York, NY: Springer-Verlag.
- Hartigan, J. A. and Kleiner, B. (1984). A mosaic of television ratings. *The American Statistician*, 38, 32–35.
- Heuer, J. (1979). *Selbstmord Bei Kinder Und Jugendlichen*. Stuttgart: Ernst Klett Verlag. [Suicide by children and youth.].
- Hilbe, J. (2011). *Negative Binomial Regression*. Cambridge University Press, 2nd edn.
- Hoaglin, D. C. (1980). A poissonness plot. *The American Statistician*, 34, 146–149.
- Hoaglin, D. C. and Tukey, J. W. (1985). Checking the shape of discrete distributions. In D. C. Hoaglin, F. Mosteller, and J. W. Tukey, eds., *Exploring Data Tables, Trends and Shapes*, chap. 9. New York: John Wiley and Sons.
- Hofmann, H. and Vendettuoli, M. (2013). Common angle plots as perception-true visualizations of categorical associations. *IEEE Transactions on Visualization and Computer Graphics*, 19(12), 2297–2305.
- Holm, S. (1979). A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics*, 6(2), 65–70.
- Hout, M., Duncan, O. D., and Sobel, M. E. (1987). Association and heterogeneity: Structural models of similarities and differences. *Sociological Methodology*, 17, 145–184.
- Inselberg, A. (1985). The plane with parallel coordinates. *The Visual Computer*, 1, 69–91.
- Inselberg, A. (1989). Discovering multi-dimensional structure using parallel coordinates. In *Proc. Amer. Stat. Assoc, Sec. on Stat. Graphics*, (pp. 1–16). American Statistical Association.
- Jinkinson, R. A. and Slater, M. (1981). Critical discussion of a graphical method for identifying discrete distributions. *The Statistician*, 30, 239–248.
- Johnson, N. L., Kotz, S., and Kemp, A. W. (1992). *Univariate Discrete Distributions*. New York, NY: John Wiley and Sons, 2nd edn.
- Kemp, A. W. and Kemp, C. D. (1991). Weldon’s dice data revisited. *The American Statistician*, 45, 216–222.
- Kendall, M. G. and Stuart, A. (1961). *The Advanced Theory of Statistics*, vol. 2. London: Griffin.
- Kendall, M. G. and Stuart, A. (1963). *The Advanced Theory of Statistics*, vol. 1. London: Griffin.

- Koch, G. and Edwards, S. (1988). Clinical efficiency trials with categorical data. In K. E. Peace, ed., *Biopharmaceutical Statistics for Drug Development*, (pp. 403–451). New York: Marcel Dekker.
- Kosambi, D. D. (1949). Characteristic properties of series distributions. *Proceedings of the National Institute of Science of India*, 15, 109–113.
- Kosara, R., Bendix, F., and Hauser, H. (2006). Parallel sets: Interactive exploration and visual analysis of categorical data. *IEEE Transactions on Visualization and Computer Graphics*, 12(4), 558–568.
- Kosslyn, S. M. (1985). Graphics and human information processing: A review of five books. *Journal of the American Statistical Association*, 80, 499–512.
- Kosslyn, S. M. (1989). Understanding charts and graphs. *Applied Cognitive Psychology*, 3, 185–225.
- Kundel, H. L. and Polansky, M. (2003). Measurement of observer agreement. *Radiology*, 228(2), 303–308.
- Labby, Z. (2009). Weldon’s dice, automated. *Chance*, 22(4), 6–13.
- Landis, J. R. and Koch, G. G. (1977). The measurement of observer agreement for categorical data. *Biometrics*, 33, 159–174.
- Landis, R. J., Heyman, E. R., and Koch, G. G. (1978). Average partial association in three-way contingency tables: A review and discussion of alternative tests. *International Statistical Review*, 46, 237–254.
- Lebart, L., Morineau, A., and Warwick, K. M. (1984). *Multivariate Descriptive Statistical Analysis: Correspondence Analysis and Related Techniques for Large Matrices*. New York: John Wiley and Sons.
- Lee, A. J. (1997). Modelling scores in the Premier League: Is Manchester United really the best? *Chance*, 10(1), 15–19.
- Leifeld, P. (2013). texreg: Conversion of statistical model output in r to latex and html tables. *Journal of Statistical Software*, 55(8), 1–24.
- Lewandowsky, S. and Spence, I. (1989). The perception of statistical graphs. *Sociological Methods & Research*, 18, 200–242.
- Lindsey, J. K. (1995). *Modelling Frequency and Count Data*. Oxford, UK: Oxford University Press.
- Lindsey, J. K. and Altham, P. M. E. (1998). Analysis of the human sex ratio by using overdispersion models. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 47(1), 149–157.
- Lindsey, J. K. and Mersch, G. (1992). Fitting and comparing probability distributions with log linear models. *Computational Statistics and Data Analysis*, 13, 373–384.
- Mersey, L. (1912). Report on the loss of the “Titanic” (S. S.). Parliamentary command paper 6352.

- Meyer, D., Zeileis, A., and Hornik, K. (2006). The strucplot framework: Visualizing multi-way contingency tables with *vcd*. *Journal of Statistical Software*, 17(3), 1–48.
- Mosteller, F. and Wallace, D. L. (1963). Inference in an authorship problem. *Journal of the American Statistical Association*, 58(302), 275–309.
- Mosteller, F. and Wallace, D. L. (1984). *Applied Bayesian and Classical Inference: The Case of the Federalist Papers*. New York, NY: Springer-Verlag.
- Noack, A. (1950). A class of random variables with discrete distributions. *Annals of Mathematical Statistics*, 21, 127–132.
- Ord, J. K. (1967). Graphical methods for a class of discrete distributions. *Journal of the Royal Statistical Society, Series A*, 130, 232–238.
- Pearson, K. (1900). On the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen by random sampling. *Philosophical Magazine*, 50(5th Series), 157–175.
- Riedwyl, H. and Schüpbach, M. (1983). Siebdiagramme: Graphische darstellung von kontingenztafeln. Tech. Rep. 12, Institute for Mathematical Statistics, University of Bern, Bern, Switzerland.
- Riedwyl, H. and Schüpbach, M. (1994). Parquet diagram to plot contingency tables. In F. Faulbaum, ed., *Softstat '93: Advances In Statistical Software*, (pp. 293–299). New York: Gustav Fischer.
- Schonlau, M. (2003). Visualizing categorical data arising in the health sciences using hammock plots. In *Proceedings of the Section on Statistical Graphics*. RAND Corporation, American Statistical Association.
- Shrout, P. E. and Fleiss, J. L. (1979). Intraclass correlations: Uses in assessing rater reliability. *Psychological Bulletin*, 86, 420–428.
- Simpson, E. H. (1951). The interpretation of interaction in contingency tables. *Journal of the Royal Statistical Society, Series B*, 30, 238–241.
- Skellam, J. G. (1948). A probability distribution derived from the binomial distribution by regarding the probability of success as variable between the sets of trials. *Journal of the Royal Statistical Society. Series B (Methodological)*, 10(2), 257–261.
- Snee, R. D. (1974). Graphical display of two-way contingency tables. *The American Statistician*, 28, 9–12.
- Spence, I. (1990). Visual psychophysics of simple graphical elements. *Journal of Experimental Psychology: Human Perception and Performance*, 16, 683–692.
- Spence, I. and Lewandowsky, S. (1990). Graphical perception. In J. Fox and J. S. Long, eds., *Modern Methods of Data Analysis*, chap. 1, (pp. 13–57). Sage Publications, Inc.
- Srole, L., Langner, T. S., Michael, S. T., Kirkpatrick, P., Opler, M. K., and Rennie, T. A. C. (1978). *Mental Health in the Metropolis: The Midtown Manhattan Study*. New York: NYU Press.

- Stokes, M. E., Davis, C. S., and Koch, G. G. (2000). *Categorical Data Analysis Using the SAS System*. Cary, NC: SAS Institute, 2nd edn.
- Theus, M. and Lauer, S. R. W. (1999). Visualizing loglinear models. *Journal of Computational and Graphical Statistics*, 8(3), 396–412.
- Thornes, B. and Collard, J. (1979). *Who Divorces?* London: Routledge & Kegan.
- Tufte, E. (2006). *Beautiful Evidence*. Cheshire, CT: Graphics Press.
- Tufte, E. R. (1983). *The Visual Display of Quantitative Information*. Cheshire, CT: Graphics Press.
- Tufte, E. R. (1990). *Envisioning Information*. Cheshire, CT: Graphics Press.
- Tufte, E. R. (1997). *Visual Explanations: Images and Quantities, Evidence and Narrative*. Cheshire, CT: Graphics Press.
- Tukey, J. W. (1977). *Exploratory Data Analysis*. Reading, MA: Addison Wesley.
- Tukey, J. W. (1993). Graphic comparisons of several linked aspects: Alternative and suggested principles. *Journal of Computational and Graphical Statistics*, 2(1), 1–33.
- Upton, G. J. G. (1976). The diagrammatic representation of three-party contests. *Political Studies*, 24, 448–454.
- Upton, G. J. G. (1994). Picturing the 1992 British general election. *Journal of the Royal Statistical Society, Series A*, 157(Part 2), 231–252.
- van der Heijden, P. G. M., de Falguerolles, A., and de Leeuw, J. (1989). A combined approach to contingency table analysis using correspondence analysis and log-linear analysis. *Applied Statistics*, 38(2), 249–292.
- van der Heijden, P. G. M. and de Leeuw, J. (1985). Correspondence analysis used complementary to loglinear analysis. *Psychometrika*, 50, 429–447.
- von Bortkiewicz, L. (1898). *Das Gesetz der Kleinen Zahlen*. Leipzig: Teubner.
- Von Eye, A. and Mun, E. (2006). *Analyzing Rater Agreement: Manifest Variable Methods*. Taylor & Francis.
- Wainer, H. (1996). Using trilinear plots for NAEP state data. *Journal of Educational Measurement*, 33(1), 41–55.
- Wegman, E. J. (1990). Hyperdimensional data analysis using parallel coordinates. *Journal of the American Statistical Association*, 85(411), 664–675.
- Wickham, H. (2009). *ggplot2: elegant graphics for data analysis*. Springer New York.
- Wickham, H. and Chang, W. (2013). *ggplot2: An implementation of the Grammar of Graphics*. R package version 0.9.3.1.
- Wilkinson, L. (2005). *The Grammar of Graphics*. New York: Springer, 2nd edn.
- Wimmer, G. and Altmann, G. (1999). *Thesaurus of univariate discrete probability distributions*. Stamm.

- Woolf, B. (1995). On estimating the relation between blood group and disease. *Annals of Human Genetics*, 19, 251–253.
- Zeileis, A., Meyer, D., and Hornik, K. (2007). Residual-based shadings for visualizing (conditional) independence. *Journal of Computational and Graphical Statistics*, 16(3), 507–525.
- Zelterman, D. (1999). *Models for Discrete Data*. New York: Oxford University Press.

Index

- β -binomial, 88
- ϕ coefficient, 194
-
- addmargins, 28
- addmargins(), 34
- aggregate(), 35
- agreement
 - Cohen's κ , 127
 - intraclass correlation, 127
 - observer agreement chart, 129
 - partial, 130
- agreementplot(), 130, 131
- anova(), 159
- aperm(), 27
- apply(), 35, 36, 62, 108, 168, 171
- array(), 20
- as.data.frame(), 37, 38
- as.data.frame.table(), 37, 42
- as.matrix.ftable(), 205
- as.numeric(), 38
- as.table(), 26
- aspect ratio, 200
- assoc(), 147
- association, 6
- association graph, 156
- association plot, 123
- association plots, 98
- assocstats(), 7, 104, 105, 108, 137, 138
- asymmetric map, 196
- axes
 - equating, 200
-
- baseline models, 155
- Binary events, 48
- binary variables, 2
- binomial distribution, 56–59, 83, 84
- binomial samples, 102
- biplot, 193
- biplot(), 196
-
- c(), 16, 24
- ca package, 196, 197, 199, 211
- CA(), 197
- ca(), 196–198, 204, 209
- canonical analysis of categorical data, 193
- canonical correlation, 193
- case form, 3
- categorical variable, 2
- cbind(), 20
- chisq.test(), 7, 152
- CMHtest(), 105–108, 137, 138
- Cochran-Mantel-Haenszel tests, 105
 - general association, 105, 106
 - linear association, 105, 106
 - row means differ, 105
- Cohen's κ , 127
- coinddep_test(), 151, 171
- collapse.table(), 36, 37
- common angle plot, 186
- common angle plots, 185
- compositional data, 133
- conditional distributions, 101
- conditional(), 161
- confint.Kappa(), 129
- contingency table, 97
- controlled comparison, 115
- coord_flip(), 186
- coplot, 166
- corresp(), 196
- correspondence analysis
 - asymmetric map, 196
 - principal coordinates, 195
 - properties, 195–196
 - stacking, 203–209
 - standard coordinates, 195
 - supplementary variables, 209
 - symmetric map, 196
 - two-way tables, 194–202
- correspondence matrix, 194

- cotabplot(), 147, 166, 171, 173
- count, 48
- count data, 4
- count metamer, 84
- cross-sectional study, 102
- CrossTable(), 101
- cut(), 184
- data frame, 15
- data sets
 - Arbuthnot, 48
 - Arthritis, 3, 5, 23, 40, 100, 108, 152, 179
 - arthritis treatment, 104, 108
 - Bartlett, 174, 181, 192
 - Berkeley admissions, 114
 - Bundesliga, 94
 - Butterfly, 80
 - CoalMiners, 116
 - CyclingDeaths, 54, 95
 - DanishWelfare, 44
 - DaytonSurvey, 35, 36
 - death by horse kick, 82, 86
 - Depends, 95
 - Employment, 166, 168
 - Federalist, 82, 93
 - Federalist papers, 82
 - Geissler, 45, 94
 - HairEye, 7
 - HairEyeColor, 27, 98, 124, 158, 197
 - HorseKick, 71
 - HorseKicks, 33, 34, 38, 52, 74, 86
 - Hospital, 138
 - iris, 182, 184
 - JobSat, 26
 - Lifeboats, 134, 135, 138
 - Mammograms, 131
 - Mental, 99, 200
 - MSPatients, 126, 131
 - multiple sclerosis diagnosis, 132
 - PreSex, 163, 174
 - Punishment, 170
 - RepVict, 201
 - Saxony, 45, 90
 - SexualFun, 126, 129
 - Space shuttle disaster, 8–9
 - struc, 189
 - Suicide, 205
 - Titanic, 134, 185
 - UCBADmissions, 44
 - UCBAdmissions, 46, 98, 109, 178
 - UKSoccer, 45, 60, 61, 94, 153
 - visual acuity, 120
 - VisualAcuity, 120, 138
 - WeldonDice, 51, 57, 74
 - WomenQueue, 82, 93
- data.frame(), 20, 24, 38
- datasets(), 44
- dbinom(), 56, 58
- ddoublebinom(), 90
- ddply(), 36
- dgeom(), 56
- dim(), 18
- dimnames(), 18
- direct.label(), 64
- directlabels package, 63
- dispersion parameter, 66
- distplot(), 70, 85, 86, 88, 92, 94, 95
- dlogseries(), 56
- dnbinom(), 56, 67
- double binomial, 89
- doubledecker plots, 145
- doubledecker(), 147
- dpois(), 56, 63
- dual scaling, 193
- effect ordering, 145, 184
- effect plot, 186
- expand.dft(), 40, 62
- expand.grid(), 24, 58, 63, 67
- explanatory, 98
- explanatory variables, 5
- exponential family, 89
- FactoMineR package, 197
- factor, 20
- fitdistr(), 72
- fitted(), 188
- foreign package, 21
- fourfold display
 - confidence rings, 113
- fourfold display, 110–118
- fourfold(), 113, 115, 137
- frequency, 48
- frequency data, 4
- frequency form, 4
- ftable(), 25, 29, 30, 45, 46, 204

- gdata package, 21
- generalized pairs plot, 174
- geometric distribution, 69
- GGally package, 179
- ggparallel package, 185
- ggparallel(), 185, 186
- ggplot2 package, 10, 63–65, 133, 179
- ggtern package, 133
- ggtern(), 133
- glm(), 5, 89–91
- goodfit(), 70, 73, 74, 79, 88, 92–95
- goodness-of-fit, 71
- gpairs package, 174, 179
- gpairs(), 179, 180
- grapcon functions, 145
- grid package, 148, 179
- hammock plot, 186
- hammock plots, 185
- hanging rootogram, 77
- high-order terms, 155
- HistData package, 48
- homogeneity analysis, 193
- homogeneity of association, 6
- ICC(), 127
- independence_table(), 118
- index of dispersion, 62
- indicator matrix, 211
- inertia, 194
- inter-rater agreement, 98
- interactive coding, 203
- interp(), 150
- intraclass correlation, 127
- joint distribution, 101
- joint independence, 123
- joint(), 161
- Kappa(), 129
- Kruskal-Wallis test, 105
- labeling_border(), 146
- labeling_cboxed(), 146
- labeling_cells(), 146
- labeling_conditional(), 146
- labeling_doubledecker(), 146
- labeling_lboxed(), 146
- labeling_left(), 146
- labeling_left2(), 146
- labeling_list(), 146
- labeling_residuals(), 146
- labeling_value(), 146
- lattice package, 59, 63, 182
- legend(), 202
- legend_fixed(), 146
- legend_resbased(), 146
- levels(), 42
- likelihood ratio test, 105
- line width illusion, 185
- lines(), 57
- lm(), 5
- log odds, 8, 102
- log odds ratio, 103
- logarithmic series distribution, 69–70, 80, 82
- logit, 102
- logit function, 8
- loglin(), 161
- loglin2formula(), 161
- loglin2string(), 161
- loglinear model, 109
- loglm(), 110, 122, 138, 142, 155, 159, 161, 167, 168, 171, 188
- margin.table, 28
- margin.table(), 28, 35
- marginal distributions, 101
- marginal frequencies, 101
- marginal homogeneity, 127, 131
- marginal model, 209
- Marimekko chart, 148
- markov(), 161
- MASS package, 40, 72, 110, 159, 196
- matrix, 17
- matrix(), 18
- mca(), 40, 196
- mean-square contingency coefficient, 194
- mjca(), 197, 211
- mosaic display, 7, 139
- mosaic matrix, 174, 211
- mosaic(), 32, 147, 152, 155, 161, 166, 169, 173
- mosaic3d(), 181, 182, 190
- mosaicplot(), 30, 146
- multinomial sample, 102
- multiple correspondence analysis
 - bivariate, 211

- `mutual()`, 161
- negative binomial distribution, 65–69, 82, 84
- nominal, 2
- observer agreement, 126
- observer agreement chart, 129
- odds, 102
- odds ratio, 12, **102**, 102–104, 109
- `oddsratio()`, 116
- optimal scaling, 193
- Ord plot, 79–83
- `Ord_plot()`, 80–83, 85, 92, 95
- `ordered()`, 22, 26
- ordinal, 2
- `outer()`, 211
- overdispersion, 65
- `p.adjust()`, 115
- package
 - `ca`, 196, 197, 199, 211
 - `directlabels`, 63
 - `FactoMineR`, 197
 - `foreign`, 21
 - `gdata`, 21
 - `GGally`, 179
 - `ggparallel`, 185
 - `ggplot2`, 10, 63–65, 133, 179
 - `ggtern`, 133
 - `gpairs`, 174, 179
 - `grid`, 148, 179
 - `HistData`, 48
 - `lattice`, 59, 63, 182
 - `MASS`, 40, 72, 110, 159, 196
 - `plyr`, 36
 - `psych`, 127
 - `rgl`, 181, 196
 - `rmutil`, 90
 - `stats`, 30, 56
 - `TeachingDemos`, 133
 - `texreg`, 33
 - `vcd`, 3, 23, 25, 30, 44, 45, 51, 52, 60, 73, 78, 80, 82, 85, 109, 116, 118, 120, 126, 129, 134, 135, 138, 145, 147, 148, 150, 151, 163, 166, 170, 171, 174, 179, 205, 219
 - `vcdExtra`, 35, 36, 40, 41, 44, 45, 54, 70, 94, 95, 99, 105, 131, 159, 161, 174, 181
 - `XLConnect`, 21
 - `xlsx`, 21
 - `xtable`, 33
- `pairs()`, 147, 174, 176, 189
- `pairs.table()`, 178, 190
- `palette()`, 149
- panel functions, 178
- parallel sets, 185
- `parallelplot()`, 182, 184
- parquet diagram, 119
- partial association, 165
- Pascal distribution, 65
- `paste()`, 27
- `pbinom()`, 56
- `pchisq()`, 72
- `pdoublebinom()`, 90
- `pgeom()`, 56
- phi coefficient, 7
- phi (ϕ) coefficient, 194
- `plogseries()`, 56
- `plot()`, 32, 57, 117, 200
- `plot.ca()`, 196, 198, 199
- `plot.goodfit()`, 77
- `plot.gootfit()`, 93
- `plot.mjca()`, 211
- `plot3d.ca()`, 196
- plyr package, 36
- `plyr::daply()`, 41
- `plyr::summarise()`, 36
- `pnbinom()`, 56
- Poisson distribution, 59–65, 82, 84
- Poissonness plot, 84–87
- Polya distribution, 66
- Polytomous events, 48
- polytomous variables, 2
- power series distributions, 70–71
- `ppois()`, 56, 62
- principal component analysis, 211
- principal components analysis, 193
- principal coordinates, 195
- `print()`, 73
- `print.goodfit()`, 73
- `prop.table()`, 28, 29
- psych package, 127
- `qbinom()`, 56
- `qdoublebinom()`, 90
- `qgeom()`, 56

- qlogseries(), 56
- qnbinom(), 56
- qpois(), 56
- quasi-independence, 120, 121
- radial diagram, 110
- rbinom(), 56
- rdoublebinom(), 90
- read.csv(), 21, 22
- read.delim(), 21
- read.table(), 21, 40, 41
- reciprocal averaging, 193
- rep(), 17
- residuals(), 142
- response, 98
- response variables, 5
- rgeom(), 56
- rgl package, 181, 196
- rlogseries(), 56
- rmutil package, 90
- rnbinom(), 56
- rootogram, 77
- rootogram(), 78, 92, 95
- rownames(), 34
- rpois(), 56
- sample odds ratio, 103
- sample(), 20
- sapply(), 171
- saturated model, 156
- scatterplot matrix, 173, 211
- scree plot, 197
- segments(), 202
- seq(), 17
- seq_loglm(), 162
- set.seed(), 151
- shading_binary(), 146
- shading_Friendly(), 146
- shading_hcl(), 146
- shading_hsv(), 146
- shading_max(), 146, 151, 153
- shading_sieve(), 146
- sieve diagram, 118–123
- sieve diagrams, 97
- sieve(), 147
- simple effects, 165
- Simpson's paradox, 115
- singular value decomposition, 194, 195
- sort(), 26
- spacing_conditional(), 146
- spacing_dimequal(), 146
- spacing_equal(), 146
- spacing_highlighting(), 146
- spacing_increase(), 146
- stacking, 203
- standard coordinates, 195
- stats package, 30, 56
- str(), 18, 27
- stratified analysis, 6, 108
- stratifying variable, 97
- struc_assoc(), 146
- struc_mosaic(), 146
- struc_sieve(), 146
- strucplot framework, 121, 145
- strucplot(), 147
- structable(), 25, 29–31, 36, 45, 46, 145, 204
- subset(), 45
- sum(), 35
- summarise(), 159
- summary(), 73
- summary.goodfit(), 73
- summary.Kappa(), 129
- supplementary variables, 209
- symmetric map, 195, 196
- symmetry, 120, 121
- t(), 32, 34
- table, 28
- table form, 4
- table(), 20, 25, 28, 29, 39
- TeachingDemos package, 133
- ternary plot, 98
- texreg package, 33
- theme(), 65
- toeplitz(), 149
- trilinear plot, 98, 132
- triplot(), 133
- type-token, 54
- variable
 - response γ 5–6
- vcd package, 3, 23, 25, 30, 44, 45, 51, 52, 60, 73, 78, 80, 82, 85, 109, 116, 118, 120, 126, 129, 134, 135, 138, 145, 147, 148, 150, 151, 163, 166, 170, 171, 174, 179, 205, 219

vcdExtra package, 35, 36, 40, 41, 44, 45,
54, 70, 94, 95, 99, 105, 131, 159,
161, 174, 181

vector, 15, 16

visual impact, 115

weighted.mean(), 39, 71

within(), 41, 61

XLConnect package, 21

xlsx package, 21

xtable package, 33

xtable(), 33

xtabs(), 20, 25, 28, 29, 36, 38, 41, 42, 46,
104, 205, 209

xyplot(), 59, 63, 64, 67, 93

This document was produced using:

```
print(sessionInfo(), locale = FALSE)

## R version 3.0.1 (2013-05-16)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
##
## attached base packages:
## [1] grid      stats      graphics  grDevices  utils
## [6] datasets  methods   base
##
## other attached packages:
##  [1] RColorBrewer_1.0-5      gpairs_1.1
##  [3] colorspace_1.2-4       barcode_1.1
##  [5] ggtern_1.0.3.1         MASS_7.3-29
##  [7] gmodels_2.15.4.1       ggplot2_0.9.3.1
##  [9] directlabels_2014.1.31 quadprog_1.5-5
## [11] lattice_0.20-25        xtable_1.7-3
## [13] vcdExtra_0.5-12        gnm_1.0-7
## [15] ca_0.55                vcd_1.3-2
## [17] knitr_1.5
##
## loaded via a namespace (and not attached):
##  [1] dichromat_2.0-0 digest_0.6.4      evaluate_0.5.1
##  [4] formatR_0.10    gdata_2.13.2     gtable_0.1.2
##  [7] gtools_3.3.0    highr_0.3        labeling_0.2
## [10] Matrix_1.1-2-1 munsell_0.4.2    nnet_7.3-7
## [13] plyr_1.8        proto_0.3-10     qvcalc_0.8-8
## [16] relimp_1.0-3    reshape2_1.2.2   scales_0.2.3
## [19] sp_1.0-15       stringr_0.6.2    tcltk_3.0.1
## [22] tools_3.0.1
```