6.5.2 Biadditive biplots

A different use of biplots for contingency tables stems from the close analogy between additive relations for a quantitative response when there is no interaction between factors, and the multiplicative relations for a contingency table when there is no association.

For quantitative data, Bradu and Gabriel (1978) show how the biplot can be used to diagnose additive relations among rows and columns. For example, when a two-way table is well-described by a two-factor ANOVA model with no interaction,

$$y_{ij} = \mu + \alpha_i + \beta_j + \epsilon_{ij} \iff \boldsymbol{Y} \approx \boldsymbol{a}\mathbf{1}^\mathsf{T} + \mathbf{1}\boldsymbol{b}^\mathsf{T} ,$$

then the row points, $\boldsymbol{a}_i$, and the column points, $\boldsymbol{b}_j$, will fall on two straight lines at right angles to each other in the biplot. For a contingency table, the multiplicative relations among frequencies under independence become additive relations in terms of log frequency, and Gabriel et al. (1997) illustrate how biplots of log frequency can be used to explore associations in two-way and three-way tables.

That is, for a two-way table, independence, $A \perp B$, implies that ratios of frequencies should be proportional for any two rows, i, i' and any two columns, j, j'. Equivalently, this means that the log odds ratio for all such sets of four cells should be zero:

$$A \perp B \iff \log \theta_{ii',jj'} = \log \left(\frac{n_{ij} n_{i'j'}}{n_{i'j} n_{ij'}} \right) = 0 .$$

Now, if the log frequencies have been centered by subtracting the grand mean, Gabriel et al. (1997) show that $\log \theta_{ii',jj'}$ is approximated in the biplot (of $\log(n_{ij}) - \overline{\log(n_{ij})}$),

$$\log \theta_{ii',jj'} \approx \boldsymbol{a}_i^\mathsf{T} \boldsymbol{b}_j - \boldsymbol{a}_{i'}^\mathsf{T} \boldsymbol{b}_j - \boldsymbol{a}_i^\mathsf{T} \boldsymbol{b}_{j'} + \boldsymbol{a}_i^\mathsf{T} \boldsymbol{b}_{j'} = (\boldsymbol{a}_i - \boldsymbol{a}_{i'})^\mathsf{T} (\boldsymbol{b}_i - \boldsymbol{b}_{i'}) .$$

Therefore, this biplot criterion for independence in a two-way table is whether $(\boldsymbol{a}_i - \boldsymbol{a}_{i'})^\mathsf{T} (\boldsymbol{b}_i - \boldsymbol{b}_{i'}) \approx 0$ for all pairs of rows, i, i', and all pairs of columns, j, j'. But $(\boldsymbol{a}_i - \boldsymbol{a}_{i'})$ is the vector connecting $\boldsymbol{a}_i$ to $\boldsymbol{a}_{i'}$ and $(\boldsymbol{b}_j - \boldsymbol{b}_{j'})$ is the vector connecting $\boldsymbol{b}_j$ to $\boldsymbol{b}_{j'}$, as shown in Figure 6.15, and the inner product of any two vectors equals zero *iff* they are orthogonal. Hence, this criterion implies that all lines connecting pairs of row points are orthogonal to lines connecting pairs of column points, as illustrated in Figure 6.15.

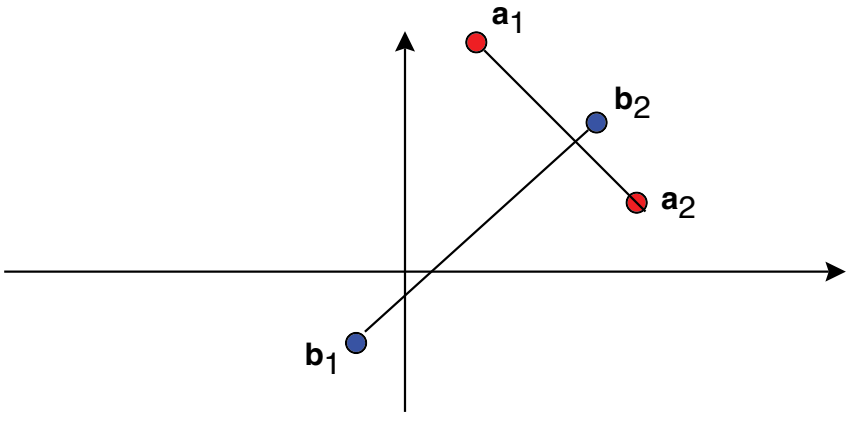


Figure 6.15: Independence implies orthogonal vector differences in a biplot of log frequency. The line joining $\boldsymbol{a}_1$ to $\boldsymbol{a}_2$ represents $(\boldsymbol{a}_1 - \boldsymbol{a}_2)$. This line is perpendicular to the line $(\boldsymbol{b}_1 - \boldsymbol{b}_2)$ under independence.

EXAMPLE 6.11: UK soccer scores

We examined the data on UK soccer scores in Example 5.5 and saw that the number of goals scored by the home and away teams were largely independent (see Figure 5.10). This data set provides a good test of the ability of the biplot to diagnose independence.

```
> data("UKSoccer", package = "vcd")
> dimnames(UKSoccer) <- list(Home = paste0("H", 0:4),
+                            Away = paste0("A", 0:4))
```

Basic biplots in R are provided by `biplot()`, which works mainly with the result calculated by `prcomp()` or `princomp()`. Here, we use `prcomp()` on the log frequencies in the *UKSoccer* table, adding 1, because there is one cell with zero frequency.

```
> soccer.pca <- prcomp(log(UKSoccer + 1), center = TRUE, scale. = FALSE)
```

The result is plotted using a customized plot based on `biplot()`, as shown in Figure 6.16.

```
> biplot(soccer.pca, scale = 0, var.axes = FALSE,
+   col = c("blue", "red"), cex = 1.2, cex.lab = 1.2,
+   xlab = "Dimension 1", ylab = "Dimension 2")
```

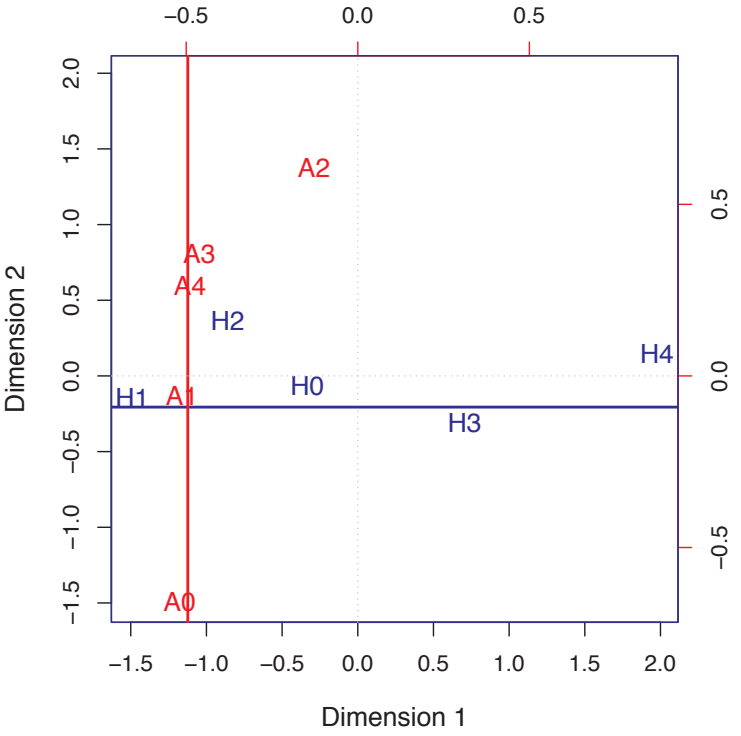


Figure 6.16: Biplot for the biadditive representation of independence for the UK soccer scores. The row and column categories are independent in this plot when they appear as points on approximately orthogonal lines.

To supplement this plot and illustrate the orthogonality of row and column category points under independence, we added horizontal and vertical lines as calculated below, using the results returned by `prcomp()`. The initial version of this plot showed that two points, A2 and H2, did not align with the others, so these were excluded from the calculations.

```
> # get the row and column scores
> rscores <- soccer.pca$x[, 1 : 2]
> cscores <- soccer.pca$rotation[, 1 : 2]
> # means, excluding A2 and H2
> rmean <- colMeans(rscores[-3,])[2]
> cmean <- colMeans(cscores[-3,])[1]
>
> abline(h = rmean, col = "blue", lwd = 2)
> abline(v = cmean, col = "red", lwd = 2)
> abline(h = 0, lty = 3, col = "gray")
> abline(v = 0, lty = 3, col = "gray")
```

You can see that all the A points (except for A2) and all the H points (except for H2) lie along straight lines, and these lines are indeed at right angles, signifying independence. The fact that these straight lines are parallel to the coordinate axes is incidental, and unrelated to the independence interpretation.

△

6.6 Chapter summary

- Correspondence analysis is an exploratory technique, designed to show the row and column categories in a two- (or three-) dimensional space. These graphical displays, and various extensions, provide ways to interpret the patterns of association and visually explore the adequacy of certain loglinear models.
- The scores assigned to the categories of each variable are optimal in several equivalent ways. Among other properties, they maximize the (canonical) correlations between the quantified variables (weighted by cell frequencies), and make the regressions of each variable on the other most nearly linear, for each CA dimension.
- Multi-way tables may be analyzed in several ways. In the "stacking" approach, two or more variables may be combined interactively in the rows and/or columns of an n-way table. Simple CA of the restructured table reveals associations between the row and column categories of the restructured table, but hides associations between the variables combined interactively. Each way of stacking corresponds to a particular loglinear model for the full table.
- Multiple correspondence analysis is a generalization of CA to two or more variables based on representing the data as an indicator matrix, or the Burt matrix. The usual MCA provides an analysis of the joint, bivariate relations between all pairs of variables.
- The biplot is a related technique for visualizing the elements of a data array by points or vectors in a joint display of their row and column categories. A standard CA biplot represents the contributions to lack of independence as the projection of the points for rows (or columns) on vectors for the other categories.
- Another application of the biplot to contingency table data is described, based on analysis of log frequency. This analysis also serves to diagnose patterns of independence and partial independence in two-way and larger tables.

6.7 Lab exercises

Exercise 6.1 The `JobSat` data in vcdExtra gives a 4×4 table recording job satisfaction in relation to income.

(a) Carry out a simple correspondence analysis on this table. How much of the inertia is accounted for by a one-dimensional solution? How much by a two-dimensional solution?
(b) Plot the 2D CA solution. To what extent can you consider the association between job satisfaction and income "explained" by the ordinal nature of these variables?

Exercise 6.2 Refer to Exercise 5.1 in Chapter 5. Carry out a simple correspondence analysis on the 4×5 table `criminal` from the logmult package.

(a) What percentages of the Pearson χ^2 for association are explained by the various dimensions?
(b) Plot the 2D correspondence analysis solution. Describe the pattern of association between year and age.

Exercise 6.3 Refer to Exercise 5.2 for a description of the `AirCrash` data from the vcdExtra package. Carry out a simple correspondence analysis on the 5×5 table of `Phase` of the flight and `Cause` of the crash.

(a) What percentages of the Pearson χ^2 for association are explained by the various dimensions?
(b) Plot the 2D correspondence analysis solution. Describe the pattern of association between phase and cause. How would you interpret the dimensions?
(c) The default plot method uses `map="symmetric"` with points for both rows and columns. Try using `map="symbiplot"` with vectors (`arrows=`) for either rows or columns. (Read `help(plot.ca)` for a description of these options.)

Exercise 6.4 The data set `caith` in MASS gives a classic table tabulating hair color and eye color of people in Caithness, Scotland, originally from Fisher (1940).

(a) Carry out a simple correspondence analysis on this table. How many dimensions seem necessary to account for most of the association in the table?
(b) Plot the 2D solution. The interpretation of the first dimension should be obvious; is there any interpretation for the second dimension?

Exercise 6.5 The same data, plus a similar table for Aberdeen, are given as a three-way table as `HairEyePlace` in vcdExtra.

(a) Carry out a similar correspondence analysis to the last exercise for the data from Aberdeen. Comment on any differences in the placement of the category points.
(b) Analyze the three-way table, stacked to code hair color and place interactively, i.e., for the loglinear model [Hair Place][Eye]. What does this show?

Exercise 6.6 The data set `Gilby` in vcdExtra gives a classic (but now politically incorrect) 6×4 table of English schoolboys classified according to their clothing and their teacher's rating of "dullness" (lack of intelligence).

(a) Compute and plot a correspondence analysis for this data. Write a brief description and interpretation of these results.
(b) Make an analogous mosaic plot of this table. Interpret this in relation to the correspondence analysis plot.

Exercise 6.7 For the mental health data analyzed in Example 6.2, construct a shaded sieve diagram and mosaic plot. Compare these with the correspondence analysis plot shown in Figure 6.2. What features of the data and the association between SES and mental health status are shown in each?

Exercise 6.8 Simulated data are often useful to help understand the connections between data, analysis methods, and associated graphic displays. Section 6.3.1 illustrated interactive coding in R, using a simulated 4-way table of counts of pets, classified by age, color, and sex, but with no associations because the counts had a constant Poisson mean, $\lambda = 15$.

(a) Re-do this example, but in the call to `rpois()`, specify a non-negative vector of Poisson means to create some associations among the table factors.
(b) Use CA methods to determine if and how the structure you created in the data appears in the results.

Exercise 6.9 The *TV* data was analyzed using CA in Example 6.4, ignoring the variable `Time`. Carry out analyses of the 3-way table, reducing the number of levels of `Time` to three hourly intervals as shown below.

```
> data("TV", package="vcdExtra")
> # reduce number of levels of Time
> TV.df <- as.data.frame.table(TV)
> levels(TV.df$Time) <- rep(c("8", "9", "10"), c(4, 4, 3))
> TV3 <- xtabs(Freq ~ Day + Time + Network, TV.df)
> structable(Day ~ Network + Time, TV3)

             Day Monday Tuesday Wednesday Thursday Friday
Network Time
ABC     8             536     861       744      735   1119
        9            1401    1205      1022      682    907
        10            910    1044       668      349    711
CBS     8            1167     646       550      680    509
        9             967     959       409      385    544
        10            789     798       324      270    426
NBC     8             858    1090       512     1927    823
        9             946     890       831     1858    590
        10            825     588       869     2101    585
```

(a) Use the stacking approach (Section 6.3) to perform a CA of the table with `Network` and `Time` coded interactively. You can create this using the `as.matrix()` method for a "structable" object.

```
> TV3S <- as.matrix(structable(Day ~ Network + Time, TV3), sep=":")
```

(b) What loglinear model is analyzed by this approach?
(c) Plot the 2D solution. Compare this to the CA plot of the two-way table in Figure 6.4.
(d) Carry out an MCA analysis using `mjca()` of the three-way table `TV3`. Plot the 2D solution, and compare this with both the CA plot and the solution for the stacked three-way table.

Exercise 6.10 Refer to the MCA analysis of the *PreSex* data in Example 6.8. Use the stacking approach to analyze the stacked table with the combinations of premarital and extramarital sex in the rows and the combinations of gender and marital status in the columns. As suggested in the exercise above, you can use `as.matrix(structable())` to create the stacked table.

(a) What loglinear model is analyzed by this approach? Which associations are included and which are excluded in this analysis?
(b) Plot the 2D CA solution for this analysis. You might want to draw lines connecting some of the row points or column points to aid in interpretation.
(c) How does this analysis differ from the MCA analysis shown in Figure 6.10?

Exercise 6.11 Refer to Exercise 5.10 for a description of the *Vietnam* data set in vcdExtra.

(a) Using the stacking approach, carry out a correspondence analysis corresponding to the loglinear model [R][YS], which asserts that the response is independent of the combinations of year an sex.
(b) Construct an informative 2D plot of the solution, and interpret in terms of how the response varies with year for males and females.
(c) Use `mjca()` to carry out an MCA on the three-way table. Make a useful plot of the solution and interpret in terms of the relationship of the response to year and sex.

Exercise 6.12 Refer to Exercise 5.9 for a description of the *Accident* data set in vcdExtra. The data set is in the form of a frequency data frame, so first convert to table form.

```
> accident.tab <- xtabs(Freq ~ age + result + mode + gender, data=Accident)
```

(a) Use `mjca()` to carry out an MCA on the four-way table `accident.tab`.
(b) Construct an informative 2D plot of the solution, and interpret in terms of how the variable `result` varies in relation to the other factors.

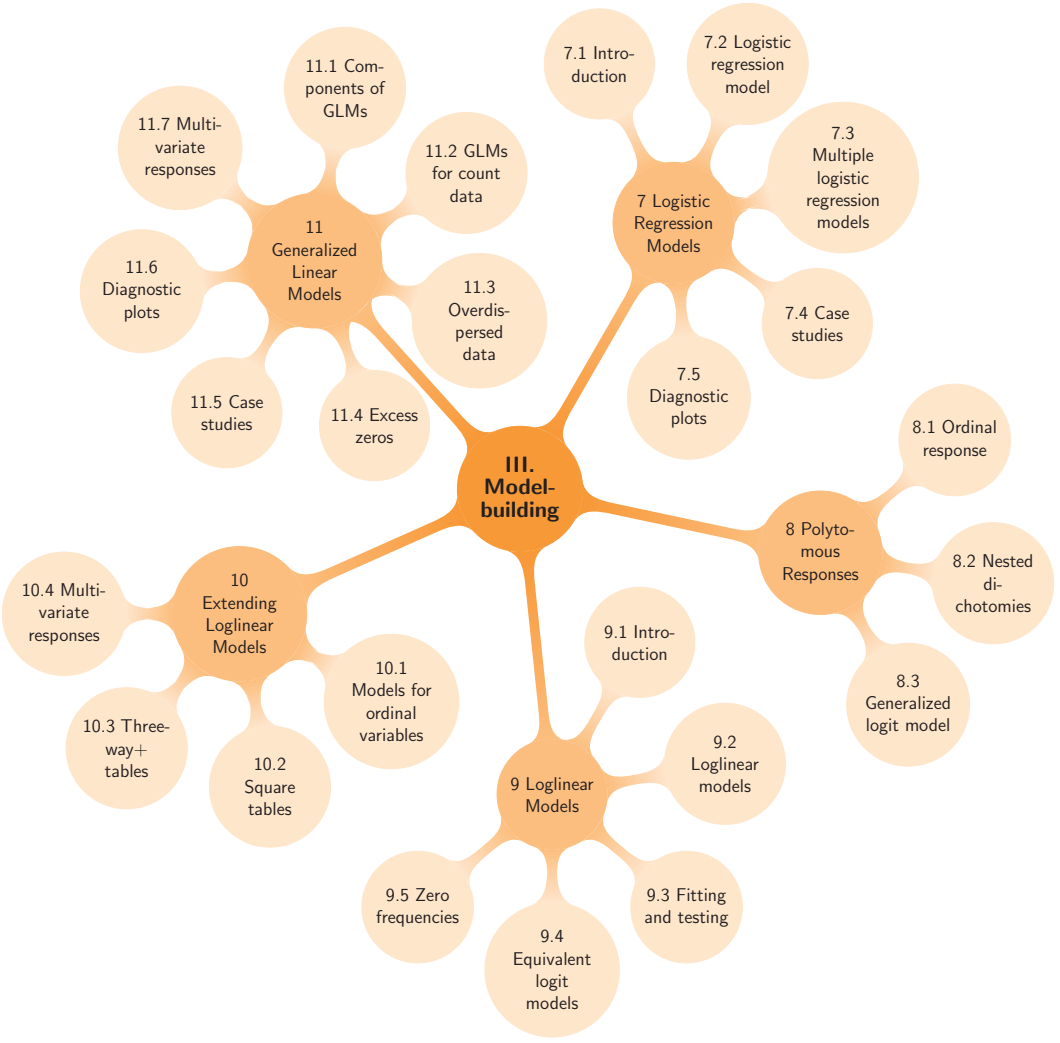
Exercise 6.13 The *UCBAdmissions* data was featured in numerous examples in Chapter 4 (e.g., Example 4.11, Example 4.15) and Chapter 5 (e.g., Example 5.14, Example 5.18).

(a) Use `mjca()` to carry out an MCA on the three-way table `UCBAdmissions`.
(b) Plot the 2D MCA solution in a style similar to that shown in Figure 6.10 and Figure 6.11
(c) Interpret the plot. Is there some interpretation for the first dimension? What does the plot show about the relation of admission to the other factors?

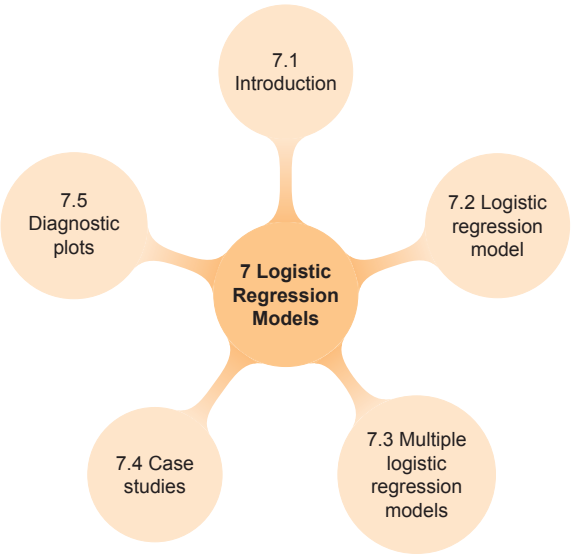
This page intentionally left blank

Part III

Model-Building Methods



This page intentionally left blank



7

Logistic Regression Models

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

7.1 Introduction

> All models are wrong, but some are useful.
>
> George E. P. Box, (Box and Draper, 1987, p. 424)

Chapters 4–6 have been concerned primarily with simple exploratory methods for studying the relations among categorical variables and with testing hypotheses about their associations through non-parametric tests and with overall goodness-of-fit statistics.

This chapter begins our study of model-based methods for the analysis of discrete data. These models differ from those we have examined earlier primarily in that they consider *explicitly* an assumed probability distribution for the observations, and make clear distinctions between the systematic component, which is explained by the model, and the random component, which is not. More importantly, the model-based approach allows a compact summary of categorical data in terms of a (hopefully) small number of parameters accompanied by measures of uncertainty (standard errors), and the ability to estimate predicted values over the range of explanatory variables.

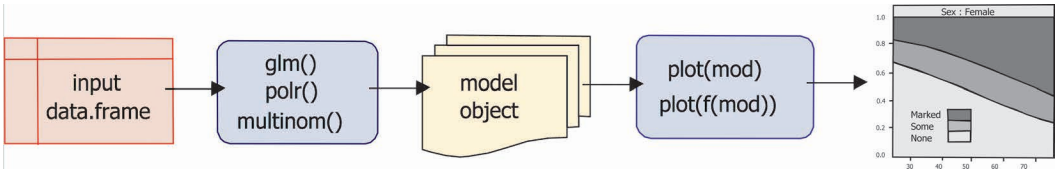


Figure 7.1: Overview of fitting and graphing for model-based methods in R.

This model-fitting approach has several advantages: (a) Inferences for the model parameters include both hypothesis tests and confidence intervals. (b) The former help us to assess which explanatory variables affect the outcome; the size of the estimated parameters and the widths of their confidence intervals help us to assess the strength and importance of these effects. (c) There are a variety of methods for model selection, designed to help determine a favorable trade-off between goodness-of-fit and parsimony. (d) Finally, the predicted values obtained from the model effectively smooth the discrete responses, allow predictions for unobserved values of the explanatory variables, and provide important means to interpret the fitted relationship graphically.

Figure 7.1 provides a visual overview of the steps for fitting and graphing with model-based methods in R. (a) A modeling function such as `glm()` is applied to an input data frame. The result is a ***model object*** containing all the information from the fitting process. (b) As is standard in R, `print()` and `summary()` methods give, respectively, basic and detailed printed output. (c) Many modeling functions have `plot()` methods that produce different types of summary and diagnostic plots. (d) For visualizing the fitted model, most model methods provide a `predict()` method that can be used to plot the fitted values from the model over the ranges of the predictors. Such plots can be customized by the addition of points (showing the observations), lines, confidence bands, and so forth.

In this chapter we consider models for a ***binary response***, such as "success" or "failure," or the number of "successes" in a fixed number of "trials," where we might reasonably assume a binomial distribution for the random component. As we will see in Chapter 8, these methods extend readily to a ***polytomous response*** with more than two outcome categories, such as improvement in therapy, with categories "none," "some," and "marked."

These models can be seen as simple extensions of familiar ANOVA and regression models for quantitative data. They are also important special cases of a more general approach, the ***generalized linear model*** that subsumes a wide variety of families of techniques within a single, unified framework. However, rather than starting at the top with the fully general version, this chapter details the important special cases of models for discrete outcomes, beginning with binary responses.

This chapter proceeds as follows: in Section 7.2 we introduce the simple logistic regression model for a binary response and a single quantitative predictor. This model extends directly to models for grouped, binomial data (Section 7.2.4) and to models with any number of regressors (Section 7.3), which can be quantitative, discrete factors, and more general forms.

For interpreting and understanding the results of a fitted model, we emphasize plotting predicted probabilities and predicted log odds in various ways, for which effect plots (Section 7.3.3) are particularly useful for complex models.

Section 7.4 presents several case studies to highlight issues of data analysis, model building, and visualization in the context of constructing and interpreting multiple logistic regression models. These focus on the combination of exploratory plots to see the data, modeling steps, and graphs to interpret a given model. Individual observations sometimes exert great influence on a fitted model. Some measures of influence and diagnostic plots are illustrated in Section 7.5.

7.2 The logistic regression model

The logistic regression model describes the relationship between a discrete outcome variable, the "response," and a set of explanatory variables. The response variable is often ***dichotomous***, although extensions to the model permit multi-category, ***polytomous*** outcomes, discussed in Chapter 8. The explanatory variables may be continuous or (with factor variables) discrete.

For a binary response, Y, and a continuous explanatory variable, X, we may be interested in modeling the probability of a successful outcome, which we denote $\pi(x) \equiv \Pr(Y = 1 \,|\, X = x)$. That is, at a given value $X = x$, you can imagine that there is a binomial distribution of the responses, $\text{Bin}(\pi(x), n_x)$.

The simplest naive model, called the ***linear probability model***, supposes that this probability, $\pi(x)$, varies linearly with the value of x,

$$E(Y \,|\, x) = \pi(x) = \alpha + \beta x \,, \tag{7.1}$$

where the notation $E(Y \,|\, x)$ indicates that the probability $\pi(x)$ represents the population conditional average of the 1s and 0s for all observations with a fixed value of x. For binary observations, this is simply the proportion of 1s.

Figure 7.2 illustrates the basic setup for modeling a binary outcome using the `Arthritis` data, and described more fully in Example 7.1–Example 7.3: The "Better" response represents a positive effect of some arthritis medicament, given age. The 0/1 observations are shown as (jittered) points. The predicted values under the linear probability model (Eqn. (7.1)) are shown as the red lines in both panels. As you can see, this model cannot be right, because it predicts a probability less than 0 for small values of Age, and would also predict probabilities greater than 1 for larger values of Age.

The linear probability model is also wrong because it assumes that the distribution of residuals, $Y_i - \hat{\pi}(x_i)$, is normal, with mean 0 and constant variance. However, because Y is dichotomous, the residuals are also dichotomous, and have variance $\pi(x_i)(1 - \pi(x_i))$, which is maximal for $\pi = 0.5$ and decreases as π goes toward 0 or 1.

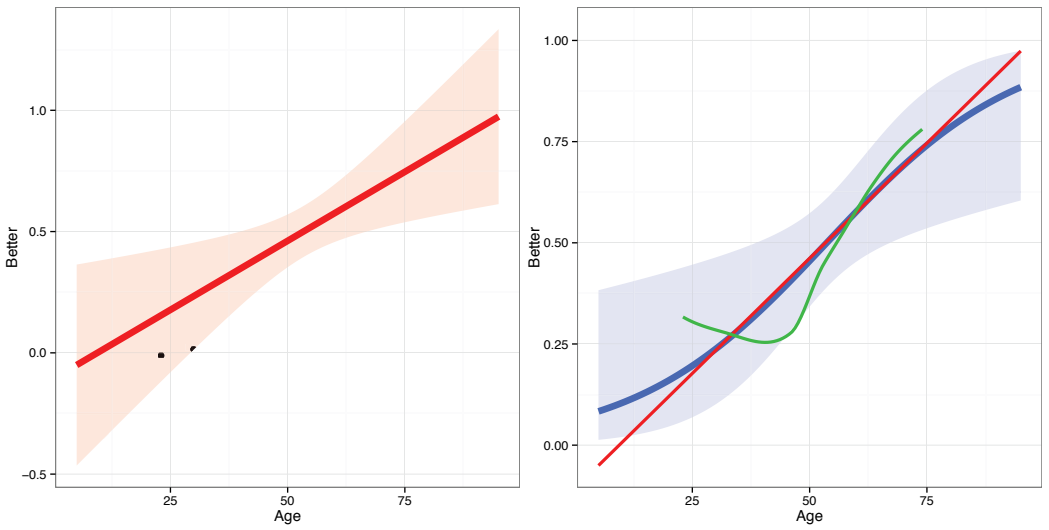


Figure 7.2: Arthritis treatment data, for the relationship of the binary response "Better" to Age, shown as jittered points. The left panel shows the predicted values and 95% confidence envelope under the linear probability model. The right panel shows the fitted logistic regression, together with the simple linear regression (red) and a non-parametric (loess) smoothed curve (green).

One way around the difficulty of needing to constrain the predicted values to the interval [0, 1] is to re-specify the model so that a *transformation* of π has a linear relation to x, and that transformation keeps $\hat{\pi}$ between 0 and 1 for all x. This idea of modeling a transformation of the response that has desired statistical properties is one of the fundamental ones that led to the development of ***generalized linear model***s, which we treat more fully later in Chapter 11.

A particularly convenient choice of the transformation gives the ***linear logistic regression model*** (or ***linear logit model***[1]), which posits a linear relation between the ***log odds*** (or ***logit***) of this probability and x,

$$\text{logit}[\pi(x)] \equiv \log\left(\frac{\pi(x)}{1-\pi(x)}\right) = \alpha + \beta x \,. \tag{7.2}$$

When $\beta > 0$, $\pi(x)$ and the log odds increase as X increases; when $\beta < 0$ they decrease with X.

This model can also be expressed as a model for the probabilities $\pi(x)$ in terms of the *inverse* of the logit transformation used in Eqn. (7.2),

$$\pi(x) = \text{logit}^{-1}[\pi(x)] = \frac{1}{1+\exp[-(\alpha+\beta x)]} \,. \tag{7.3}$$

This transformation uses the cumulative distribution function of the logistic distribution, $\Lambda(p) = \frac{1}{1+exp(-p)}$, giving rise to the term *logistic regression*.[2]

From Eqn. (7.2) we see that the odds of a success response can be expressed as

$$\text{odds}(Y=1) \equiv \frac{\pi(x)}{1-\pi(x)} = \exp(\alpha+\beta x) = e^{\alpha}(e^{\beta})^{x} \,, \tag{7.4}$$

which is a multiplicative model for the odds.

So, under the logistic model,

- β is the change in the log odds associated with a unit increase in x. The odds are multiplied by e^{β} for each unit increase in x.
- α is log odds at $x = 0$; e^{α} is the odds of a favorable response at this x-value (which may not have a reasonable interpretation if $X = 0$ is far from the range of the data).

It is easy to explore the relationships among probabilities, odds, and log odds using R, as we show below, using the function `fractions()` in MASS to print the odds corresponding to probability `p` as a fraction.

```
> library(MASS)
> p <- c(.05, .10, .25, .50, .75, .90, .95)
> odds <- p / (1 - p)
> data.frame(p,
+            odds = as.character(fractions(odds)),
+            logit = log(odds))

     p odds   logit
1 0.05 1/19 -2.9444
2 0.10  1/9 -2.1972
3 0.25  1/3 -1.0986
4 0.50    1  0.0000
5 0.75    3  1.0986
6 0.90    9  2.1972
7 0.95   19  2.9444
```

[1] Some writers use the term *logit model* to refer to those using only categorical predictors; we use the terms logistic regression and logit regression interchangeably.

[2] Any other cumulative probability transformation serves the purpose of constraining the probabilities to the interval [0, 1]. The cumulative normal transformation $\pi(x) = \Phi(\alpha + \beta x)$ gives the ***linear probit regression*** model. We don't treat probit models here because: (a) The logistic and probit models give results so similar that it is hard to distinguish them in practice; (b) The logistic model is simpler to interpret as a linear model for the log odds or a multiplicative model for the odds.

Thus, a probability of $\pi = 0.25$ represents an odds of 1 to 3, or 1/3, while a probability of $\pi = 0.75$ represents an odds of 3 to 1, or 3. The logits are symmetric around 0, so $\text{logit}(.25) = -\text{logit}(.75)$.

Another simple way to interpret the parameter β in the logistic regression model is to consider the relationship between the probability $\pi(x)$ and x. From Eqn. (7.3) it can be shown that the fitted curve (the blue line in Figure 7.2) has slope equal to $\beta\pi(1-\pi)$. This has a maximum value of $\beta/4$ when $\pi = \frac{1}{2}$, so taking $\beta/4$ gives a quick estimate of the maximum effect of x on the probability scale.

In Figure 7.2 and other plots later in this chapter we try to show the binary responses (as jittered points or a rug plot) to help you appreciate how the fitted logistic curve arises from their distribution across the range of a predictor. For didactic purposes this can be seen more readily by plotting the conditional distributions of $f(x \mid y), y \in \{0, 1\}$ as a histogram, boxplot, or density plot. The function `logi.hist.plot()` in the popbio (Stubben et al., 2012) package is a nice implementation of this idea (de la Cruz Rot, 2005). The call below produces Figure 7.3, and it is easy to see how increasing age produces a greater probability of a Better response.

```
> with(Arthritis,
+       logi.hist.plot(Age, Improved > "None", type = "hist",
+                      counts = TRUE, ylabel = "Probability (Better)",
+                      xlab = "Age", col.cur = "blue",
+                      col.hist = "lightblue", col.box = "lightblue")
+   )
```

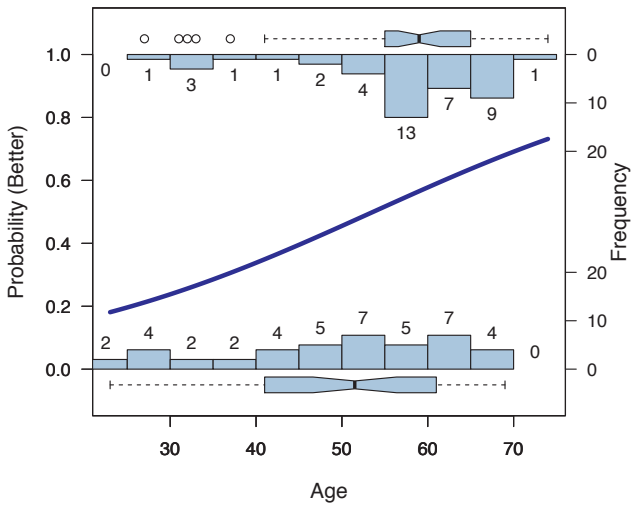


Figure 7.3: Plot of the Arthritis treatment data, showing the conditional distributions of the 0/1 observations of the Better response by histograms and boxplots.

7.2.1 Fitting a logistic regression model

Logistic regression models are the special case of generalized linear models fit in R using `glm()` for a binary response using `family=binomial`. We first illustrate how simple models can be fit and interpreted.

EXAMPLE 7.1: Arthritis treatment

In Chapter 4 we examined the data on treatment for rheumatoid arthritis in relation to two

categorical predictors, sex of patient and treatment. In addition, the *Arthritis* data gives the age of each patient in this study, and we focus here on the relationship between Age and the outcome, Improved. This response variable has three categories (none, some, or marked improvement), but for now we consider whether the patient showed any improvement at all, defining the event Better to be some or marked improvement.

```
> data("Arthritis", package = "vcd")
> Arthritis$Better <- as.numeric(Arthritis$Improved > "None")
```

The logistic regression model is fit using glm() as shown below, specifying family=binomial for a binary response.

```
> arth.logistic <- glm(Better ~ Age, data = Arthritis, family = binomial)
```

As usual for R modeling functions, the print() method for "glm" objects gives brief printed output, while the summary() method is more verbose, and includes standard errors and hypothesis tests for the model coefficients. To save some space, it is convenient to use the generic function coeftest() from the lmtest (Hothorn et al., 2014) package. Then, we can use this instead of the more detailed summary():

```
> library(lmtest)
> coeftest(arth.logistic)


z test of coefficients:

            Estimate Std. Error z value Pr(>|z|)
(Intercept)  -2.6421     1.0732   -2.46    0.014 *
Age           0.0492     0.0194    2.54    0.011 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

In the output above, the parameter estimates are $\alpha = -2.642$, and $\beta = 0.0492$. So, the estimated odds of a better response are multiplied by $e^{\beta} = \exp(0.0492) = 1.05$ for each one-year increase in age. Equivalently, you can think of this as a 5% increase per year (using $100(e^{\beta} - 1)$ to convert). Over 10 years, the odds are multiplied by $\exp(10 \times 0.0492) = 1.64$, a 64% increase, a substantial effect in the range for these data. You can do these calculations in R using the coef() method for the "glm" object.

```
> exp(coef(arth.logistic))

(Intercept)         Age
   0.071214    1.050482

> exp(10 * coef(arth.logistic)["Age"])

   Age
1.6364
```

For comparison with the logistic model, we could fit the linear probability model Eqn. (7.1) using either lm() or glm() with the default family=gaussian argument.

```
> arth.lm <- glm(Better ~ Age, data = Arthritis)
> coef(arth.lm)

(Intercept)         Age
  -0.107170    0.011379
```

The coefficient for age can be interpreted to indicate that the probability of a better response increases by 0.011 for each one-year increase in age. You can compare this with the $\beta/4$ rule of thumb, which gives 0.0492/4 = 0.0123. Even though the linear probability model is inappropriate theoretically, you can see in Figure 7.2 (the black line) that it gives similar predicted probabilities to those of the logistic model between age 25–75, where most of the data points are located.

△

7.2.2 Model tests for simple logistic regression

There are two main types of hypothesis tests one might want to perform for a logistic regression model. We postpone general discussion of this topic until Section 7.3, but introduce the main ideas here using the analysis of the `Arthritis` data.

- The most basic test answers the question, "How much better is the fitted model, $\text{logit}(\pi) = \alpha + \beta x$, than the null model $\text{logit}(\pi) = \alpha$, which includes only the regression intercept?" One answer to this question is given by the (Wald) test of the coefficient for age, testing the hypothesis $H_0 : \beta = 0$ that appeared in the output from `summary(arth.logistic)` shown above. The more direct test compares the deviance[3] of the fitted model to the deviance of the null model, and can be obtained using the `anova()` function:

```
> anova(arth.logistic, test = "Chisq")

Analysis of Deviance Table

Model: binomial, link: logit

Response: Better

Terms added sequentially (first to last)


     Df Deviance Resid. Df Resid. Dev Pr(>Chi)
NULL                    83        116
Age   1     7.29        82        109    0.007 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

- A second question is, "How bad is this model, compared to a model (the ***saturated model***) that fits the data perfectly?" This is a test of the size of the residual deviance, which is given by the function `LRstats()` in vcdExtra.

```
> library(vcdExtra)
> LRstats(arth.logistic)

Likelihood summary table:
              AIC BIC LR Chisq Df Pr(>Chisq)
arth.logistic 113 118      109 82      0.024 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The summary of these tests is that linear logistic model Eqn. (7.2) fits significantly better than the null model, but that model also shows significant lack of fit.

[3]The deviance is basically defined as -2 times the log-likelihood ratio of some reduced model to the full model. Two nested models can thus be compared by computing the difference of the corresponding deviances. If the larger model has k more parameters than the reduced one, this difference follows a chi-squared distribution with k degrees of freedom.

7.2.3 Plotting a binary response

It is often difficult to understand how a binary response can give rise to a smooth, continuous relation between the predicted response, usually the probability of an event, and a continuous explanatory variable. Beyond this, plots of the data together with fitted models help you to interpret what these models imply.

We illustrate two approaches below using the `Arthritis` data shown in Figure 7.2, first using R base graphics, and then with the ggplot2 package that makes such graphs somewhat easier to do.

That plot, which was designed for didactic purposes, has the following features:

- It shows the *data*, that is, the 0/1 observations of the `Better` response in relation to age. To do this effectively and avoid over-plotting, the binary responses are jittered.
- It plots the predicted (fitted) logistic regression relationship on the scale of probability, together with a 95% confidence band.
- It also plots the predicted probabilities from the linear probability model.
- A smoothed, non-parametric regression curve for the binary observations is also added to the plot to give some indication of possible nonlinearity in the relationship of Better to age.

EXAMPLE 7.2: Arthritis treatment — Plotting logistic regression with base graphics

Here we explain how plots similar to Figure 7.2 can be constructed using R base graphics. We describe the steps needed to calculate predicted values and confidence bands and how to add these to a basic plot. These ideas are the basis for the higher-level and more convenient plotting methods illustrated later in this chapter (Section 7.3.2) The steps detailed below give the plot shown in Figure 7.4.

First, we set up the basic plot of the jittered values of `Better` vs. `Age`, setting `xlim` to a larger range than that in the data, only to emphasize where the logistic and linear probability models diverge.

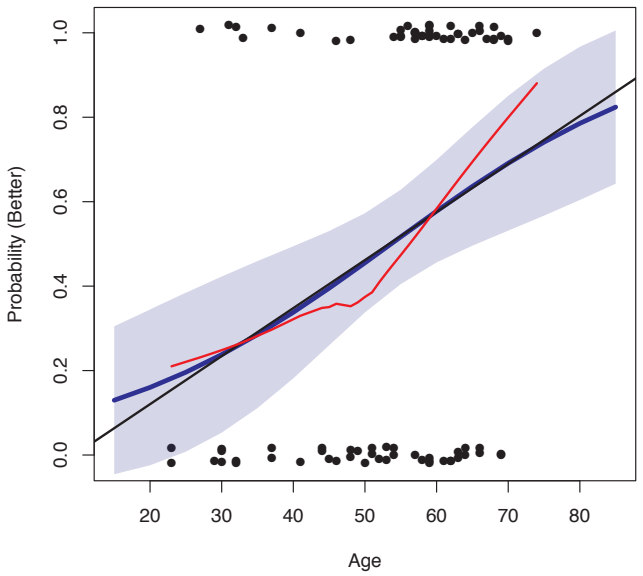


Figure 7.4: A version of plot of the Arthritis treatment data (Figure 7.2) produced with R base graphics, showing logistic, linear regression and lowess fits.

```
> plot(jitter(Better, .1) ~ Age, data = Arthritis,
+      xlim = c(15, 85), pch = 16,
+      ylab="Probability (Better)")
```

The fitted logistic curve can be obtained using the `predict()` method for the "glm" object `arth.logistic`. For this example, we wanted to get fitted values for the range of Age from 15–85, which is specified in the `newdata` argument.[4] The argument `type="response"` gives fitted values of the probabilities. (The default, `type="link"` would give predicted logits.) Standard errors of the fitted values are not calculated by default, so we set `se.fit=TRUE`.

```
> xvalues <- seq(15, 85, 5)
> pred.logistic <- predict(arth.logistic,
+                          newdata = data.frame(Age = xvalues),
+                          type = "response", se.fit = TRUE)
```

When `se.fit=TRUE`, the `predict()` function returns its result in a list, with components `fit` for the fitted values and `se.fit` for the standard errors. From these, we can calculate 95% pointwise prediction intervals using the standard normal approximation.

```
> upper <- pred.logistic$fit + 1.96 * pred.logistic$se.fit
> lower <- pred.logistic$fit - 1.96 * pred.logistic$se.fit
```

We can then plot the confidence band using `polygon()` and the fitted logistic curve using `lines`. A graphics trick is to use a transparent color for the confidence band using `rgb(r, g, b, alpha)`, where `alpha` is the transparency value.

```
> polygon(c(xvalues, rev(xvalues)),
+         c(upper, rev(lower)),
+         col = rgb(0, 0, 1, .2), border = NA)
> lines(xvalues, pred.logistic$fit, lwd=4 , col="blue")
```

This method, using `predict()` for calculations and `polygon()` and `lines()` for plotting can be used to display the predicted relationships and confidence bands under other models. Here, we simply used `abline()` to plot the fitted line for the linear probability model `arth.lm` and `lowess()` to calculate a smoothed, non-parametric curve.

```
> abline(arth.lm, lwd = 2)
> lines(lowess(Arthritis$Age, Arthritis$Better, f = .9),
+       col = "red", lwd = 2)
```

△

EXAMPLE 7.3: Arthritis treatment — Plotting logistic regression with ggplot2

Model-based plots such as Figure 7.2 are relatively more straightforward to produce using ggplot2. The basic steps here are to:

- set up the plot frame with `ggplot()` using Age and Better as (x, y) coordinates;
- use `geom_point()` to plot the observations, whose positions are jittered with `position_jitter()`;
- use `stat_smooth()` with `method = "glm"` and `family = binomial` to plot the predicted probability curve and confidence band. By default, `stat_smooth()` calculates and plots 95% confidence bands on the response (probability) scale.

[4] Omitting the `newdata` argument would give predicted values using the linear predictors in the data used for the fitted model. Some care needs to be taken if the predictor(s) contain missing values.

```
> library(ggplot2)
> # basic logistic regression plot
> gg <- ggplot(Arthritis, aes(x = Age, y = Better)) +
+   xlim(5, 95) +
+   geom_point(position = position_jitter(height = 0.02, width = 0)) +
+   stat_smooth(method = "glm", family = binomial,
+               alpha = 0.1, fill = "blue", size = 2.5, fullrange = TRUE)
```

Finally, we can add other smoothers to the plot, literally by using + to add these to the "ggplot" object.

```
> # add linear model and loess smoothers
> gg <- gg + stat_smooth(method = "lm", se = FALSE,
+                        size = 1.2, color = "black", fullrange = TRUE)
> gg <- gg + stat_smooth(method = "loess", se = FALSE,
+                        span = 0.95, colour = "red", size = 1.2)
> gg  # show the plot
```

△

7.2.4 Grouped binomial data

A related case occurs with grouped data, where rather than binary observations, $y_i \in \{0, 1\}$ in case form, the data is given in what is called ***events/trials form*** that records the number of successes, y_i that occurred in n_i trials associated with each setting of the explanatory variable(s) x_i.[5] Case form, with binary observations, is the special case where $n_i = 1$.

Data in events/trials form often arises from contingency table data with a binary response. For example, in the *UCBAdmissions* data, the response variable `Admit` with levels `"Admitted"`, `"Rejected"` could be treated in this way using the number of applicants as the number of trials.

As before, we can consider y_i/n_i to estimate the probability of success, π_i, and the distribution of Y to be binomial, $\text{Bin}(\pi_i, n_i)$ at each x_i.

In practical applications, there are two main differences between the cases of ungrouped, case form data and grouped, event/trials form.

- In fitting models using `glm()`, the model formula, `response ~ terms`, can be given using a `response` consisting of a two-column matrix, whose columns contain the numbers of successes y_i and failures $n_i - y_i$. Alternatively, the `response` can be given as the proportion of successes, y_i/n_i, but then it is necessary to specify the number of trials as a `weight`.
- In plotting the fitted model on the scale of probability, you usually have to explicitly plot the fraction of successes, y_i/n_i.

EXAMPLE 7.4: Space shuttle disaster

In Example 1.2 and Example 1.10 we described the background behind the post-mortem examination of the evidence relating to the disastrous launch of the space shuttle *Challenger* on January 28, 1986. Here we consider a simple, but proper analysis of the data available at the time of launch. We also use this example to illustrate some details of the fitting and plotting of grouped binomial data. As well, we describe some of the possibilities for dealing with missing data.

The data set *SpaceShuttle* in vcd contains data on the failures of the O-rings in 24 NASA launches preceding the launch of *Challenger*, as given by Dalal et al. (1989) and Tufte (1997), also analyzed by Lavine (1991).

[5] Alternatively, the data may record the number of successes, y_i, and number of failures, $n_i - y_i$.

Each launch used two booster rockets with a total of six O-rings, and the data set records as `nFailures` the number of these that were considered damaged after the rockets were recovered at sea. In one launch (flight # 4), the rocket was lost at sea, so the relevant response variables are missing.

In this example, we focus on the variable `nFailures` as a binomial with $n_i = 6$ trials. The missing data for flight 4 can be handled in several ways in the call to `glm()`

```
> data("SpaceShuttle", package = "vcd")
> shuttle.mod <- glm(cbind(nFailures, 6 - nFailures) ~ Temperature,
+           data = SpaceShuttle, na.action = na.exclude,
+           family = binomial)
```

Alternatively, we can add an explicit `trials` variable, represent the response as the proportion `nFailures/trials`, and use `weight = trials` to indicate the total number of observations.

```
> SpaceShuttle$trials <- 6
> shuttle.modw <- glm(nFailures / trials ~ Temperature, weight = trials,
+           data = SpaceShuttle, na.action = na.exclude,
+           family = binomial)
```

These two approaches give identical results for all practical purposes:

```
> all.equal(coef(shuttle.mod), coef(shuttle.modw))

[1] TRUE
```

As before, we can test whether temperature significantly improves prediction of failure probability using `anova()`:

```
> # testing, vs. null model
> anova(shuttle.mod, test = "Chisq")

Analysis of Deviance Table

Model: binomial, link: logit

Response: cbind(nFailures, 6 - nFailures)

Terms added sequentially (first to last)


            Df Deviance Resid. Df Resid. Dev Pr(>Chi)
NULL                           22       24.2
Temperature  1     6.14        21       18.1    0.013 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The code below gives a ggplot2 version in Figure 7.5 of the plot we showed earlier in Example 1.2 (Figure 1.2). The relevant details here are:

- We specify `y = nFailures / trials` to calculate the failure probabilities.
- Points are jittered in the call to `geom_point()` to prevent overplotting.
- In the call to `geom_smooth()`, we need to use `weight = trials`, just as in the call to `glm()` above.
- `fullrange = TRUE` makes the fitted regression curve and confidence band extend across the entire plot.

```
> library(ggplot2)
> ggplot(SpaceShuttle, aes(x = Temperature, y = nFailures / trials)) +
+   xlim(30, 81) +
+   xlab("Temperature (F)") +
+   ylab("O-Ring Failure Probability") +
+   geom_point(position=position_jitter(width = 0, height = 0.01),
+              aes(size = 2)) +
+   theme(legend.position = "none") +
+   geom_smooth(method = "glm", family = binomial, fill = "blue",
+               aes(weight = trials), fullrange = TRUE, alpha = 0.2,
+               size = 2)
```

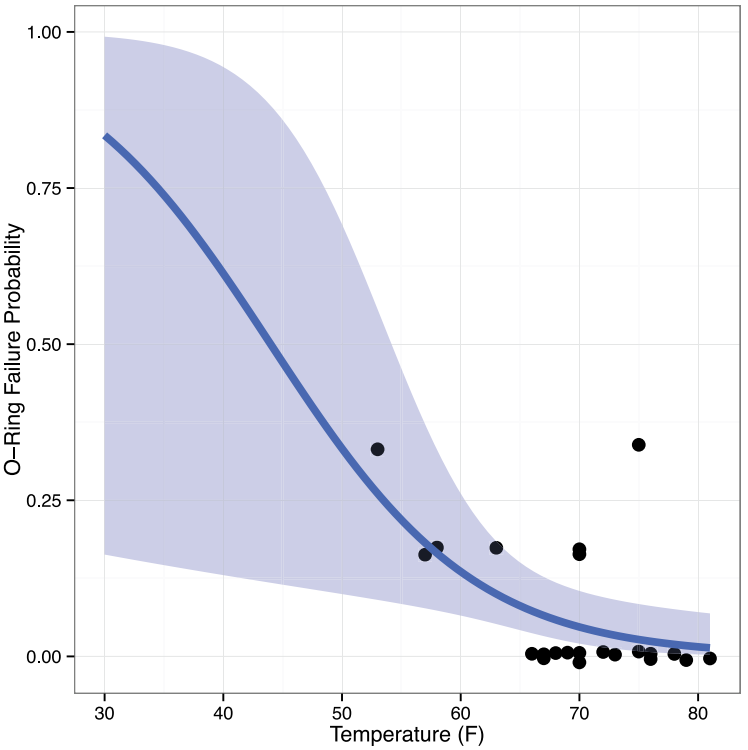


Figure 7.5: Space shuttle data, with fitted logistic regression model.

△

7.3 Multiple logistic regression models

As is the case in classical regression, generalizing the simple logistic regression to an arbitrary number of explanatory variables is quite straightforward. We let $\boldsymbol{x}_i = (x_{i1}, x_{i2}, \ldots, x_{ip})$ denote the vector of p explanatory variables for case or cluster i. Then the general logistic regression model can be expressed as

$$\begin{aligned} \text{logit}(\pi_i) \equiv \log \frac{\pi_i}{1-\pi_i} &= \alpha + \boldsymbol{x}_i^{\mathsf{T}} \boldsymbol{\beta} \\ &= \alpha + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip} \,. \end{aligned} \tag{7.5}$$

Equivalently, we can represent this model in terms of probabilities as the logistic transformation of the ***linear predictor***, $\eta_i = \alpha + \boldsymbol{x}_i^\mathsf{T}\boldsymbol{\beta}$,

$$\begin{aligned} \pi_i = \Lambda(\eta_i) &= \Lambda(\alpha + \boldsymbol{x}_i^\mathsf{T}\boldsymbol{\beta}) \\ &= \frac{1}{1 + \exp(\alpha + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip})} \,. \end{aligned} \tag{7.6}$$

The xs can include any of the following sorts of regressors, as in the general linear model:

- **quantitative** variables (e.g., age, income)
- **polynomial** powers of quantitative variables (e.g., age, age^2, age^3)
- **transformations** of quantitative variables (e.g., log salary)
- factors, represented as **dummy** variables for qualitative predictors (e.g., P_1, P_2, P_3 for four political party affiliations)
- **interaction** terms (e.g., sex $\times$ age, or age $\times$ income)

EXAMPLE 7.5: Arthritis treatment

We continue with the analysis of the `Arthritis` data, fitting a model containing the main effects of `Age`, `Sex`, and `Treatment`, with `Better` as the response. This model has the form

$$\text{logit}(\pi_i) = \alpha + \beta_1 x_{i1} + \beta_2 x_{i2} + \beta_3 x_{i3}$$

where x_1 is `Age` and x_2 and x_3 are the factors representing `Sex` and `Treatment`, respectively.

Using the default (0/1) dummy coding that R uses ("treatment" contrasts against the lowest factor level),[6] they are defined as:

$$x_2 = \begin{cases} 0 & \text{if Female} \\ 1 & \text{if Male} \end{cases} \qquad x_3 = \begin{cases} 0 & \text{if Placebo} \\ 1 & \text{if Treatment} \end{cases}$$

In this model,

- α doesn't have a sensible interpretation here, but formally it would be the log odds of improvement for a person at age $x_1 = 0$ in the baseline or reference group, with $x_2 = 0$ and $x_3 = 0$—that is, females receiving the placebo. To make the intercept interpretable, we will fit the model centering age near the mean, by using $x_1 - 50$ as the first regressor.
- β_1 is the increment in log odds of improvement for each one-year increase in age.
- β_2 is the increment in log odds for male as compared to female. Therefore, e^{β_2} gives the odds of improvement for males relative to females.
- β_3 is the increment in log odds for being in the treated group. e^{β_3} gives the odds of improvement for the active treatment group relative to placebo.

We fit the model as follows. In `glm()` model formulas, "–" has a special meaning, so we use the identity function, `I(Age-50)` to center age.

```
> arth.logistic2 <- glm(Better ~ I(Age-50) + Sex + Treatment,
+                       data = Arthritis,
+                       family = binomial)
```

[6] For factor variables with the default treatment contrasts, you can change the reference level using `relevel()`. In this example, you could make male the baseline category using `Arthritis$Sex <- relevel(Arthritis$Sex, ref = "Male")`.

The parameters defined here are *incremental effects*. The intercept corresponds to a baseline group (50-year-old females given the placebo); the other parameters are incremental effects for the other groups compared to the baseline group. Thus, when α, β_1, β_2, and β_3 have been estimated, the fitted logits and predicted odds at `Age==50` are:

Sex	Treatment	Logit	Odds Improved
Female	Placebo	α	e^{α}
Female	Treated	$\alpha + \beta_3$	$e^{\alpha+\beta_3}$
Male	Placebo	$\alpha + \beta_2$	$e^{\alpha+\beta_2}$
Male	Treated	$\alpha + \beta_2 + \beta_3$	$e^{\alpha+\beta_2+\beta_3}$

We first focus on the interpretation of the coefficients estimated for this model shown below.

```
> coeftest(arth.logistic2)


z test of coefficients:

                  Estimate Std. Error z value Pr(>|z|)
(Intercept)        -0.5781     0.3674   -1.57    0.116
I(Age - 50)         0.0487     0.0207    2.36    0.018 *
SexMale            -1.4878     0.5948   -2.50    0.012 *
TreatmentTreated    1.7598     0.5365    3.28    0.001 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

To interpret these in terms of odds ratios and also find confidence intervals, just use `exp()` and `confint()`.

```
> exp(cbind(OddsRatio = coef(arth.logistic2),
+           confint(arth.logistic2)))

                  OddsRatio   2.5 %  97.5 %
(Intercept)          0.5609 0.26475  1.1323
I(Age - 50)          1.0500 1.01000  1.0963
SexMale              0.2259 0.06524  0.6891
TreatmentTreated     5.8113 2.11870 17.7266
```

Here,

- $\alpha = -0.578$: At age 50, females given the placebo have an odds of improvement of $\exp(-0.578) = 0.56$.
- $\beta_1 = 0.0487$: Each year of age multiplies the odds of improvement by $\exp(0.0487) = 1.05$, or a 5% increase.
- $\beta_2 = -1.49$: Males are only $\exp(-1.49) = 0.26$ times as likely to show improvement relative to females. (Or, females are $\exp(1.49) = 4.437$ times more likely than males to improve.)
- $\beta_3 = 1.76$: People given the active treatment are $\exp(1.76) = 5.8$ times more likely to show improvement compared to those given the placebo.

As you can see, the interpretation of coefficients in multiple logistic models is straightforward, though a bit cumbersome. This becomes more difficult in larger models, particularly when there are interactions. In these cases, you can understand (and explain) a fitted model more easily through plots of predicted values, either on the scale of response probability or on the logit scale of the linear predictor. We describe these methods in Section 7.3.1–Section 7.3.3 below.

△

7.3.1 Conditional plots

The simplest kind of plots display the data together with a representation of the fitted relationship (predicted values, confidence bands) separately for subsets of the data defined by one or more of the predictors. Such plots can show the predicted values for the response variable on the ordinate against one chosen predictor on the abscissa, and can use multiple curves and multiple panels to represent other predictors.

However, these plots are ***conditional plots***, meaning that the data shown in each panel and used in each fitted curve are limited to the subset of the observations defined by the curve and panel variables. As well, predictors that are not shown in a given plot are effectively ignored (or marginalized), as was the case in Figure 7.2 that showed only the effect of age in the `Arthritis` data.

EXAMPLE 7.6: Arthritis treatment — conditional plots

For the `Arthritis` data, a basic conditional plot of `Better` vs. `Age`, showing the observations as jittered points (with `geom_point()`) and the fitted logistic curves (with `stat_smooth()` using `method="glm"`), can be produced with ggplot2 as shown below, giving Figure 7.6.

```
> library(ggplot2)
> gg <- ggplot(Arthritis, aes(Age, Better, color = Treatment)) +
+   xlim(5, 95) + theme_bw() +
+   geom_point(position = position_jitter(height = 0.02, width = 0)) +
+   stat_smooth(method = "glm", family = binomial, alpha = 0.2,
+               aes(fill = Treatment), size = 2.5, fullrange = TRUE)
> gg    # show the plot
```

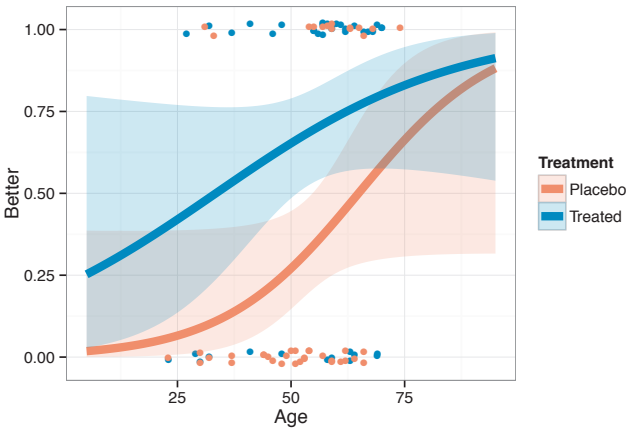


Figure 7.6: Conditional plot of Arthritis data showing separate points and fitted curves stratified by Treatment. A separate fitted curve is shown for the two treatment conditions, ignoring Sex.

In this call to `ggplot()`, specifying `color=Treatment` gives different point and line colors, but also automatically stratifies the fitted curves using the levels of this variable.

With such a plot, it is easy to add further stratifying variables in the data using ***facets*** to produce separate panels (functions `facet_wrap()` or `facet_grid()`, with different options to control the details). The following line further stratifies by `Sex`, producing Figure 7.7.

```
> gg + facet_wrap(~ Sex)
```

However, you can see from this plot how this method breaks down when the sample size is small in some of the groups defined by the stratifying factors. The panel for males shows a paradoxical

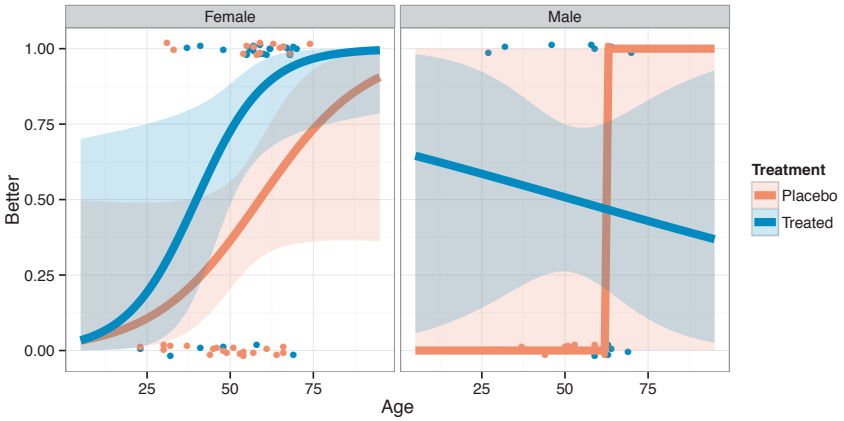


Figure 7.7: Conditional plot of Arthritis data, stratified by Treatment and Sex. The unusual patterns in the panel for Males signals a problem with this data.

negative relation with age for the treated group and a step function for the placebo group. The explanation for this is shown in the two-way frequency table of the sex and treatment combinations:

```
> addmargins(xtabs(~Sex + Treatment, data = Arthritis), 2)

        Treatment
Sex      Placebo Treated Sum
  Female      32      27  59
  Male        11      14  25
```

Less than 1/3 of the sample were males, and of these only 11 were in the placebo group. `glm()` cannot estimate the fitted relationship against `Age` here—the slope coefficient is infinite, and the fitted probabilities are all either 0 or 1.[7]

△

7.3.2 Full-model plots

For a model with two or more explanatory variables, *full-model plots* display the fitted response surface for all predictors together, rather than stratified by conditioning variables. Such plots show the predicted values for the response variable on the ordinate against one chosen predictor on the abscissa, and can use multiple curves and multiple panels to represent other predictors.

The programming steps used to plot a fitted logistic regression with base graphics and ggplot2 in the style of earlier examples (Examples 7.2, 7.2, and 7.4) become more tedious with multiple predictors. The vcd package provides the function `binreg_plot()` designed to plot the predicted response surface for a binary outcome directly from a fitted model object. At the time of writing, this function does not yet handle multiple panels or facets, but separate plots for panel variables can be produced using the `subset` argument as illustrated in the next example.

EXAMPLE 7.7: Arthritis treatment — full-model plots

This example shows how to plot the fitted main effects model using `binreg_plot()`. These plots can be shown either on the logit scale (with `type = "link"`) or the probability scale (`type = "response"`, the default).

[7] This is called ***complete separation***, and occurs whenever the responses have no overlap on the predictor variable(s) used in fitting the logistic regression model.

This plot method is designed to use a numeric predictor (`Age` here) as the horizontal axis, and show separate point symbols and curves for the levels of the combinations of factors (if any). A basic plot on the logit scale (not included here) showing both factors (`Sex`, `Treatment`) can be produced using:

```
> library(vcd)
> binreg_plot(arth.logistic2, type = "link")
```

With two or more factors, such plots are often easier to read when the main factor(s) to be compared appear (`Treatment` here) as lines or curves within a plot, and other factors (`Sex`) are shown in separate panels. Figure 7.8 does this in two plots, using the `subset` argument to select the appropriate data and predicted values for males and females. When this is done, it is important to include the same `xlim` and `ylim` arguments so that the scales of all plots are identical.

```
> binreg_plot(arth.logistic2, type = "link", subset = Sex == "Female",
+             main = "Female", xlim=c(25, 75), ylim = c(-3, 3))
> binreg_plot(arth.logistic2, type = "link", subset = Sex == "Male",
+             main = "Male", xlim=c(25, 75), ylim = c(-3, 3))
```

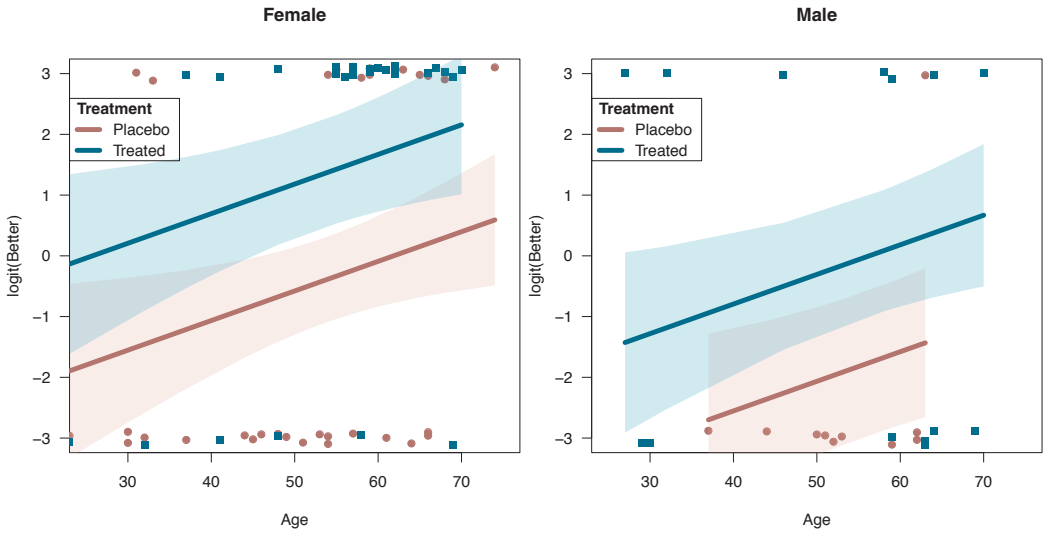


Figure 7.8: Full-model plot of Arthritis data, showing fitted logits by Treatment and Sex.

This plot method has several nice features:

- Plotting on the logit scale shows the additive linear effects of all predictors (parallel lines for the combinations of `Sex` and `Treatment`).
- It provides a visual representation of the information contained in the table of coefficients.
- The choice to display `Treatment` within each panel makes it easier to judge the size of this effect, compared to the effect of `Sex`, which must be judged across the panels.
- It shows the data as points, and the fitted lines and confidence bands are restricted to the range of the data in each. You can easily see the reason for the unusual pattern in the conditional plot for Males shown in Figure 7.7.
- It generalizes directly to any fitted model, because the predicted values are obtained from the model object. For example, you could easily add the interaction term `Age:Sex` and plot the result.

While plots on the logit scale have a simpler form, many people find it easier to think about such relationships in terms of probabilities, as we have done in earlier plots in this chapter. Figure 7.9 shows these plots using the default `type = "response"`.

```
> binreg_plot(arth.logistic2, subset = Sex == "Female",
+             main = "Female", xlim = c(25, 75))
> binreg_plot(arth.logistic2, subset = Sex == "Male",
+             main = "Male", xlim = c(25, 75))
```

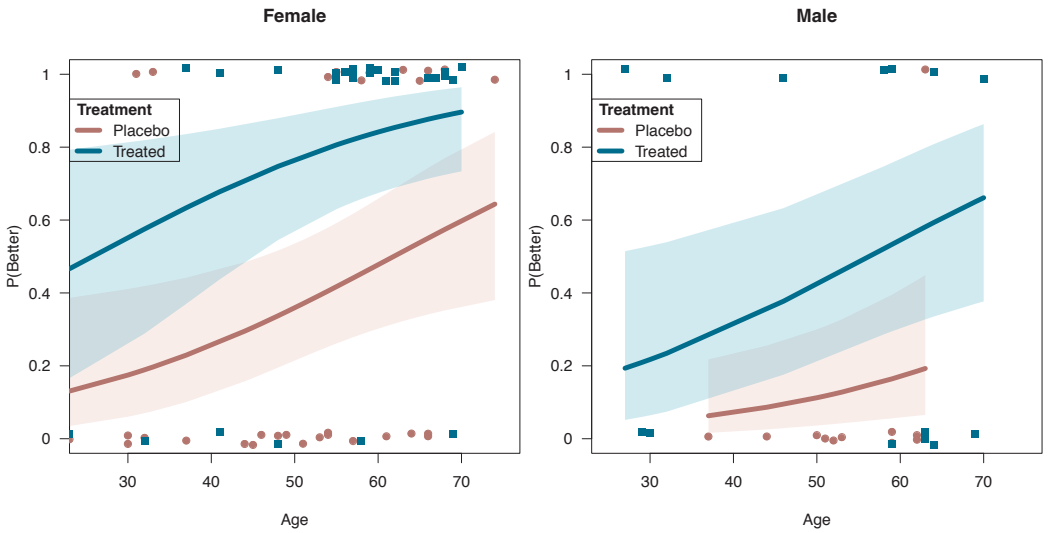


Figure 7.9: Full-model plot of Arthritis data, showing fitted probabilities by Treatment and Sex.

△

7.3.3 Effect plots

For more than two variables, full-model plots of the fitted response surface can be cumbersome, particularly when the model contains interactions or when the main substantive interest is focused on a given main effect or interaction, controlling for all other explanatory variables. The method of ***effect displays*** (tables and graphs), developed by John Fox (1987, 2003) and implemented in the effects package, is a useful solution to these problems.

The idea of effect plots is quite simple but very general and handles models of arbitrary complexity:[8] consider a particular subset of predictors (*focal predictors*) we wish to visualize in a given linear model or generalized linear model. The essence is to calculate fitted values (and standard errors) for the model terms involving these variables and all low-order relatives (e.g., main effects that are marginal to an interaction), as these variables are allowed to vary over their range.

All other variables are "controlled" by being fixed at typical values. For example, a quantitative covariate could be fixed at its mean or median; a factor could be fixed at equal proportions of its levels or its proportions in the data. The result, when plotted, shows all effects of the focal predictors and their low-order relatives, but with all other variables controlled (or "adjusted for").

[8]Less general expression of these ideas include the use of ***adjusted means*** in analysis of covariance, and ***least squares means*** or ***population marginal means*** (Searle et al., 1980) in analysis of variance; for example, see the lsmeans (Lenth and Hervé, 2015) package for classical linear models.

7.3.3.1 The score model matrix*

More formally, assume we have fit a model with a linear predictor $\eta_i = \alpha + \boldsymbol{x}_i^\mathsf{T} \boldsymbol{\beta}$ (on the logit scale, for logistic regression). Letting $\beta_0 = \alpha$ and $\boldsymbol{x}_0 = \mathbf{1}$, we can rewrite this in matrix form as $\boldsymbol{\eta} = \boldsymbol{X}\boldsymbol{\beta}$ where $\boldsymbol{X}$ is the model matrix constructed by the modeling function, such as `glm()`. Fitting the model gives the estimated coefficients $\boldsymbol{b}$ and its estimated covariance matrix $\widehat{\mathcal{V}}(\boldsymbol{b})$.

The `Effect()` function constructs an analogous *score model matrix*, $\boldsymbol{X}^*$, where the focal variables have been varied over their range, and all other variables represented as constant, typical values. Using this as input (the `newdata` argument) to the `predict()` function then gives the fitted values $\boldsymbol{\eta}^* = \boldsymbol{X}^* \boldsymbol{b}$. Standard errors used for confidence intervals are calculated by `predict()` (when `se.fit=TRUE`) as the square roots of $\text{diag}\,(\boldsymbol{X}^* \widehat{\mathcal{V}(\boldsymbol{b})} \boldsymbol{X}^{*\mathsf{T}})$. Note that these ideas work not only for `glm()` models, but potentially for any modeling function that has a `predict()` and `vcov()` method.[9]

These results are calculated on the scale of the linear predictor $\boldsymbol{\eta}$ (logits, for logistic regression) when the `type` argument to `predict()` is `type="link"` or on the response scale (probabilities, here) when `type="response"`. The latter makes use of the inverse transformation, Eqn. (7.6).

There are two main calculation functions in the effects package:

- `Effect()` takes a character vector of the names of a subset of focal predictors and constructs the score matrix $\boldsymbol{X}^*$ by varying these over their ranges, while holding all other predictors constant at "typical" values. There are many options that control these calculations. For example, `xlevels` can be used to specify the values of the focal predictors; `typical` or `given.values`, respectively, can be used to specify either a function (`mean`, `median`) or a list of specific typical values used for the variables that are controlled. The result is an object of class "eff", for which there are `print()`, `summary()`, and (most importantly) `plot()` methods. See `help(Effect)` for a complete description.

- `allEffects()` takes a model object, and calculates the effects for each high-order term in the model (including their low-order) relatives. Similar optional arguments control the details of the computation. The result is an object of class "efflist".

In addition, the plotting methods for "eff" and "efflist" objects offer numerous options to control the plot details, only a few of which are used in the examples below. For logistic regression models, they also solve the problem of the trade-off between plots on the logit scale, which have a simple representation in terms of additive effects, and plots on the probability scale which are usually simpler to understand. By default, the fitted model effects are plotted on the logit scale, but the response y axis is labeled with the corresponding probability values.

7.3.3.2 Partial residuals

We noted earlier that for discrete response data, it is usually important to display the *data* in some fashion, along with the fitted relationship. Conditional and full-model plots do this by jittering the binary values at 0 and 1 so you can see where the data exists.

The effects package takes this idea further, by allowing the display of ***partial residuals***. Letting $\boldsymbol{r}$ denote the vector of residuals for a given model (see Section 7.5.1 for details), the partial residuals $\boldsymbol{r}_j$ pertaining to predictor $\boldsymbol{x}_j$ are defined as

$$\boldsymbol{r}_j = \boldsymbol{r} + \widehat{\beta}_j \boldsymbol{x}_j \; .$$

[9] For example, the effects package presently provides methods for models fit by `lm()` (including multivariate linear response models), `glm()`, `gls()`, multinomial (`multinom()` in the nnet (Ripley, 2015b) package) and proportional odds models (`polr()` in MASS), polytomous latent class models (poLCA (Linzer and Lewis., 2014) package), as well as a variety of multi-level and mixed-effects linear models fit with `lmer()` from the lme4 (Bates et al., 2014) package, or with `lme()` from the nlme (Pinheiro et al., 2015) package.

These are a natural extension of residuals in simple regression to the multiple regression setting, in that the slope of a simple regression of $\boldsymbol{r}$ on $\boldsymbol{x}$ is equal to the value of $\widehat{\beta}_j$ in the full multiple regression model (Cook, 1993). Adding partial residuals to an effect plot (together with a non-parametric smoothing) can help to visualize lack of fit or misspecification of the response mean attributable to continuous predictors, such as a nonlinear relation or an omitted interaction (Fox and Weisberg, 2015b).

EXAMPLE 7.8: Arthritis treatment

Here we illustrate the use of the effects package with the simple main effects model that was fit in Example 7.5. `allEffects()` is used to calculate the predicted probabilities of the `Better` response for `Age` and the two factors, `Sex` and `Treatment`. Partial residuals (for quantitative predictors) must be requested in the call to `allEffects()` or `Effect()`.

```
> library(effects)
> arth.eff2 <- allEffects(arth.logistic2, partial.residuals = TRUE)
> names(arth.eff2)

[1] "I(Age-50)" "Sex"       "Treatment"
```

The result, `arth.eff2`, is a list containing the fitted values (response probabilities, by default) for each of the model terms. For example the main effect for `Sex` is shown below; the associated score `model.matrix` illustrates how `Sex` is varied over its range, while `Age-50` and `Treatment` are fixed at their average values in the data.

```
> arth.eff2[["Sex"]]


 Sex effect
Sex
 Female    Male
0.60932 0.26050

> arth.eff2[["Sex"]]$model.matrix

  (Intercept) I(Age - 50) SexMale TreatmentTreated
1           1      3.3571       0           0.4881
2           1      3.3571       1           0.4881
```

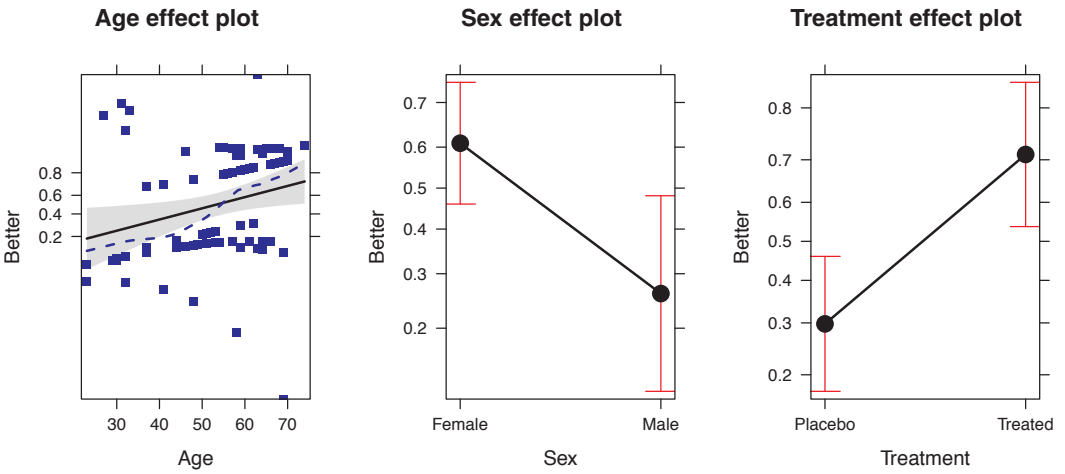


Figure 7.10: Plot of all effects in the main effects model for the Arthritis data. Partial residuals and their loess smooth are also shown for the continuous predictor, Age.

The default plot method for the "efflist" object produces one plot for each high-order term, which are just the main effects in this model. The call below produces Figure 7.10.

```
> plot(arth.eff2, rows = 1, cols = 3,
+       type="response", residuals.pch = 15)
```

The smoothed loess curve for the partial residuals with respect to age show a hint of nonlinearity, but perhaps not enough to worry about.

You can quite easily also produce effect plots for several predictors jointly, or full-model plots by using all predictors in the model in a call to `Effect()`, as shown in the call below.

```
> arth.full <- Effect(c("Age", "Treatment", "Sex"), arth.logistic2)
```

Then plotting the result, with some options, gives the plot shown in Figure 7.11.

```
> plot(arth.full, multiline = TRUE, ci.style = "bands",
+       colors = c("red", "blue"), lwd = 3,
+       ticks = list(at = c(.05, .1, .25, .5, .75, .9, .95)),
+       key.args = list(x = .52, y = .92, columns = 1),
+       grid = TRUE)
```

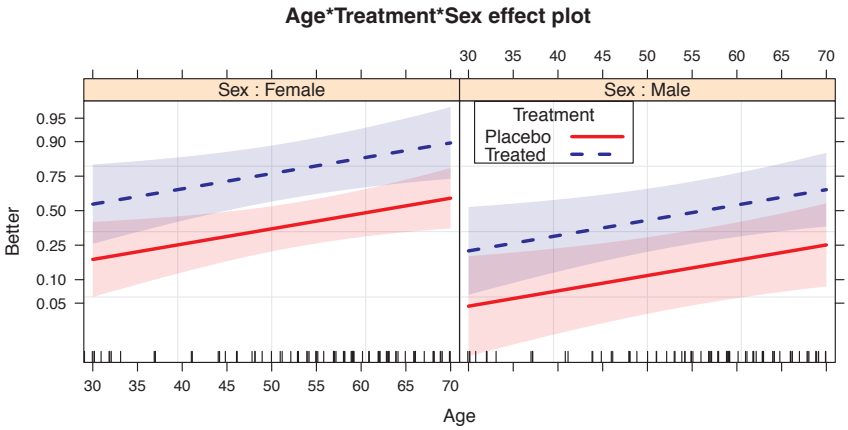


Figure 7.11: Full-model plot of the effects of all predictors in the main effects model for the Arthritis data, plotted on the logit scale.

Alternatively, we can plot these results directly on the scale of probabilities, as shown in Figure 7.12.

```
> plot(arth.full, multiline = TRUE, ci.style = "bands",
+       type="response",
+       colors = c("red", "blue"), lwd = 3,
+       key.args = list(x = .52, y = .92, columns = 1),
+       grid = TRUE)
```

△

7.4 Case studies

The examples below take up some issues of data analysis, model building, and visualization in the context of multiple logistic regression models. We focus on the combination of exploratory plots to see the data, modeling steps, and graphs to interpret a given model.

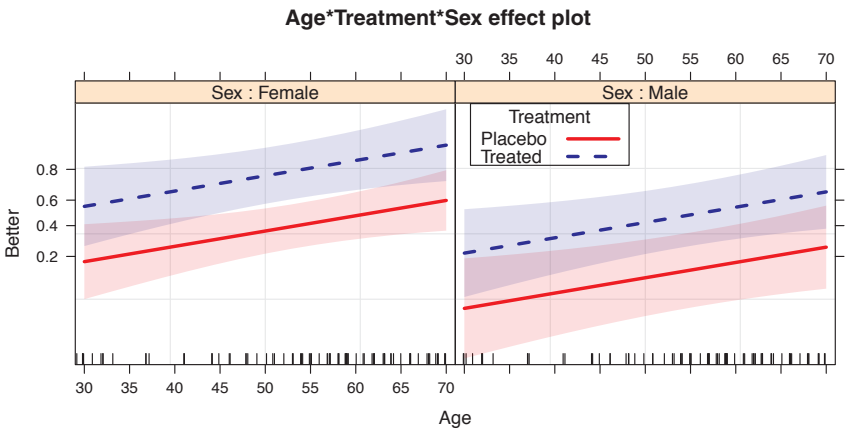


Figure 7.12: Full-model plot of the effects of all predictors in the main effects model for the Arthritis data, plotted on the probability scale.

7.4.1 Simple models: Group comparisons and effect plots

EXAMPLE 7.9: Donner Party

In Chapter 1, Example 1.3, we described the background behind the sad story of the Donner Party, perhaps the most famous tragedy in the history of the westward settlement in the United States. In brief, the party was stranded on the eastern side of the Sierra Nevada mountains by heavy snow in late October 1846, and by the time the last survivor was rescued in April 1847, nearly half of the members had died from famine and exposure to extreme cold. Figure 1.3 showed that survival decreased strongly with age.

Here we consider a more detailed analysis of these data, which are contained in the data set *Donner* in vcdExtra. This data set lists 90 people in the Donner Party by name, together with age, sex, survived (0/1), and the date of death for those who died.[10]

```
> data("Donner", package = "vcdExtra")    # load the data
> library(car)                            # for some() and Anova()
> some(Donner, 8)

                      family age    sex survived      death
Breen, Peter           Breen   3   Male        1       <NA>
Donner, Jacob         Donner  65   Male        0 1846-12-21
Foster, Jeremiah   MurFosPik   1   Male        0 1847-03-13
Graves, Nancy         Graves   9 Female        1       <NA>
McCutchen, Harriet McCutchen   1 Female        0 1847-02-02
Reed, James             Reed  46   Male        1       <NA>
Reinhardt, Joseph      Other  30   Male        0 1846-12-21
Wolfinger, Doris    FosdWolf  20 Female        1       <NA>
```

The main purpose of this example is to try to understand, through graphs and models, how survival was related to age and sex. However, first, we do some data preparation and exploration. The response variable, `survived`, is a 0/1 integer, and it is more convenient for some purposes to make it a factor.

[10] Most historical sources count the number in the Donner Party at 87 or 89. An exact accounting of the members of the Donner Party is difficult, because: (a) several people joined the party in mid-route, at Fort Bridger and in the Wasatch Mountains; (b) several rode ahead to search for supplies and one (Charles Stanton) brought two more with him (Luis and Salvador); (c) five people died before reaching the Sierra Nevada mountains. *Donner* incorporates updated information from Kristin Johnson's listing, `http://user.xmission.com/~octa/DonnerParty/Roster.htm`.

```
> Donner$survived <- factor(Donner$survived, labels = c("no", "yes"))
```

Some historical accounts (Grayson, 1990) link survival in the Donner Party to kinship or family groups, so we take a quick look at this factor here. The variable `family` reflects a recoding of the last names of individuals to reduce the number of factor levels. The main families in the Donner party were: Donner, Graves, Breen, and Reed. The families of Murphy, Foster, and Pike are grouped as `"MurFosPik"`, those of Fosdick and Wolfinger are coded as `"FosdWolf"`, and all others as `"Other"`.

```
> xtabs(~ family, data = Donner)

family
    Breen    Donner      Eddy  FosdWolf    Graves  Keseberg
        9        14         4         4        10         4
McCutchen MurFosPik     Other      Reed
        3        12        23         7
```

For the present purposes, we reduce these 10 family groups further, collapsing some of the small families into `"Other"`, and reordering the levels. Assigning new values to the `levels()` of a factor is a convenient trick for recoding factor variables.

```
> # collapse small families into "Other"
> fam <- Donner$family
> levels(fam)[c(3, 4, 6, 7, 9)] <- "Other"
>
> # reorder, putting Other last
> fam = factor(fam,levels(fam)[c(1, 2, 4:6, 3)])
> Donner$family <- fam
> xtabs(~family, data=Donner)

family
    Breen    Donner    Graves MurFosPik      Reed     Other
        9        14        10        12         7        38
```

`xtabs()` then shows the counts of survival by these family groups:

```
> xtabs(~ survived + family, data = Donner)

        family
survived Breen Donner Graves MurFosPik Reed Other
     no      0      7      3         6    1    25
     yes     9      7      7         6    6    13
```

Plotting this distribution of survival by family with a formula gives a ***spineplot***, a special case of the mosaic plot, or a generalization of a stacked bar plot, shown in Figure 7.13. The widths of the bars are proportional to family size, and the shading highlights in light blue the proportion who survived in each family.

```
> plot(survived ~ family, data = Donner, col = c("pink", "lightblue"))
```

A generalized pairs plot (Section 5.6.2), shown in Figure 7.14, gives a visual overview of the data. The diagonal panels here show the marginal distributions of the variables as bar plots, and highlight the skewed distribution of age and the greater number of males than females in the party. The boxplots and barcode plots for survived and age show that those who survived were generally younger than those who perished.

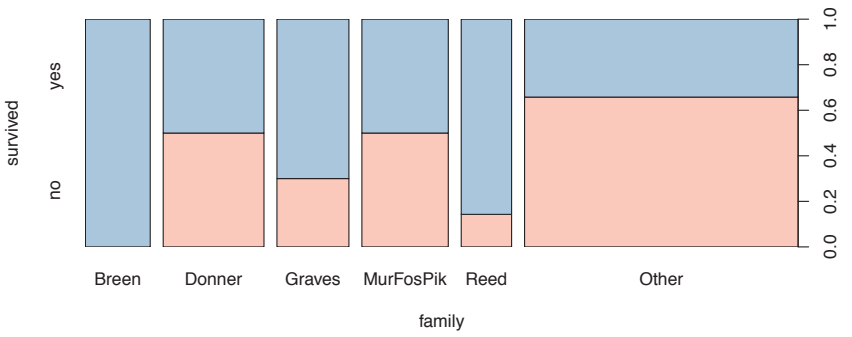


Figure 7.13: Spineplot of survival in the Donner Party by family.

```
> library(gpairs)
> library(vcd)
> gpairs(Donner[,c(4, 2, 3, 1)],
+      diag.pars = list(fontsize = 20, hist.color = "gray"),
+      mosaic.pars = list(gp = shading_Friendly),
+      outer.rot = c(45, 45)
+ )
```

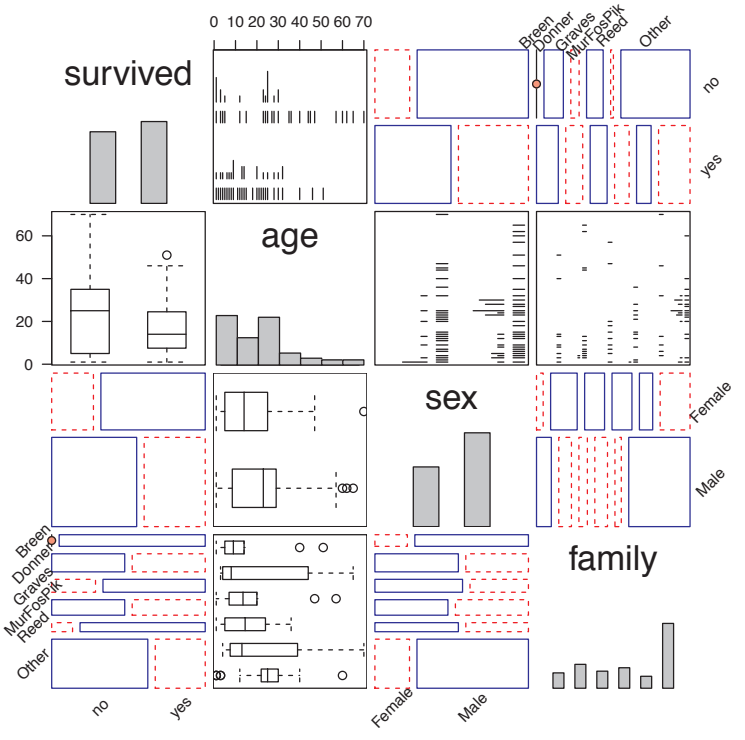


Figure 7.14: Generalized pairs plot for the Donner data.

From an exploratory perspective, we now proceed to examine the relationship of survival to age and sex, beginning with the kind of conditional plots we illustrated earlier (in Example 7.6). Figure 7.15 shows a plot of `survived`, converted back to a 0/1 variable as required by `ggplot()`, together with the binary responses as points and the logistic regressions fitted separately for males and females.

```
> # basic plot: survived vs. age, colored by sex, with jittered points
> gg <- ggplot(Donner, aes(age, as.numeric(survived=="yes"),
+                          color = sex)) +
+   ylab("Survived") + theme_bw() +
+   geom_point(position = position_jitter(height = 0.02, width = 0))
>
> # add conditional linear logistic regressions
> gg + stat_smooth(method = "glm", family = binomial, formula = y ~ x,
+                  alpha = 0.2, size = 2, aes(fill = sex))
```

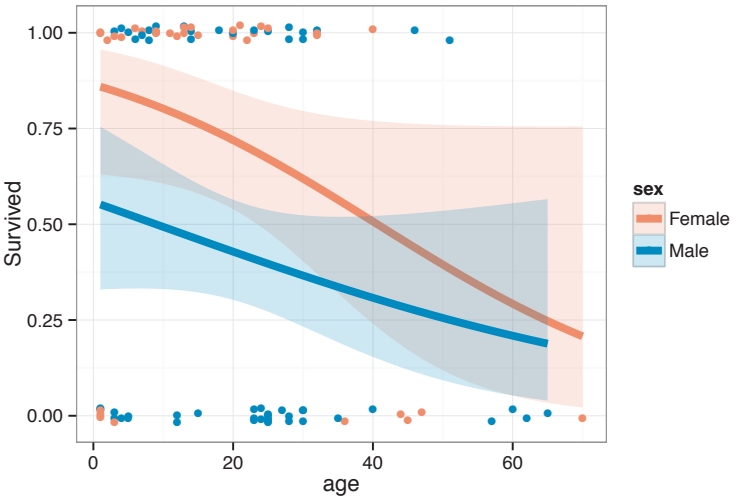


Figure 7.15: Conditional plot of the Donner data, showing the relationship of survival to age and sex. The smoothed curves and confidence bands show the result of fitting separate linear logistic regressions on age for males and females.

It is easy to see that survival among women was greater than for men, perhaps narrowing the gap among the older people, but the data gets thin towards the upper range of age.

The curves plotted in Figure 7.15 assume a linear relationship between the log odds of survival and age (expressed as `formula = y ~ x` in the call to `stat_smooth()`). One simple way to check whether the relationship between survival and age is nonlinear is to re-do this plot, but now allow a quadratic relationship with age, using `formula = y ~ poly(x,2)`. The result is shown in the left panel of Figure 7.16.

```
> # add conditional quadratic logistic regressions
> gg + stat_smooth(method = "glm", family = binomial,
+                  formula = y ~ poly(x,2), alpha = 0.2, size = 2,
+                  aes(fill = sex))
>
> # add loess smooth
> gg + stat_smooth(method = "loess", span=0.9, alpha = 0.2, size = 2,
+                  aes(fill = sex)) +
+   coord_cartesian(ylim = c(-.05,1.05))
```

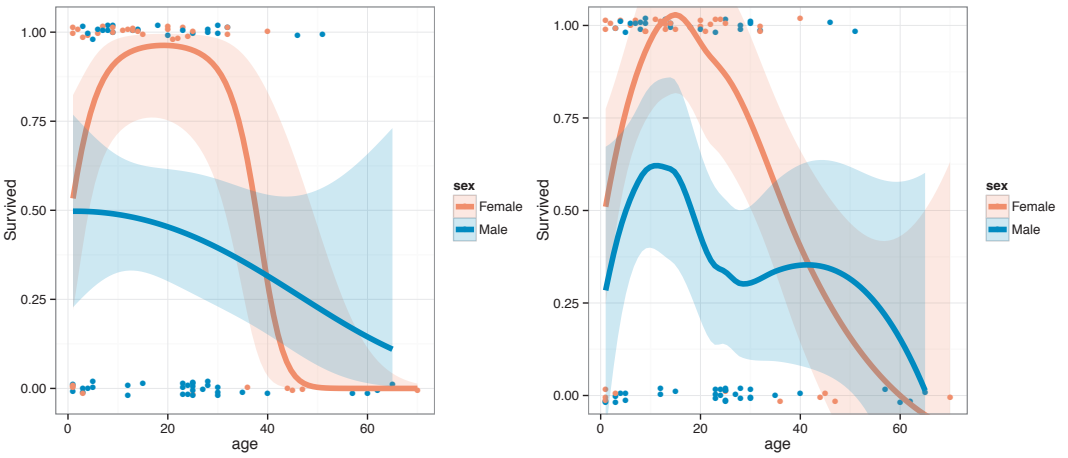


Figure 7.16: Conditional plots of the Donner data, showing the relationship of survival to age and sex. Left: The smoothed curves and confidence bands show the result of fitting separate quadratic logistic regressions on age for males and females. Right: Separate loess smooths are fit to the data for males and females.

This plot is quite surprising. It suggests quite different regimes relating to survival for men and women. Among men, survival probability decreases steadily with age, at least after age 20. For women, those in the age range 10–35 were very likely to have lived, while those over 40 were almost all predicted to perish.

Another simple technique is to fit a non-parametric loess smooth, as shown in the right panel of Figure 7.16.[11] The curve for females is similar to that of the quadratic fit in the left panel, but the curve for males suggests that survival also has a peak around the teenage years. One lesson to be drawn from these graphs is that a linear logistic regression, as shown in Figure 7.16, may tell only part of the story, and, for a binary response, it is not easy to discern whether the true relationship is linear. If it really is, all these graphs would look much more similar. As well, we usually obtain a more realistic smoothing of the data using full-model plots or effect plots.

The suggestions from these exploratory graphs can be used to define and test some models for survival in the Donner Party. The substantive questions of interest are:

- Is the relationship different for men and women? This is, is it necessary to allow for an interaction of age with sex, or separate fitted curves for men and women?
- Is the relationship between survival and age well-represented in a linear logistic regression model?

The first question is the easiest to deal with: we can simply fit a model allowing an interaction of age (or some function of age) and sex,

```
survived ~ age * sex
survived ~ f(age) * sex
```

and compare the goodness of fit with the analogous additive, main-effects models.

From a modeling perspective, there is a wide variety of approaches for testing for nonlinear relationships. We only scratch the surface here, and only for a single quantitative predictor, x, such

[11] A technical problem with the use of the loess smoother for binary data is that it can produce fitted values outside the [0–1] interval, as happens in the right panel of this figure. Kernel smoothers, such as the KernSmooth (Wand, 2015) package avoid this problem, but are not available through ggplot2.

as age in this example. One simple approach, illustrated in Figure 7.16, is to allow a quadratic (or higher-power, e.g., cubic) function to describe the relationship between the log odds and x,

$$\begin{aligned} \text{logit}(\pi_i) &= \alpha + \beta_1 x_i + \beta_2 x_i^2 \\ \text{logit}(\pi_i) &= \alpha + \beta_1 x_i + \beta_2 x_i^2 + \beta_3 x_i^3 \\ \cdots &\quad . \end{aligned}$$

In R, these model terms can be fit using `poly(x, 2)`, `poly(x, 3)` ..., which generate orthogonal polynomials for the powers of x. A simple way to test for nonlinearity is a likelihood ratio test comparing the more complex model to the linear one. This method is often sufficient for a hypothesis test, and, if the relationship truly is linear, the fitted logits and probabilities will not differ greatly from what they would be under a linear model. A difficulty with this approach is that polynomial models are often unrealistic, particularly for data that approach an asymptote.

Another simple approach is to use a ***regression spline***, which fits the relationship with x in terms of a set of piecewise polynomials, usually cubic, joined at a collection of points, called *knots*, so that the overall fitted relationship is smooth and continuous. See Fox (2008, Section 17.2) for a cogent, brief description of these methods.

One particularly convenient method is a ***natural spline***, implemented in the splines package in the `ns()` function. This method constrains the fitted cubic spline to be linear at lower and upper limits of x, and, for k knots, fits df $= k + 1$ parameters not counting the intercept. The k knots can be conveniently chosen as k cutpoints in the percentiles of the distribution of x. For example, with $k = 1$, the knot would be placed at the median, or 50^{th} percentile; with $k = 3$, the knots would be placed at the quartiles of the distribution of x; $k = 0$ corresponds to no knots, i.e., a simple linear regression.

In the `ns()` function, you can specify the locations of knots or the number of knots with the `knots` argument, but it is conceptually simpler to specify the number of degrees of freedom used in the spline fit. Thus, `ns(x, 2)` and `poly(x, 2)` both specify a term in `x` of the same complexity, the former a natural spline with $k = 1$ knot and the later a quadratic function in `x`.

We illustrate these ideas in the remainder of this example, fitting a 2×2 collection of models to the *Donner* data corresponding to: (a) whether or not age and sex effects are additive; (b) whether the effect is linear on the logit scale or nonlinear (quadratic, here). A brief summary of each model is given using the `Anova()` in the car package, providing Type II tests of each effect. As usual, `summary()` would give more detailed output, including tests for individual coefficients. First, we fit the linear models, without and with an interaction term:

```
> donner.mod1 <- glm(survived ~ age + sex,
+                    data = Donner, family  =binomial)
> Anova(donner.mod1)

Analysis of Deviance Table (Type II tests)

Response: survived
    LR Chisq Df Pr(>Chisq)
age     5.52  1     0.0188 *
sex     6.73  1     0.0095 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

> donner.mod2 <- glm(survived ~ age * sex,
+                    data = Donner, family = binomial)
> Anova(donner.mod2)

Analysis of Deviance Table (Type II tests)
```

```
Response: survived
        LR Chisq Df Pr(>Chisq)
age         5.52  1     0.0188 *
sex         6.73  1     0.0095 **
age:sex     0.40  1     0.5269
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The main effects of `age` and `sex` are both significant here, but the interaction term, `age:sex`, is not in model `donner.mod2`. Note that the terms tested by `Anova()` in `donner.mod1` are a redundant subset of those in `donner.mod2`.

Next, we fit nonlinear models, representing the linear and nonlinear trends in age by `poly(age,2)`.[12] The `Anova()` results for terms in both models are contained in the output from `Anova(donner.mod4)`.

```
> donner.mod3 <- glm(survived ~ poly(age, 2) + sex,
+                    data = Donner, family = binomial)
> donner.mod4 <- glm(survived ~ poly(age, 2) * sex,
+                    data = Donner, family = binomial)
> Anova(donner.mod4)

Analysis of Deviance Table (Type II tests)

Response: survived
                 LR Chisq Df Pr(>Chisq)
poly(age, 2)         9.91  2     0.0070 **
sex                  8.09  1     0.0044 **
poly(age, 2):sex     8.93  2     0.0115 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Now, in model `donner.mod4`, the interaction term `poly(age, 2):sex` is significant, indicating that the fitted quadratics for males and females differ in "shape," meaning either their linear (slope) or quadratic (curvature) components.

These four models address the questions posed earlier. A compact summary of these models, giving the likelihood ratio tests of goodness of fit, together with AIC and BIC statistics, are shown below, using the `LRstats()` method in vcdExtra for a list of "glm" models.

```
> library(vcdExtra)
> LRstats(donner.mod1, donner.mod2, donner.mod3, donner.mod4)

Likelihood summary table:
            AIC BIC LR Chisq Df Pr(>Chisq)
donner.mod1 117 125    111.1 87      0.042 *
donner.mod2 119 129    110.7 86      0.038 *
donner.mod3 115 125    106.7 86      0.064 .
donner.mod4 110 125     97.8 84      0.144
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

By AIC and BIC, `donner.mod4` is best, and it is also the only model with a non-significant LR χ^2 (residual deviance). Because these models comprise a 2×2 set of hypotheses, it is easier to compare models by extracting the LR statistics and arranging these in a table, together with their row and column differences. The entries in the table below are calculated as follows.

[12] Alternatively, we could use the term `ns(age,2)`, or higher-degree polynomials, or natural splines with more knots, but we don't do this here.

```
> mods <- list(donner.mod1, donner.mod2, donner.mod3, donner.mod4)
> LR <- sapply(mods, function(x) x$deviance)
> LR <- matrix(LR, 2, 2)
> rownames(LR) <- c("additive", "non-add")
> colnames(LR) <- c("linear", "non-lin")
> LR <- cbind(LR, diff = LR[,1] - LR[,2])
> LR <- rbind(LR, diff = c(LR[1,1:2] - LR[2,1:2], NA))
```

	linear	non-linear	$\Delta\chi^2$	p-value
additive	111.128	106.731	4.396	0.036
non-additive	110.727	97.799	12.928	0.000
$\Delta\chi^2$	0.400	8.932		
p-value	0.527	0.003		

Thus, the answer to our questions seems to be that: (a) there is evidence that the relationship of survival to age differs for men and women in the Donner Party; (b) these relationships are not well-described by a linear logistic regression.

For simplicity, we used a quadratic effect, `poly(age,2)`, to test for nonlinearity here. An alternative test of the same complexity could use a regression spline, `ns(age,2)`, also with 2 degrees of freedom for the main effect and interaction, or allow more knots. To illustrate, we fit two natural spline modes models with 2 and 4 df, and compare these with the quadratic model (`donner.mod4`), all of which include the interaction of age and sex.

```
> library(splines)
> donner.mod5 <- glm(survived ~ ns(age,2) * sex,
+                    data = Donner, family = binomial)
> Anova(donner.mod5)

Analysis of Deviance Table (Type II tests)

Response: survived
              LR Chisq Df Pr(>Chisq)
ns(age, 2)        9.28  2     0.0097 **
sex               7.98  1     0.0047 **
ns(age, 2):sex    8.71  2     0.0129 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

> donner.mod6 <- glm(survived ~ ns(age,4) * sex,
+                    data = Donner, family = binomial)
> Anova(donner.mod6)

Analysis of Deviance Table (Type II tests)

Response: survived
              LR Chisq Df Pr(>Chisq)
ns(age, 4)       22.05  4     0.0002 ***
sex              10.49  1     0.0012 **
ns(age, 4):sex    8.54  4     0.0737 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

> LRstats(donner.mod4, donner.mod5, donner.mod6)

Likelihood summary table:
            AIC BIC LR Chisq Df Pr(>Chisq)
donner.mod4 110 125     97.8 84       0.14
donner.mod5 111 126     98.7 84       0.13
donner.mod6 106 131     86.1 80       0.30
```

With four more parameters, `donner.mod6` fits better and has a smaller AIC.

We conclude this example with an effect plot for the spline model `donner.mod6` shown in Figure 7.17. The complexity of the fitted relationships for men and women is intermediate between the two conditional plots shown in Figure 7.16. (However, note that the fitted effects are plotted on the logit scale in Figure 7.17 and labeled with the corresponding probabilities, whereas the conditional plots are plotted directly on the probability scale.)

```
> library(effects)
> donner.eff6 <- allEffects(donner.mod6, xlevels = list(age=seq(0, 50, 5)))
> plot(donner.eff6, ticks = list(at=c(0.001, 0.01, 0.05, 0.1, 0.25, 0.5,
+                                     0.75, 0.9, 0.95, 0.99, 0.999)))
```

Figure 7.17: Effect plot for the spline model `donner.mod6` fit to the Donner data.

This plot confirms that for women in the Donner Party, survival was greatest for those aged 10–30. Survival among men was overall much less and there is a hint of greater survival for men aged 10–15.

Of course, this statistical analysis does not provide explanations for these effects, and it ignores the personal details of the Donner Party members and the individual causes and circumstances of death, which are generally well-documented in the historical record (Johnson, 1996). See `http://user.xmission.com/~octa/DonnerParty/` for a comprehensive collection of historical sources.

Grayson (1990) attributes the greater survival of women of intermediate age to demographic arguments that women are overall better able to withstand conditions of famine and extreme cold, and high age-specific mortality rates among the youngest and oldest members of human societies. He also concludes (without much analysis) that members with larger social and kinship networks would be more likely to survive. △

EXAMPLE 7.10: Racial profiling: Arrests for marijuana possession

In the summer of 2002, the *Toronto Star* newspaper launched an investigation on the topic of possible racial profiling by the Toronto police service. Through freedom of information requests, they obtained a data base of over 600,000 arrest records on all potential charges in the period from 1996–2002, the largest data bases on crime arrests and disposition ever assembled in Canada. An initial presentation of this study was given in Example 1.4.

In order to examine the issue of racial profiling (different treatment as a function of race) they excluded all charges such as assault, robbery, speeding, and driving under the influence, where the police have no discretion regarding the laying of a charge. They focused instead on a subset of arrests, where the police had various options.

Among these, for people arrested for a single charge of simple possession of a small amount of marijuana, police have the option of releasing the arrestee, with a summons ("Form 9") to appear in court (similar to a parking ticket), or else the person could be given harsher treatment—brought to a police station or held in jail for a bail hearing ("Show cause"). The main question for the *Toronto Star* was whether the subject's skin color had any influence on the likelihood that the person would be released with a summons.[13]

Their results, published in a week-long series of articles in December 2002, concluded that there was strong evidence that black and white subjects were treated differently. For example, the analysis showed that blacks were 1.5 times more likely than whites to be given harsher treatment than release with a summons; if the subject was taken to the police station, a black was 1.6 times more likely to be held in jail for a bail hearing. An important part of the analysis and the public debate that ensued was to show that other variables that might account for these differences had been controlled or adjusted for.[14]

The data set `Arrests` in the effects package gives a simplified version of the *Star* database, containing records for 5,226 cases of arrest on the charge of simple possession of marijuana analyzed by the newspaper. The response variable here is `released` (Yes/No) and the main predictor of interest is skin color of the person arrested, `colour` (Black/White).[15] A random subset of the data set is shown below.

```
> library(effects)
> data("Arrests", package = "effects")
> Arrests[sample(nrow(Arrests), 6),]

     released colour year age  sex employed citizen checks
3768      Yes  Black 2000  23 Male       No     Yes      4
4576      Yes  Black 2001  17 Male      Yes     Yes      0
3976       No  White 2002  20 Male       No     Yes      3
4629      Yes  White 2000  18 Male      Yes     Yes      1
2384       No  Black 2000  19 Male      Yes     Yes      3
869       Yes  White 2001  15 Male      Yes     Yes      1
```

Other available predictors, to be used as control variables, included the `year` of the arrest, `age` and `sex` of the person, and binary indicators of whether the person was `employed` and a `citizen` of Canada. In addition, when someone is stopped by police, his/her name is checked in six police data bases that record previous arrests, convictions, whether on parole, etc. The variable `checks` records the number, 0–6, in which the person's name appeared.

A variety of logistic models were fit to these data including all possible main effects and some two-way interactions. To allow for possible nonlinear effects of `year`, this variable was treated as a factor rather than as a (linear) numeric variable, but the effects of `age` and `checks` were reasonably linear on the logit scale. A reasonable model included the interactions of `colour` with both `year` and `age`, as fit below:

[13] Another discretionary charge they investigated was police stops for non-moving violations under the Ontario *Highway Traffic Act*, such as being pulled over for a faulty muffler or having an expired license plate renewal sticker. A disproportionate rate of charges against blacks is sometimes referred to as "driving while black" (DWB). This investigation found that the number of blacks so charged, but particularly young black males, far outweighed their representation in the population.

[14] The Toronto Police Service launched a class-action libel law suit against the *Toronto Star* and the first author of this book, who served as their statistical consultant, claiming damages of $5,000 for every serving police officer in the city, a total of over 20 million dollars. The suit was thrown out of court, and the Toronto police took efforts to enhance training programs to combat the perception of racial profiling.

[15] The original data set also contained the categories Brown and Other, but these appeared with small frequencies.

```
> Arrests$year <- as.factor(Arrests$year)
> arrests.mod <- glm(released ~ employed + citizen + checks
+                    + colour*year + colour*age,
+                    family = binomial, data = Arrests)
```

For such models, significance tests for the model terms are best carried out using the `Anova()` function in the car package that uses Type II tests:

```
> library(car)
> Anova(arrests.mod)

Analysis of Deviance Table (Type II tests)

Response: released
           LR Chisq Df Pr(>Chisq)
employed       72.7  1    < 2e-16 ***
citizen        25.8  1    3.8e-07 ***
checks        205.2  1    < 2e-16 ***
colour         19.6  1    9.7e-06 ***
year            6.1  5    0.29785
age             0.5  1    0.49827
colour:year    21.7  5    0.00059 ***
colour:age     13.9  1    0.00019 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The difficulty in interpreting these results from tables of coefficients can be seen in the output below:

```
> coeftest(arrests.mod)


z test of coefficients:

                     Estimate Std. Error z value Pr(>|z|)
(Intercept)           0.34443    0.31007    1.11  0.26665
employedYes           0.73506    0.08477    8.67  < 2e-16 ***
citizenYes            0.58598    0.11377    5.15  2.6e-07 ***
checks               -0.36664    0.02603  -14.08  < 2e-16 ***
colourWhite           1.21252    0.34978    3.47  0.00053 ***
year1998             -0.43118    0.26036   -1.66  0.09770 .
year1999             -0.09443    0.26154   -0.36  0.71805
year2000             -0.01090    0.25921   -0.04  0.96647
year2001              0.24306    0.26302    0.92  0.35541
year2002              0.21295    0.35328    0.60  0.54664
age                   0.02873    0.00862    3.33  0.00086 ***
colourWhite:year1998  0.65196    0.31349    2.08  0.03756 *
colourWhite:year1999  0.15595    0.30704    0.51  0.61152
colourWhite:year2000  0.29575    0.30620    0.97  0.33411
colourWhite:year2001 -0.38054    0.30405   -1.25  0.21073
colourWhite:year2002 -0.61732    0.41926   -1.47  0.14091
colourWhite:age      -0.03737    0.01020   -3.66  0.00025 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

By direct calculation (e.g., using `exp(coef(arrests.mod))`) you can find that the odds of a quick release was $\exp(0.735) = 2.08$ times greater for someone employed, $\exp(0.586) = 1.80$ times more likely for a Canadian citizen, and $\exp(1.21) = 3.36$ times more likely for a white than a black person. It is much more difficult to interpret the interaction terms.

The primary question for the newspaper concerned the overall difference between the the treatment of blacks and whites— the main effect of `colour`. We plot this as shown below, giving the plot shown in Figure 7.18. This supports the claim by the *Star* because the 95% confidence limits for blacks and whites do not overlap, and all other relevant predictors that could account for this effect have been controlled or adjusted for.

```
> plot(Effect("colour", arrests.mod),
+       lwd = 3, ci.style = "bands", main = "",
+       xlab = list("Skin color of arrestee", cex = 1.25),
+       ylab = list("Probability(released)", cex = 1.25)
+   )
```

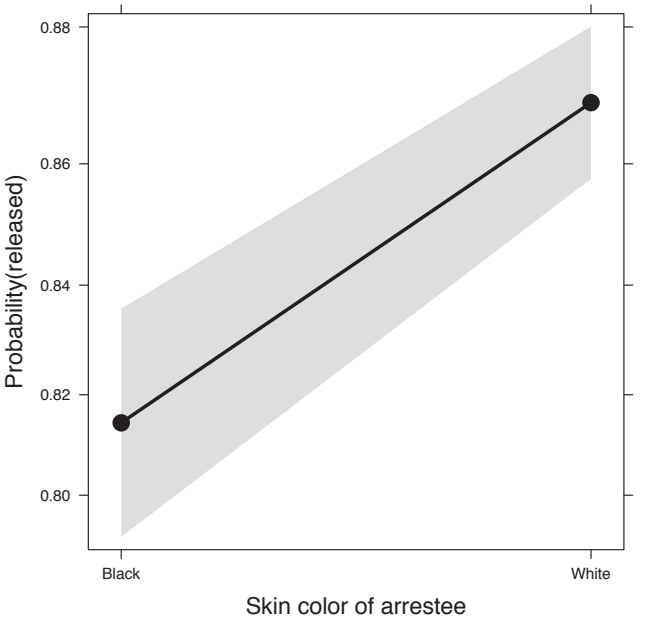


Figure 7.18: Effect plot for the main effect of skin color in the Arrests data.

Of course, one should be very wary of interpreting main effects when there are important interactions, and the story turned out to be far more nuanced than was reported in the newspaper. In particular, the interactions of color with with age and year provided a more complete account. Effect plots for these interactions are shown in Figure 7.19.

```
> # colour x age interaction
> plot(Effect(c("colour", "age"), arrests.mod),
+       lwd = 3, multiline = TRUE, ci.style = "bands",
+       xlab = list("Age", cex = 1.25),
+       ylab = list("Probability(released)", cex = 1.25),
+       key.args = list(x = .05, y = .99, cex = 1.2, columns = 1)
+       )
> # colour x year interaction
> plot(Effect(c("colour", "year"), arrests.mod),
+       lwd = 3, multiline = TRUE,
+       xlab = list("Year", cex = 1.25),
+       ylab = list("Probability(released)", cex = 1.25),
+       key.args = list(x = .7, y = .99, cex = 1.2, columns = 1)
+       )
```

From the left panel in Figure 7.19, it is immediately apparent that the effect of age was in opposite directions for blacks and whites: Young blacks were indeed treated more severely than young whites; however, for older people, blacks were treated less harshly than whites, controlling for all other predictors.

The right panel of Figure 7.19 shows the changes over time in the treatment of blacks and whites. It can be seen that up to the year 2000 there was strong evidence for differential treatment on these

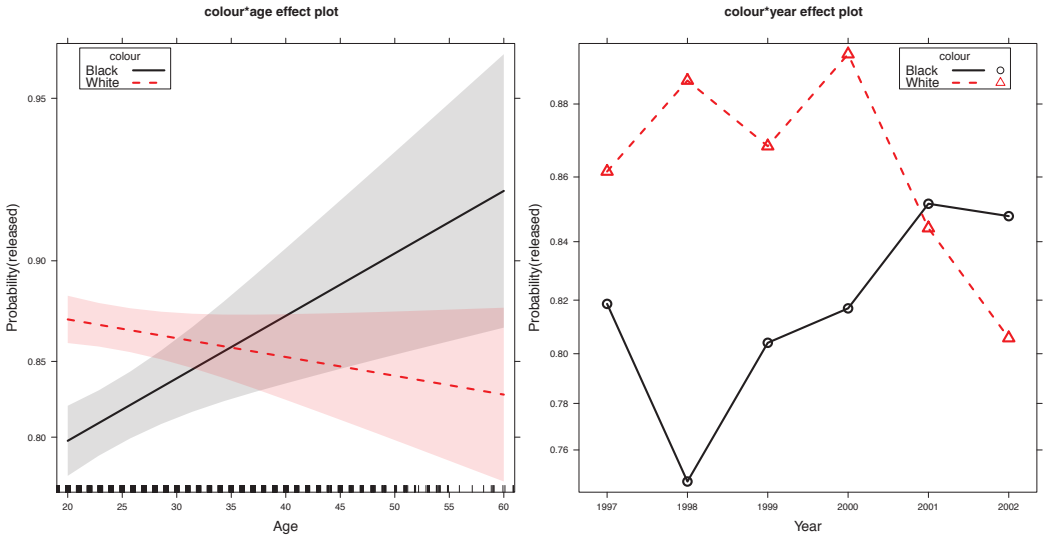


Figure 7.19: Effect plots for the interactions of color with age (left) and year (right) in the Arrests data.

charges, again controlling for other predictors. There was also evidence to support the claim by the police that in the year 2001 they began training of officers to reduce racial effects in treatment.

Finally, the effects package provides a convenience function, `allEffects()`, that calculates the effects for all high-order terms in a given model. The `plot()` method for the "efflist" object can be used to plot individual terms selectively from a graphic menu, or plot all terms together in one comprehensive display using `ask=FALSE`.

```
> arrests.effects <- allEffects(arrests.mod,
+                               xlevels = list(age = seq(15, 45, 5)))
> plot(arrests.effects,
+      ylab = "Probability(released)", ci.style = "bands", ask = FALSE)
```

The result, shown in Figure 7.20, is a relatively compact and understandable summary of the `arrests.mod` model: (a) people were more likely to be released if they were employed and citizens; (b) each additional police check decreased the likelihood of release with a summons; (c) the effect of skin color varied with age and year of arrest, in ways that tell a far more nuanced story than reported in the newspaper.

Finally, another feature of this plot bears mention: by default, the scales for each effect plot are determined separately for each effect, to maximize use of the plot region. However, you have to read the Y scale values to judge the relative sizes of these effects. An alternative plot, using the *same* scale in each subplot,[16] would show the relative sizes of these effects.

△

7.4.2 More complex models: Model selection and visualization

Models with more predictors or more complex terms (interactions, nonlinear terms) present additional challenges for model fitting, summarization, and visualization and interpretation. These problems increase rapidly with the number of potential predictors.

[16] With the effects package, you can set the `ylim` argument to equate the vertical range for all plots, but this should be done on the logit scale. For this plot, `ylim = plogis(c(0.5, 1))` would work.

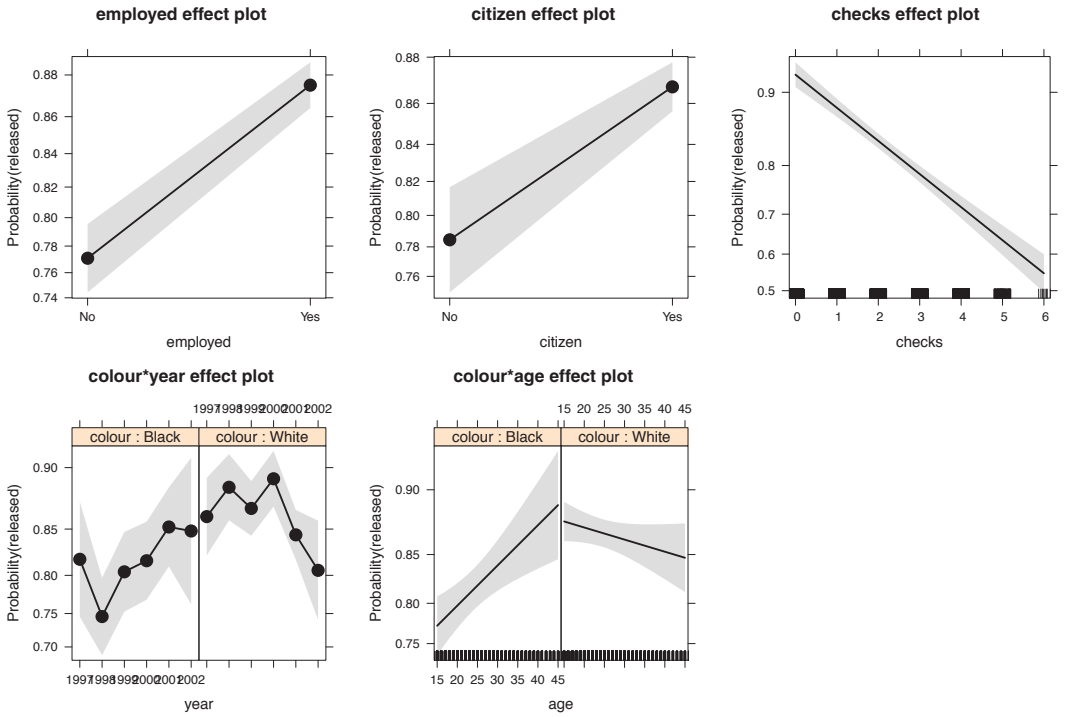


Figure 7.20: Effect plot for all high-order terms in the model for the Arrests data.

A very complicated model, with many terms and interactions, may fit the data at hand quite well. However, because goodness-of-fit is optimized in the sample, terms that appear significant are less likely to be important in a future sample, and we need to worry about inflation of Type I error rates that accompany multiple significance tests. As well, it becomes increasingly difficult to visualize and understand a fitted model as the model becomes increasingly complex. On the other hand, a very simple model may omit important predictors, interactions, or nonlinear relationships with the response and give an illusion of a comfortable interpretation.

Model selection for logistic regression seeks to balance the trade-off between the competing goals of goodness-of-fit and simplicity. A full discussion of this topic is beyond the scope of this book, but is well treated in Agresti (2013, Chapter 6), and extensively in Harrell (2001, Chapters 10–13).

Here, we illustrate some important ideas using the AIC and BIC statistics as parsimony-adjusted measures of goodness-of-fit. These are discussed Section 9.3.2. AIC is defined as

$$\text{AIC} = -2 \log \mathcal{L} + 2k$$

where $\log \mathcal{L}$ is the maximized log likelihood and k is the number of parameters estimated in the model. Better models correspond to *smaller* AIC. BIC is similar, but uses a penalty of $\log(n)k$, and so prefers smaller models as the sample size n increases.

EXAMPLE 7.11: Death in the ICU

In this example we briefly examine some aspects of logistic regression related to model selection and graphical display with a large collection of potential predictors, including both quantitative and discrete variables. We use data from a classic study by Lemeshow et al. (1988) of patients admitted to an intensive care unit at Baystate Medical Center in Springfield, Massachusetts. The major goal of this study was to develop a model to predict the probability of survival (until hospital discharge)

of these patients and to study the risk factors associated with ICU mortality. The data, contained in the data set *ICU* in vcdExtra, gives the results for a sample of 200 patients that was presented in Hosmer et al. (2013) (and earlier editions).

The *ICU* data set contains 22 variables of which the first, `died`, is a factor. Among the predictors, two variables (`race`, `coma`) were represented initially as 3-level factors, but then recoded to binary variables (`white`, `uncons`).

```
> data("ICU", package = "vcdExtra")
> names(ICU)

 [1] "died"      "age"       "sex"       "race"      "service"
 [6] "cancer"    "renal"     "infect"    "cpr"       "systolic"
[11] "hrtrate"   "previcu"   "admit"     "fracture"  "po2"
[16] "ph"        "pco"       "bic"       "creatin"   "coma"
[21] "white"     "uncons"

> ICU <- ICU[,-c(4, 20)]   # remove redundant race, coma
```

Removing the 3-level versions leaves 19 predictors, of which three (age, heart rate, systolic blood pressure) are quantitative and the remainder are either binary (service, cancer) or had previously been dichotomized (e.g., `ph<7.25`).

As an initial step, and a basis for comparison, we fit the full model containing all 19 predictors.

```
> icu.full <- glm(died ~ ., data = ICU, family = binomial)
> summary(icu.full)


Call:
glm(formula = died ~ ., family = binomial, data = ICU)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.8040  -0.5606  -0.2044  -0.0863   2.9773

Coefficients:
                Estimate Std. Error z value Pr(>|z|)
(Intercept)     -6.72670    2.38551   -2.82   0.0048 **
age              0.05639    0.01862    3.03   0.0025 **
sexMale          0.63973    0.53139    1.20   0.2286
serviceSurgical -0.67352    0.60190   -1.12   0.2631
cancerYes        3.10705    1.04585    2.97   0.0030 **
renalYes        -0.03571    0.80165   -0.04   0.9645
infectYes       -0.20493    0.55319   -0.37   0.7110
cprYes           1.05348    1.00661    1.05   0.2953
systolic        -0.01547    0.00850   -1.82   0.0686 .
hrtrate         -0.00277    0.00961   -0.29   0.7732
previcuYes       1.13194    0.67145    1.69   0.0918 .
admitEmergency   3.07958    1.08158    2.85   0.0044 **
fractureYes      1.41140    1.02971    1.37   0.1705
po2<=60          0.07382    0.85704    0.09   0.9314
ph<7.25          2.35408    1.20880    1.95   0.0515 .
pco>45          -3.01844    1.25345   -2.41   0.0160 *
bic<18          -0.70928    0.90978   -0.78   0.4356
creatin>2        0.29514    1.11693    0.26   0.7916
whiteNon-white   0.56573    0.92683    0.61   0.5416
unconsYes        5.23229    1.22630    4.27    2e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)
```

```
    Null deviance: 200.16  on 199  degrees of freedom
Residual deviance: 120.78  on 180  degrees of freedom
AIC: 160.8

Number of Fisher Scoring iterations: 6
```

You can see that a few predictors are individually significant, but many are not.

However, it is useful to carry out a simultaneous global test of $H_0 : \boldsymbol{\beta} = 0$ that *all* regression coefficients are zero. If this test is not significant, it makes little sense to use selection methods to choose individually significant predictors. For convenience, we define a simple function, `LRtest()`, to calculate the likelihood ratio test from the model components.

```
> LRtest <- function(model)
+   c(LRchisq = (model$null.deviance - model$deviance),
+     df = (model$df.null - model$df.residual))
>
> (LR <- LRtest(icu.full))

LRchisq      df
 79.383  19.000

> (pvalue <- 1 - pchisq(LR[1], LR[2]))

   LRchisq
2.3754e-09
```

At this point, it is tempting to examine the output from `summary(icu.full)` shown above and eliminate those predictors that fail significance at some specified level such as the conventional $\alpha = 0.05$. This is generally a bad idea for many reasons.[17]

A marginally better approach is to remove non-significant variables whose coefficients have signs that don't make sense from the substance of the problem. For example, in the full model, both `renal` (history of chronic renal failure) and `infect` (infection probable at ICU admission) have negative signs, meaning that their presence *decreases* the odds of death. We remove those variables using `update()`; as expected they make little difference.

```
> icu.full1 <- update(icu.full, . ~ . - renal - fracture)
> anova(icu.full1, icu.full, test = "Chisq")

Analysis of Deviance Table

Model 1: died ~ age + sex + service + cancer + infect + cpr + systolic +
    hrtrate + previcu + admit + po2 + ph + pco + bic + creatin +
    white + uncons
Model 2: died ~ age + sex + service + cancer + renal + infect + cpr +
    systolic + hrtrate + previcu + admit + fracture + po2 + ph +
    pco + bic + creatin + white + uncons
  Resid. Df Resid. Dev Df Deviance Pr(>Chi)
1       182        122
2       180        121  2      1.7     0.43
```

Before proceeding to consider model selection, it is useful to get a better visual overview of the current model than is available from a table of coefficients and significance tests. Some very useful `print()`, `summary()` and `plot()` methods are available in the rsm (Lenth, 2014) package. Unfortunately, these require that the logistic model is fitted with `lrm()` in that package rather than with `glm()`. We pause here to refit the same model as `icu.full1` in order to show a plot of odds ratios for the terms in this model.

[17] It ignores the facts of (a) an arbitrary cutoff value for significance, (b) the strong likelihood that chance features of the data or outliers influence the result, and (c) problems of collinearity, etc. See Harrell (2001, Section 4.3) for a useful discussion of these issues.

```
> library(rms)
> dd <- datadist(ICU[,-1])
> options(datadist = "dd")
> icu.lrm1 <- lrm(died ~ ., data = ICU)
> icu.lrm1 <- update(icu.lrm1, . ~ . - renal - fracture)
```

The `summary()` method for "rms" objects produces a much more detailed descriptive summary of a fitted model, and the `plot()` method for that summary object gives a sensible plot of the odds ratios for the model terms together with confidence intervals, at levels (0.9, 0.95, 0.99) by default. The following lines produce Figure 7.21.

```
> sum.lrm1 <- summary(icu.lrm1)
> plot(sum.lrm1, log = TRUE, main = "Odds ratio for 'died'", cex = 1.25,
+      col = rgb(0.1, 0.1, 0.8, alpha = c(0.3, 0.5, 0.8)))
```

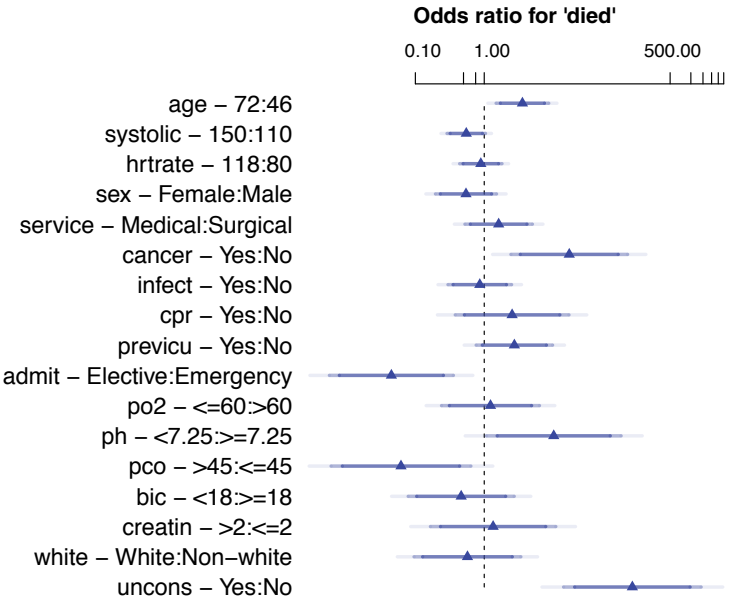


Figure 7.21: Odds ratios for the terms in the model for the ICU data. Each line shows the odds ratio for a term, together with lines for 90, 95, and 99% confidence intervals in progressively darker shades.

In this plot, continuous variables are shown at the top, followed by the discrete predictors. In each line, the range or levels of the predictors are given in the form $a : b$, such that the value a corresponds to the numerator of the odds ratio plotted. Confidence intervals that don't overlap the vertical line for odds ratio = 1 are significant, but this graph shows those at several confidence levels, allowing you to decide what is "significant" visually. As well, the widths of those intervals convey the precision of these estimates.

Among several stepwise selection methods in R for "glm" models, `stepAIC()` in the MASS package implements a reasonable collection of methods for forward, backward, and stepwise selection using penalized AIC-like criteria that balance goodness of fit against parsimony. The method takes an argument, `scope`, which is a list of two model formulae: `upper` defines the largest (most complex) model to consider and `lower` defines the smallest (simplest) model, e.g., `lower = ~ 1` is the intercept-only model.

By default, the function produces verbose printed output showing the details of each step, but we suppress that here to save space. It returns the final model as its result, along with an `anova` component that summarises the deviance and AIC from each step.

```
> library(MASS)
> icu.step1 <- stepAIC(icu.full1, trace = FALSE)
> icu.step1$anova

Stepwise Model Path
Analysis of Deviance Table

Initial Model:
died ~ age + sex + service + cancer + infect + cpr + systolic +
    hrtrate + previcu + admit + po2 + ph + pco + bic + creatin +
    white + uncons

Final Model:
died ~ age + cancer + systolic + admit + ph + pco + uncons


        Step Df Deviance Resid. Df Resid. Dev    AIC
1                              182     122.48 158.48
2      - po2  1 0.062446       183     122.54 156.54
3  - creatin  1 0.059080       184     122.60 154.60
4  - hrtrate  1 0.072371       185     122.67 152.67
5   - infect  1 0.122772       186     122.79 150.79
6    - white  1 0.334999       187     123.13 149.13
7  - service  1 0.671313       188     123.80 147.80
8      - bic  1 0.377521       189     124.18 146.18
9      - cpr  1 1.148260       190     125.33 145.33
10     - sex  1 1.543523       191     126.87 144.87
11 - previcu  1 1.569976       192     128.44 144.44
```

Alternatively, we can use the BIC criterion, by specifying $\mathrm{k}=\log(n)$, which generally will select a smaller model when the sample size is reasonably large.

```
> icu.step2 <- stepAIC(icu.full, trace = FALSE, k = log(200))
> icu.step2$anova

Stepwise Model Path
Analysis of Deviance Table

Initial Model:
died ~ age + sex + service + cancer + renal + infect + cpr +
    systolic + hrtrate + previcu + admit + fracture + po2 + ph +
    pco + bic + creatin + white + uncons

Final Model:
died ~ age + cancer + admit + uncons


         Step Df  Deviance Resid. Df Resid. Dev    AIC
1                              180     120.78 226.74
2     - renal  1 0.0019881       181     120.78 221.45
3       - po2  1 0.0067968       182     120.79 216.16
4   - creatin  1 0.0621463       183     120.85 210.92
5   - hrtrate  1 0.0658870       184     120.92 205.69
6    - infect  1 0.2033221       185     121.12 200.59
7     - white  1 0.3673180       186     121.49 195.66
8       - bic  1 0.6002993       187     122.09 190.96
9   - service  1 0.7676303       188     122.85 186.43
10 - fracture  1 1.3245086       189     124.18 182.46
11      - cpr  1 1.1482598       190     125.33 178.31
```

```
12      - sex  1 1.5435228       191     126.87 174.55
13  - previcu  1 1.5699762       192     128.44 170.83
14       - ph  1 4.4412370       193     132.88 169.97
15      - pco  1 2.7302934       194     135.61 167.40
16 - systolic  1 3.5231028       195     139.13 165.63
```

This model differs from model `icu.step1` selected using AIC in the last three steps, which also removed `ph`, `pco`, and `systolic`.

```
> coeftest(icu.step2)


z test of coefficients:

               Estimate Std. Error z value Pr(>|z|)
(Intercept)     -6.8698     1.3188   -5.21  1.9e-07 ***
age              0.0372     0.0128    2.91  0.00360 **
cancerYes        2.0971     0.8385    2.50  0.01238 *
admitEmergency   3.1022     0.9186    3.38  0.00073 ***
unconsYes        3.7055     0.8765    4.23  2.4e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

These two models are nested, so we can compare them directly using a likelihood ratio test from `anova()`.

```
> anova(icu.step2, icu.step1, test = "Chisq")

Analysis of Deviance Table

Model 1: died ~ age + cancer + admit + uncons
Model 2: died ~ age + cancer + systolic + admit + ph + pco + uncons
  Resid. Df Resid. Dev Df Deviance Pr(>Chi)
1       195        139
2       192        128  3     10.7    0.013 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The larger model is significantly better by this test, but the smaller model is simpler to interpret. We retain these both as "candidate models" to be explored further, but for ease in this example, we do so using the smaller model, `icu.step2`.

Another important step is to check for nonlinearity of quantitative predictors such as `age` and interactions among the predictors. This is easy to do using `update()` and `anova()` as shown below. First, allow a nonlinear term in `age`, and all two-way interactions of the binary predictors.

```
> icu.glm3 <- update(icu.step2, . ~ . - age + ns(age, 3) +
+                    (cancer + admit + uncons) ^ 2)
> anova(icu.step2, icu.glm3, test = "Chisq")

Analysis of Deviance Table

Model 1: died ~ age + cancer + admit + uncons
Model 2: died ~ cancer + admit + uncons + ns(age, 3) + cancer:admit +
    cancer:uncons + admit:uncons
  Resid. Df Resid. Dev Df Deviance Pr(>Chi)
1       195        139
2       191        135  4     3.73     0.44
```

Next, we can check for interactions with `age`:

```
> icu.glm4 <- update(icu.step2, . ~ . + age * (cancer + admit + uncons))
> anova(icu.step2, icu.glm4, test = "Chisq")

Analysis of Deviance Table

Model 1: died ~ age + cancer + admit + uncons
Model 2: died ~ age + cancer + admit + uncons + age:cancer + age:admit +
    age:uncons
  Resid. Df Resid. Dev Df Deviance Pr(>Chi)
1       195        139
2       192        134  3     5.37     0.15
```

None of these additional terms have much effect. △

So, we will tentatively adopt the simple main effects model, `icu.step2`, and consider how to visualize and interpret this result.

EXAMPLE 7.12: Death in the ICU — Visualization

One interesting display is a ***nomogram*** that shows how values on the various predictors translate into a predicted value of the log odds, and the relative strengths of their effects on this prediction. This kind of plot is shown in Figure 7.22, and is produced using `nomogram()` in the rms (Harrell, Jr., 2015) package as follows. This only works with models fit using `lrm()`, so we have to refit this model.

```
> icu.lrm2 <- lrm(died ~ age + cancer  + admit + uncons, data = ICU)
> plot(nomogram(icu.lrm2), cex.var = 1.2, lplabel = "Log odds death")
```

In this nomogram, each predictor is scaled according to the size of its effect on a common scale of 0–100 "points." A representative observation is shown by the marked points, corresponding to a person of age 60, without cancer, who was admitted to emergency and was unconscious at that time. Adding the points associated with each variable value gives the result shown on the scale of total points. For this observation, the result is $50 + 0 + 84 + 100 = 234$, for which the scale of log odds at the bottom gives a predicted logit of 2.2, or a predicted probability of death of $1/(1 + \exp(-2.2)) = 0.90$.

This leaves us with the problem of how to visualize the fitted model compactly and comprehensively. Multi-panel full-model plots and effect plots, as we have used them, are somewhat unwieldy with four or more predictors if we want to view all effects simultaneously, because it becomes more difficult to make comparisons across multiple panels (particularly if the vertical scales differ).

One way to reduce the visual complexity of such graphs is to combine some predictors that would otherwise be shown in separate panels into a recoding that can be shown as multiple curves for their combinations in fewer panels. In general, this can be done by combining some predictors *interactively*; for example, with sex and education as factors, their combinations, `M:Hi`, `M:Lo`, etc., could be used to define a new variable, `group`, used as the curves in one plot, rather than separate panels. This, in fact, is precisely what `binreg_plot()` does when there are two or more factors to be shown in a given plot.

In this case, because age is continuous, it makes sense to plot fitted values against age.[18] With `cancer`, `admit`, and `uncons` as binary factors associated with risk of death, it is also convenient for plotting to represent them in a way that reflects the level assiciated with higher risk. We do this by recoding their levels using `"-"` for low risk.

[18] By default, `binreg_plot()` uses the first numeric predictor as the horizontal variable.

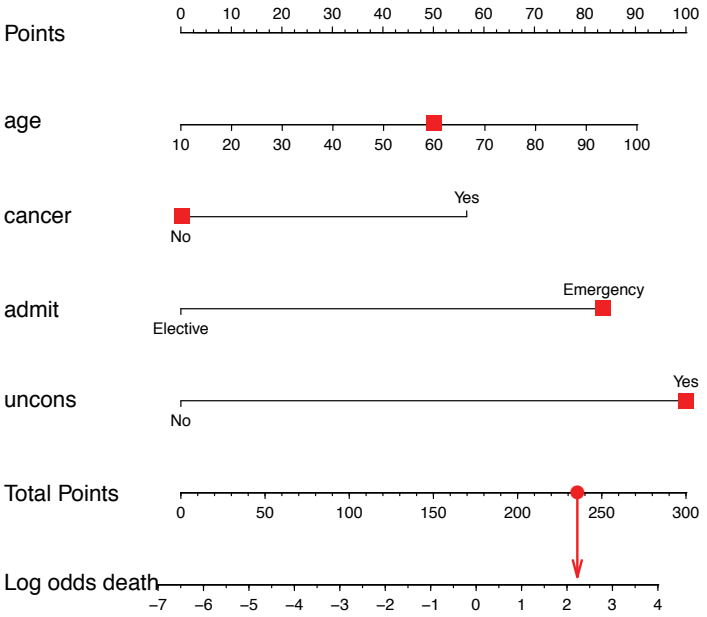


Figure 7.22: Nomogram for predicted values in the simple main effects model for the ICU data. Each predictor is scaled in relation to its effect on the outcome in terms of "points," 0–100. Adding the points for a given case gives total points that have a direct translation to log odds. The marked points show the prediction for someone of age 60, admitted to the emergency ward and unconscious.

```
> levels(ICU$cancer) <- c("-", "Cancer")
> levels(ICU$admit) <- c("-","Emerg")
> levels(ICU$uncons) <- c("-","Uncons")
>
> icu.glm2 <- glm(died ~ age + cancer + admit + uncons,
+                 data = ICU, family = binomial)
```

Then, `binreg_plot()` is called as follows, giving the plot shown in Figure 7.23. Such multiline graphs are more easily read with direct labels on the lines rather than a legend, so the legend is suppressed, and the lines are labeled using `labels = TRUE`. Points along the fitted lines are shown when `point_size>0`.

```
> binreg_plot(icu.glm2, type = "link", conf_level = 0.68,
+             legend = FALSE,
+             labels = TRUE, labels_just = c("right", "bottom"),
+             cex = 0, point_size = 0.8, pch = 15:17,
+             ylab = "Log odds (died)",
+             ylim = c(-7, 4))
```

From Figure 7.23, it is apparent that the log odds of mortality increases with age in all cases. Relative to the line labeled `"-:-:-"` (no risk factors), mortality is higher when any of these risk factors are present, particularly when the patient is admitted to emergency; it is highest when the patient is also unconscious at admission. The vertical gaps between lines that share a common risk (e.g., `Cancer`, `CancerEmerg`) indicate the additional increment from one more risk.

Finally, the plotted points show the number and age distribution of these various combinations. The greatest number of patients have only `Emerg` as a risk factor and only one patient was unconscious with no other risk.

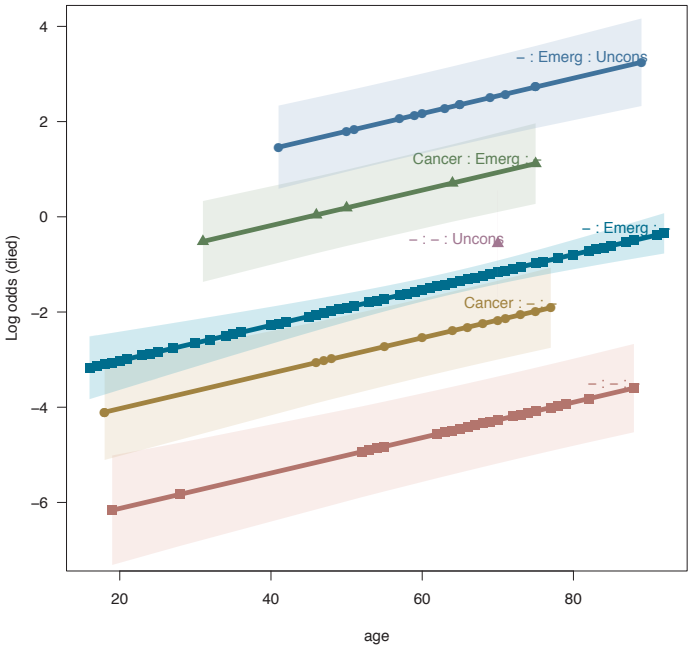


Figure 7.23: Fitted log odds of death in the ICU data for the model `icu.glm2`. Each line shows the relationship with age, for patients having various combinations of risk factors and 1 standard error confidence bands.

Before concluding that this model provides an adequate description of the data, we should examine whether any individual cases are unduly influencing the predicted results, and more importantly, the choice of variables in the model. We examine this question in Section 7.5 where we return to these data (Example 7.14).

△

7.5 Influence and diagnostic plots

In ordinary least squares (OLS) regression, measures of ***influence*** (leverage, Cook's D, DFBETAs, etc.) and associated plots help you to determine whether individual cases (or cells in grouped data) have undue impact on the fitted regression model and the coefficients of individual predictors. Analogs of most of these measures have been suggested for logistic regression and generalized linear models. Pregibon (1981) provided the theoretical basis for these methods, exploiting the relationship between logistic models and weighted least squares. Some additional problems occur in practical applications to logistic regression because the response is discrete, and because the leave-one-out diagnostics are more difficult to compute, but the ideas are essentially the same.

7.5.1 Residuals and leverage

As in ordinary least squares regression, the influence (actual impact) of an observation in logistic models depends multiplicatively on its residual (disagreement between y_i and $\hat{y}_i$) and its leverage (how unusual $\boldsymbol{x}_i$ is in the space of the explanatory variables). A conceptual formula is

$$\text{Influence} = \text{Leverage} \times \text{Residual}$$

This multiplicative definition implies that a case is influential to the extent that it is both poorly fit *and* has unusual values of the predictors.

7.5.1.1 Residuals

In logistic regression, the simple raw residual is just $e_i \equiv y_i - \hat{p}_i$, where $\hat{p}_i = 1/[1 + \exp(-\boldsymbol{x}_i^\mathsf{T}\boldsymbol{b})]$.

The Pearson and deviance residuals are more useful for identifying poorly fitted observations, and are components of overall goodness-of-fit statistics. The (raw) ***Pearson residual*** is defined as

$$r_i \equiv \frac{e_i}{\sqrt{\widehat{p}_i(1-\widehat{p}_i)}} \tag{7.7}$$

and the Pearson chi-square is therefore $\chi^2 = \sum r_i^2$. The ***deviance residual*** is

$$g_i \equiv \pm -2[y_i \log \widehat{p}_i + (1-y_i)\log(1-\widehat{p}_i)]^{1/2} \tag{7.8}$$

where the sign of g_i is the same as that of e_i. Likewise, the sum of squares of the deviance residuals gives the overall deviance, $G^2 = -2\log \mathcal{L}(\boldsymbol{b}) = \sum g_i^2$.

When y_i is a binomial count based on n_i trials (grouped data), the Pearson residuals Eqn. (7.7) then become

$$r_i \equiv \frac{y_i - n_i\widehat{p}_i}{\sqrt{n_i\widehat{p}_i(1-\widehat{p}_i)}}$$

with similar modifications made to Eqn. (7.8).

In R, `residuals()` is the generic function for obtaining (raw) residuals from a model fitted with `glm()` (or `lm()`). However, ***standardized residuals***, given by `rstandard()`, and ***studentized residual***s, provided by `rstudent()`, are often more useful because they rescale the residuals to have unit variance. They use, respectively, an overall estimate, $\hat{\sigma}^2$, of error variance, and the leave-one-out estimate, $\hat{\sigma}^2_{(-i)}$, omitting the ith observation; the studentized version is usually to be preferred in model diagnostics because it also accounts for the impact of the observation on residual variance.

7.5.1.2 Leverage

Leverage measures the *potential* impact of an individual case on the results, which is directly proportional to how far an individual case is from the centroid in the space of the predictors. Leverage is defined as the diagonal elements, h_{ii}, of the "Hat" matrix, $\boldsymbol{H}$,

$$\boldsymbol{H} = \boldsymbol{X}^\star(\boldsymbol{X}^{\star\mathsf{T}}\boldsymbol{X}^\star)^{-1}\boldsymbol{X}^{\star\mathsf{T}},$$

where $\boldsymbol{X}^\star = \boldsymbol{V}^{1/2}\boldsymbol{X}$, and $\boldsymbol{V} = \text{diag}\,[\hat{\boldsymbol{p}}(1-\hat{\boldsymbol{p}})]$.

As in OLS, leverage values are between 0 and 1, and a leverage value, $h_{ii} > \{2 \text{ or } 3\}k/n$, is considered "large;" here, $k = p + 1$ is the number of coefficients including the intercept and n is the number of cases. In OLS, however, the hat values depend only on the Xs, whereas in logistic regression, they also depend on the dependent variable values and the fitted probabilities (through $\boldsymbol{V}$). As a result, an observation may be extremely unusual on the predictors, yet not have a large hat value, if the fitted probability is near 0 or 1. The function `hatvalues()` calculates these values for a fitted "glm" model object.

7.5.2 Influence diagnostics

Influence measures assess the effect that deleting an observation has on the regression parameters, fitted values, or the goodness-of-fit statistics. In OLS, these measures can be computed exactly from

a single regression. In logistic regression, the exact effect of deletion requires refitting the model with each observation deleted in turn, a time-intensive computation. Consequently, Pregibon (1981) showed how analogous deletion diagnostics may be approximated by performing one additional step of the iterative procedure. Most modern implementations of these methods for generalized linear models follow Williams (1987).

The simplest measure of influence of observation i is the standardized change in the coefficient for each variable due to omitting that observation, termed ***DFBETA***s. From the relation (Pregibon, 1981, p. 716)

$$\boldsymbol{b} - \boldsymbol{b}_{(-i)} = (\boldsymbol{X}^\mathsf{T}\boldsymbol{V}\boldsymbol{X})^{-1}\boldsymbol{x}_i(y_i - \widehat{p}_i)/(1 - h_{ii}) \; ,$$

the estimated standardized change in the coefficient for variable j is

$$\text{DFBETA}ij \equiv \frac{b_{(-i)j} - b_j}{\hat{\sigma}(b_j)} \; , \tag{7.9}$$

where $\hat{\sigma}(b_j)$ is the estimated standard error of b_j. With k regressors, there are $k + 1$ sets of DFBETAs, which makes their examination burdensome. Graphical displays ease this burden, as do various summary measures considered below.

The most widely used summary of the overall influence of observation i on the estimated regression coefficients is ***Cook's distance***, which measures the average squared distance between $\boldsymbol{b}$ for all the data and $\boldsymbol{b}_{(-i)}$ estimated without observation i. It is defined as

$$C_i \equiv (\boldsymbol{b} - \boldsymbol{b}_{(-i)})^\mathsf{T} \, \boldsymbol{X}^\mathsf{T}\boldsymbol{V}\boldsymbol{X} \, (\boldsymbol{b} - \boldsymbol{b}_{(-i)})/k\hat{\sigma}^2 \; .$$

However, Pregibon (1981) showed that C_i could be calculated simply as

$$C_i = \frac{r_i^2 h_{ii}}{k(1 - h_{ii})^2} \; , \tag{7.10}$$

where $r_i = y_i - \hat{p}_i/\sqrt{v_{ii}(1 - h_{ii})}$ is the ith standardized Pearson residual and v_{ii} is the ith diagonal element of $\boldsymbol{V}$. Rules of thumb for noticeably "large" values of Cook's D are only rough indicators, and designed so that only "noteworthy" observations are nominated as unusually influential. One common cutoff for an observation to be treated as influential is $C_i > 1$. Others refer the values of C_i to a χ^2_k or $F_{k,n-k}$ distribution.

Another commonly used summary statistic of overall influence is the ***DFFITS*** statistic, a standardized measure of the difference between the predicted value $\hat{y}_i$ using all the data and the predicted value $\hat{y}_{(-i)}$ calculated omitting the ith observation.

$$\text{DFFITS}_i = \frac{\hat{y}_i - \hat{y}_{(-i)}}{\hat{\sigma}_{(-i)}\sqrt{h_{ii}}} \; ,$$

where $\hat{\sigma}_{(-i)}$ is the estimated standard error with the ith observation deleted. For computation, DFFITS can be expressed in terms of the standardized Pearson residual and leverage as

$$\text{DFFITS}_i = r_i \sqrt{\frac{h_{ii}}{(1 - h_{ii})} \frac{v_{ii}}{v_{(-ii)}}} \; . \tag{7.11}$$

From Eqn. (7.10) and Eqn. (7.11), it can be shown that Cook's distance is nearly the square of DFFITS divided by k,

$$C_i = \frac{v^2_{(-ii)}}{v^2_{ii}} \frac{\text{DFFITS}_i^2}{k} \; . \tag{7.12}$$

Noteworthy values of DFFITS are often nominated by the rule-of-thumb $\text{DFFITS}_i > 2$ or $3\sqrt{k/n - k}$.

In R, these influence measures are calculated for a fitted "glm" model using `cooks.distance()` and `dffits()`. A convenience function, `influence.measures()` gives a tabular display showing the DFBETA$_{ij}$ for each model variable, DFFITS, Cook's distances and the diagonal elements of the hat matrix. Cases which are influential with respect to any of these measures are marked with an asterisk.[19]

Beyond printed output of these numerical summaries, plots of these measures can shed light on potential problems due to influential or other noteworthy cases. By highlighting them, such plots provide the opportunity to determine if and how any of these affect your conclusions, or to take some corrective action.

Basic diagnostic plots are provided by the `plot()` method for a "glm" model object. These are easy to do, but the results for discrete response data are often unsatisfactory. The car package contains a variety of enhanced and extended functions for model diagnostic plots. We illustrate some of these in the examples below.

EXAMPLE 7.13: Donner Party

This example re-visits the data on the Donner Party examined in Example 7.9. For illustrative purposes, we consider the influence measures and diagnostic plots for one specific model, the model `donner.mod3`, which included a quadratic effect of age and a main effect of sex, but no interaction.

Details of all the diagnostic measures for a given model, including the DFBETAs for individual coefficients, can be obtained using `influence.measures`. This can be useful for custom plots not provided elsewhere (see Example 7.14).

```
> infl <- influence.measures(donner.mod3)
> names(infl)

[1] "infmat" "is.inf" "call"
```

The `summary()` method for the "infl" object prints those observations considered noteworthy on one or more of these statistics, as indicated by a "$*$" next to the value.

```
> summary(infl)

Potentially influential observations of
 glm(formula = survived ~ poly(age, 2) + sex, family = binomial,      data = Donner) :

                    dfb.1_ dfb.p(,2)1 dfb.p(,2)2 dfb.sxMl dffit   cov.r   cook.d hat
Breen, Patrick        0.08   0.65       0.56       0.23     0.69_*  0.93    0.32   0.09
Donner, Elizabeth    -0.26  -0.34      -0.22       0.12    -0.40    1.15_*  0.03   0.14_*
Graves, Elizabeth C. -0.24  -0.37      -0.26       0.10    -0.42    1.20_*  0.03   0.16_*
```

The simplest overview of adequacy of a fitted model is provided by the `plot()` method for a "glm" (or "lm") object, which can produce up to six different diagnostic plots. Among them, we consider the residual-leverage graph (number 5) as being the most useful for assessing influential observations, plotting residuals against leverages. An extended version is produced by the function `influencePlot()` in the car package, which additionally uses the size (area) of the plotting symbol to also show the value of Cook's D as shown in Figure 7.24. Like other diagnostic plots in car, it is considerably more general than illustrated here, because it allows for different `id.methods` to label noteworthy points, including `id.method = "identify"` for interactive point identification by clicking with the mouse. The `id.n` argument works differently than with `plot()`, because it selects the most extreme `id.n` observations on *each* of the studentized residual, hat value, and Cook's D, and labels all of these.

[19] See `help(influence.measures)` for the description of all of these functions for residuals, leverage, and influence diagnostics in generalized linear models.

```
> op <- par(mar = c(5, 4, 1, 1) + .1, cex.lab = 1.2)
> res <- influencePlot(donner.mod3, id.col = "blue", scale = 8, id.n = 2)
> k <- length(coef(donner.mod3))
> n <- nrow(Donner)
> text(x = c(2, 3) * k / n, y = -1.8, c("2k/n", "3k/n"), cex = 1.2)
```

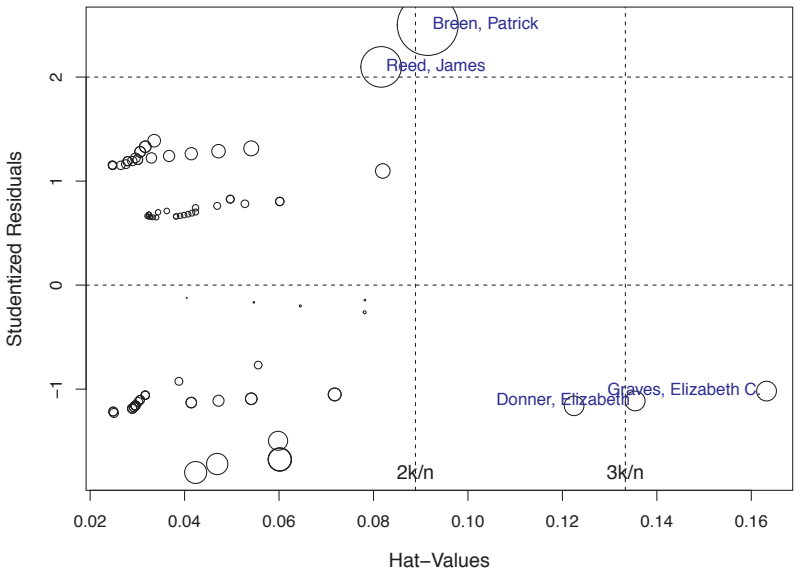


Figure 7.24: Influence plot (residual vs. leverage) for the Donner data model, showing Cook's D as the size of the bubble symbol. Horizontal and vertical reference lines show typical cutoff values for noteworthy residuals and leverage.

Conveniently, `influencePlot()` returns a data frame containing the influence statistics for the points identified in the plot (`res` in the call above). We can combine this with the data values to help learn why these points are considered influential.

```
> # show data together with diagnostics for influential cases
> idx <- which(rownames(Donner) %in% rownames(res))
> cbind(Donner[idx,2:4], res)

                     age    sex survived StudRes     Hat  CookD
Breen, Patrick        51   Male      yes   2.501 0.09148 0.5688
Donner, Elizabeth     45 Female       no  -1.114 0.13541 0.1846
Graves, Elizabeth C.  47 Female       no  -1.019 0.16322 0.1849
Reed, James           46   Male      yes   2.098 0.08162 0.3790
```

We can see that Patrick Breen and James Reed[20] are unusual because they were both older men who survived, and have large positive residuals; Breen is the most influential by Cook's D, but this value is not excessively large. The two women were among the older women who died. They are selected here because they have the largest hat values, meaning they are unusual in terms of the distribution of age and sex, but they are not particularly influential in terms of Cook's D.

A related graphical display is the collection of index plots provided by `influenceIndexPlot()` in car, which plots various influence diagnostics against the observation numbers in the data. The

[20]Breen and Reed, both born in Ireland, were the leaders of their family groups. Among others, both kept detailed diaries of their experiences, from which most of the historical record derives. Reed was also the leader of two relief parties sent out to find rescue or supplies over the high Sierra mountains, so it is all the more remarkable that he survived.

`id.n` argument here works to label that number of the most extreme observations *individually* for each measure plotted. The following call produces Figure 7.25.

```
> influenceIndexPlot(donner.mod3, vars=c("Cook", "Studentized", "hat"),
+                    id.n=4)
```

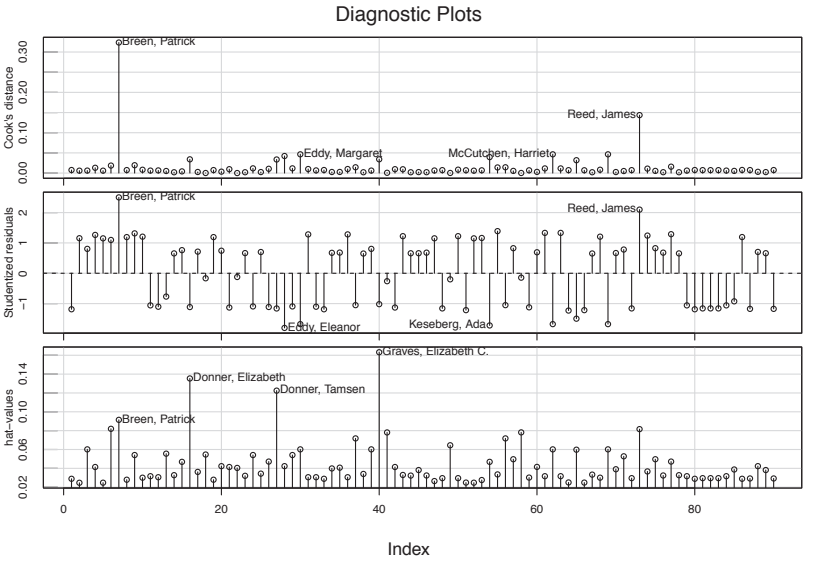


Figure 7.25: Index plots of influence measures for the Donner data model. The four most extreme observations on each measure are labeled.

In our opinion, *separate* index plots are often less useful than combined plots such as the leverage-influence plot that shows residuals, leverage and Cook's D together. However, the car version in Figure 7.25 does that too, and allows us to consider how unusual the labeled observations are both individually and in combination.

△

EXAMPLE 7.14: Death in the ICU

In Example 7.11 we examined several models to account for death in the *ICU* data set. We continue this analysis here, with a focus on the simple main effects model, `icu.glm2`, for which the fitted logits were shown in Figure 7.23. For ease of reference, we restate that model here:

```
> icu.glm2 <- glm(died ~ age + cancer  + admit + uncons,
+                 data = ICU, family = binomial)
```

The plot of residual vs. leverage for this model is shown in Figure 7.26.

```
> library(car)
> res <- influencePlot(icu.glm2, id.col = "red",
+                      scale = 8, id.cex = 1.5, id.n = 3)
```

Details for the cases identified in the figure are shown below, again using `rownames(res)` to select the relevant observations from the *ICU* data.

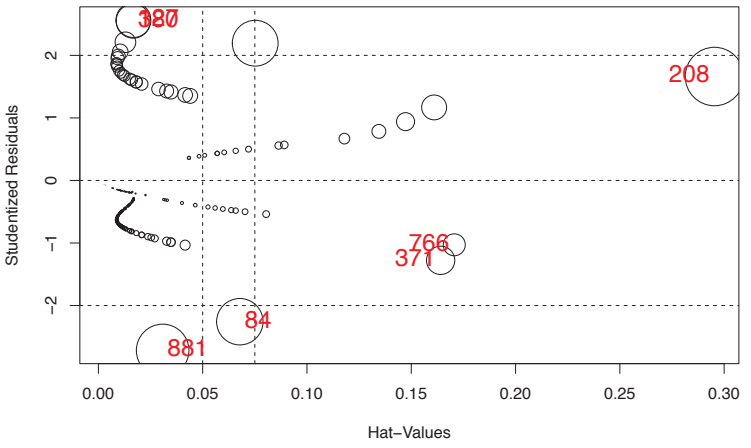


Figure 7.26: Influence plot for the main effects model for the ICU data.

```
> idx <- which(rownames(ICU) %in% rownames(res))
> cbind(ICU[idx, c("died", "age", "cancer", "admit", "uncons")], res)

    died age cancer admit uncons StudRes     Hat  CookD
84    No  59      - Emerg Uncons  -2.258 0.06781 0.3626
371   No  46 Cancer Emerg      -  -1.277 0.16408 0.2210
766   No  31 Cancer Emerg      -  -1.028 0.17062 0.1719
881   No  89      - Emerg Uncons  -2.718 0.03081 0.4106
127  Yes  19      - Emerg      -   2.565 0.01679 0.2724
208  Yes  70      -     - Uncons   1.662 0.29537 0.4568
380  Yes  20      - Emerg      -   2.548 0.01672 0.2668
```

None of the cases are particularly influential on the model coefficients overall: the largest Cook's D is only 0.45 for case 208. This observation also has the largest hat value. It is unusual on the predictors in this sample: a 70-year-old man without cancer, admitted on an elective basis, who nonetheless died. However, this case is also highly unusual in his having been unconscious on admission for an elective procedure, and signals that there might have been a coding error or other anomaly for this observation.

Another noteworthy observation identified here is case 881, an 89-year-old male, admitted unconscious as an emergency; this case is poorly predicted because he survived. Similarly, two other cases (127, 380) with large studentized residuals are poorly predicted because they died, although they were young, did not have cancer, and were conscious at admission. However, these cases have relatively small Cook's D values. From this evidence we might conclude that, case 208 bears further scrutiny, but none of these cases greatly affects the model, its coefficients, or interpretation.

For comparison with Figure 7.26, the related index plot of these measures is shown in Figure 7.27.

```
> influenceIndexPlot(icu.glm2, vars = c("Cook", "Studentized", "hat"),
+                    id.n = 4)
```

Cook's D and DFFITS are *overall* measures of the total influence that cases have on the regression coefficients and fitted values, respectively. It might be that some cases have a large impact on some individual regression coefficients, but don't appear particularly unusual in these aggregate measures.

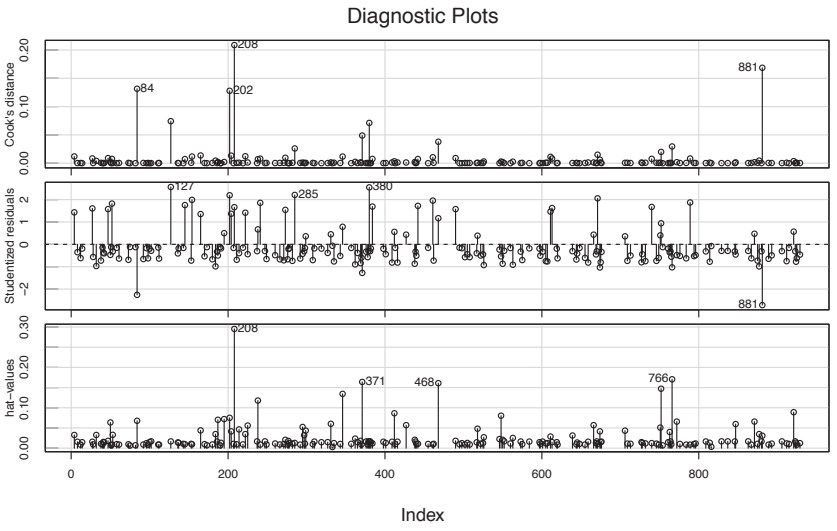


Figure 7.27: Index plots of influence measures for the ICU data model. The four most extreme observations on each measure are labeled.

One way to study this is to make plots of the DFBETA$_{ij}$ statistics. Such plots are not available (as far as we know) in R packages, but it is not hard to construct them from the result returned by `influence.measures()`. To do this, we select the appropriate columns from the `infmat` component returned by that function.

```
> infl <- influence.measures(icu.glm2)
> dfbetas <- data.frame(infl$infmat[,2:5])
> colnames(dfbetas) <- c("dfb.age", "dfb.cancer", "dfb.admit",
+                        "dfb.uncons")
> head(dfbetas)

     dfb.age dfb.cancer dfb.admit dfb.uncons
8   0.047340   0.013418  0.004067   0.009254
12  0.018988   0.018412 -0.004174   0.018106
14 -0.001051   0.014882  0.026278   0.005555
28  0.031562   0.018424 -0.001511   0.016640
32 -0.164084   0.003788 -0.036505   0.023488
38 -0.021525   0.016539 -0.011937   0.020803
```

To illustrate this idea, plotting an individual column of `dfbetas` using `type = "h"` gives an index plot against the observation number. This is shown in Figure 7.28 for the impact on the coefficient for age. The lines and points are colored blue or red according to whether the patient lived or died. Observations for which the $|\text{DFBETA}_{\text{age}}| > 0.2$ (an arbitrary value) are labeled.

```
> op <- par(mar = c(5, 5, 1, 1) + .1)
> cols <- ifelse(ICU$died == "Yes", "red", "blue")
> plot(dfbetas[,1], type = "h", col = cols,
+      xlab = "Observation index",
+      ylab = expression(Delta * beta[Age]),
+      cex.lab = 1.3)
> points(dfbetas[,1], col = cols)
> # label some points
> big <- abs(dfbetas[,1]) > .25
```

```
> idx <- 1 : nrow(dfbetas)
> text(idx[big], dfbetas[big, 1], label = rownames(dfbetas)[big],
+      cex = 0.9, pos = ifelse(dfbetas[big, 1] > 0, 3, 1),
+      xpd = TRUE)
> abline(h = c(-.25, 0, .25), col = "gray")
> par(op)
```

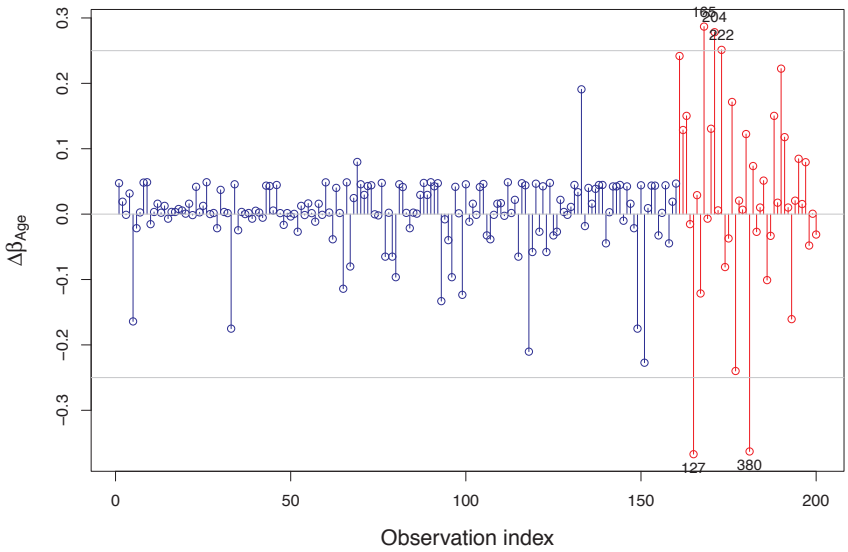


Figure 7.28: Index plot for DFBETA (Age) in the ICU data model. The observations are colored blue or red according to whether the patient lived or died.

None of the labeled points here are a cause for concern, since the standardized DFBETAs are all relatively small. However, the plot shows that patients who died have generally larger impacts on this coefficient.

An interesting alternative to individual index plots is a scatterplot matrix (Figure 7.29) that shows the pairwise changes in the regression coefficients for the various predictors. Here we use `scatterplotMatrix()` from car, which offers features for additional plot annotations, including identifying the most unusual points in each pairwise plot. In each off-diagonal panel, a 95% data ellipse and linear regression line help to show the marginal relationship between the two measures and highlight why the labeled points are atypical in each plot.[21]

```
> scatterplotMatrix(dfbetas, smooth = FALSE, id.n = 2,
+   ellipse = TRUE, levels = 0.95, robust = FALSE,
+   diagonal = "histogram",
+   groups = ICU$died, col = c("blue", "red"))
```

As Figure 7.29 illustrates, the *joint* effect of observations on *pairs* of coefficients is more complex than is apparent from the univariate views that appear in the plots along the diagonal. The DFBETAs for `cancer`, `admit`, and `uncons` are all extremely peaked, yet the pairwise plots show considerable structure. The points identified would be worthy of further study.

△

[21] This plot uses the `id.method = "mahal"` method to label the most extreme observations according to the Mahalanobis distance of each point from the centroid in the plot.

7.5.3 Other diagnostic plots*

The graphical methods described in this section are relatively straightforward indicators of the adequacy of a particular model, with a specified set of predictors, each expressed in a given way. More sophisticated methods have also been proposed, which focus on the need to include a particular predictor and whether its relationship is linear. These include the ***component-plus-residual plot***, the ***added-variable plot***, and the ***constructed variable plot***, which are all analogous to techniques developed in OLS.

7.5.3.1 Component-plus-residual plots

The ***component-plus-residual plot*** (also called a ***partial residual plot***) proposed originally by Larsen and McCleary (1972) is designed to show whether a given quantitative predictor, $\boldsymbol{x}_j$, included linearly in the model, actually shows a nonlinear relation, requiring transformation. The essential idea is to move the linear term for $\boldsymbol{x}_j$ back into the residual, by calculating the *partial residuals*,

$$\boldsymbol{r}_j^\star = \boldsymbol{r} + \beta_j \boldsymbol{x}_j \; .$$

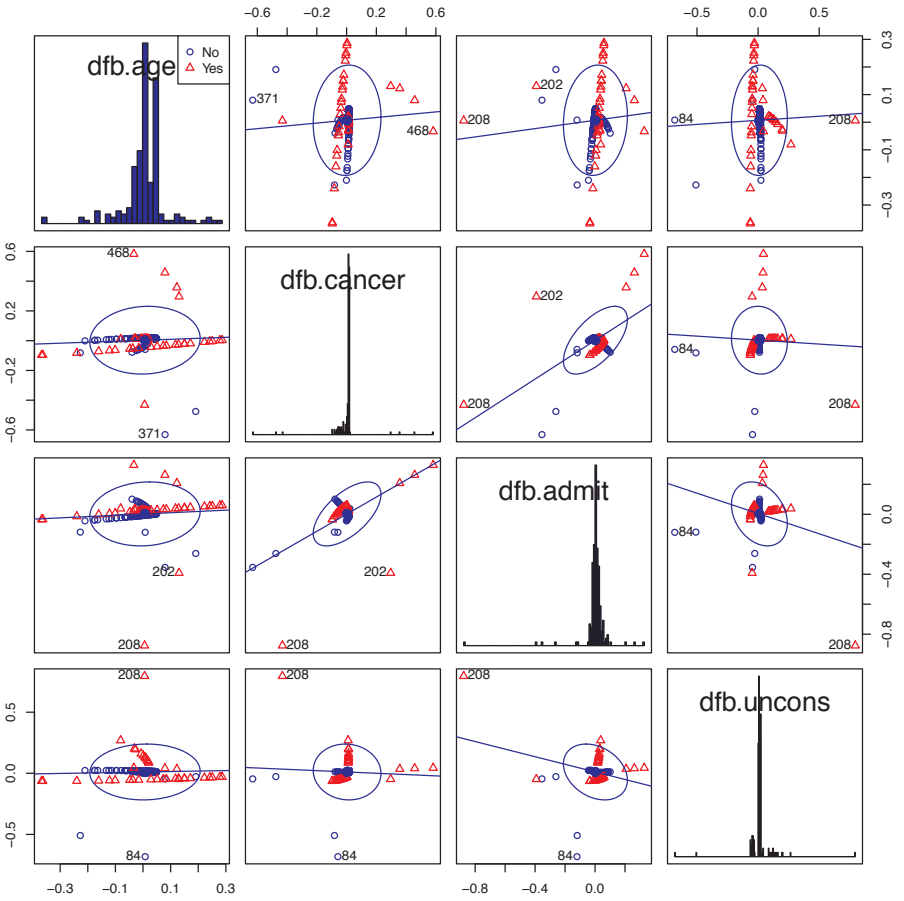


Figure 7.29: Scatterplot matrix for DFBETAs from the model for the ICU data. Those who lived or died are shown with blue circles and red triangles, respectively. The diagonal panels show histograms of each variable.

Then, a plot of $\boldsymbol{r}_j^\star$ against $\boldsymbol{x}_j$ will have the same slope, β_j, as the full model including it among other predictors. However, any nonlinear trend will be shown in the pattern of the points, usually aided by a smoothed non-parametric curve.

As adapted to logistic regression by Landwehr et al. (1984), the partial residual for variable $\boldsymbol{x}_j$ is defined as

$$\boldsymbol{r}_j^\star = \boldsymbol{V}^{-1}\boldsymbol{r} + \beta_j \boldsymbol{x}_j \; .$$

The partial residual plot is then a plot of $\boldsymbol{r}_j^\star$ against $\boldsymbol{x}_j$, possibly with the addition of a smoothed lowess curve (Fowlkes, 1987) and a linear regression line to aid interpretation. The linear regression of the partial residuals on $\boldsymbol{x}_j$ has the same slope, β_j, as in the full model.

If $\boldsymbol{x}_j$ affects the binary response linearly, the plot should be approximately linear with a slope approximately equal to β_j. A nonlinear plot suggests that x_j needs to be transformed, and the shape of the relation gives a rough guide to the required transformation. For example, a parabolic shape would suggest a term in $\boldsymbol{x}_j^2$. These plots complement the conditional data plots described earlier (Section 7.3.1), and are most useful when there are several quantitative predictors, so that it is more convenient and sensible to examine their relationships individually.

The car package implements these plots in the `crPlots()` and `crPlot()` functions. They also work for models with factor predictors (using parallel boxplots for the factor levels) but not for those with interaction terms.

EXAMPLE 7.15: Donner Party

In Example 7.13, we fit several models for the Donner Party data, and we recall two here to illustrate component-plus-residual plots. Both assert additive effects of age and sex, but the model `donner.mod3` allows a quadratic effect of age.

```
> donner.mod1 <- glm(survived ~ age + sex,
+                    data = Donner, family = binomial)
> donner.mod3 <- glm(survived ~ poly(age, 2) + sex,
+                    data = Donner, family = binomial)
```

Had we not made exploratory plots earlier (Example 7.13), and naively fit only the linear model in age, `donner.mod1`, we could use `crPlots()` to check for a nonlinear relationship of survival with age as follows, giving Figure 7.30.

```
> crPlots(donner.mod1, ~age, id.n=2)
```

The smoothed loess curve in this plot closely resembles the trend we saw in the conditional plot for age by sex (Figure 7.16), suggesting the need to include a nonlinear term for age. The points identified in this plot, by default, are those with either the most extreme x values (giving them high leverage) or the largest absolute Pearson residuals in the full model. The four structured bands of points in the plot correspond to the combinations of sex and survival.

For comparison, you can see the result of allowing for a nonlinear relationship in age in a partial residual plot for the model `donner.mod.3` that includes the effect `poly(age, 2)` for age. Note that the syntax of the `crPlots()` function requires that you specify a *term* in the model, rather than just a predictor variable.

```
> crPlots(donner.mod3, ~poly(age,2), id.n=2)
```

Except possibly at the extreme right, this plot (Figure 7.31) shows no indication of a (further) nonlinear relationship.

△

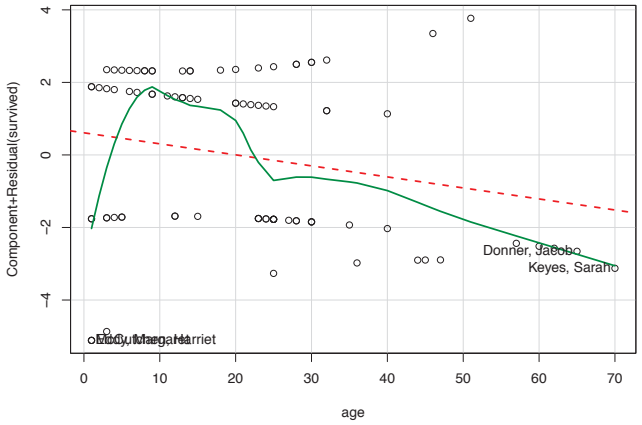


Figure 7.30: Component-plus-residual plot for the simple additive linear model, `donner.mod1`. The dashed red line shows the slope of age in the full model; the smoothed green curve shows a loess fit with span = 0.5.

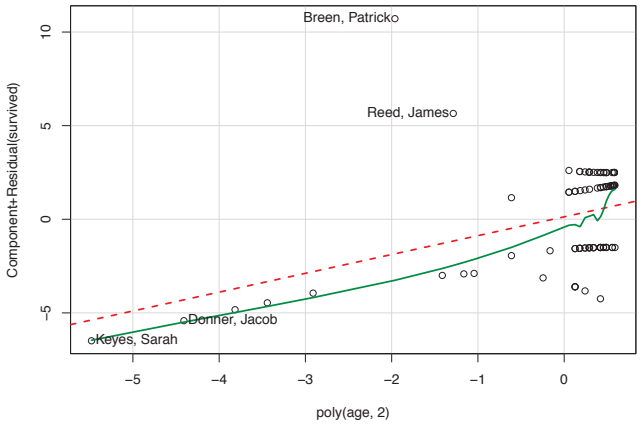


Figure 7.31: Component-plus-residual plot for the nonlinear additive model, `donner.mod3`.

7.5.3.2 Added-variable plots

Added-variable plots (Cook and Weisberg, 1999, Wang, 1985) (also called ***partial-regression plot***s) are another important tool for diagnosing problems in logistic regression and other linear or generalized linear models. These are essentially plots, for each $\boldsymbol{x}_i$, of an adjusted response, $\boldsymbol{y}_i^\star = \boldsymbol{y} \,|\, \text{others}_i$, against an adjusted predictor, $\boldsymbol{x}_i^\star = \boldsymbol{x}_i \,|\, \text{others}_i$, where $\text{others}_i = \boldsymbol{X} \notin \boldsymbol{x}_i \equiv \boldsymbol{X}^{(-i)}$ indicates all other predictors excluding $\boldsymbol{x}_i$. As such, they show the *conditional* relationship between the response and the predictor $\boldsymbol{x}_i$, controlling for, or adjusting for, all other predictors. Here, $\boldsymbol{y}_i^\star$ and $\boldsymbol{x}_i^\star$ represent, respectively, the residuals from the regressions of $\boldsymbol{y}$, and of $\boldsymbol{x}_i$, on all the other xs excluding $\boldsymbol{x}_i$.

It might seem from this description that each added-variable plot requires two additional auxiliary logistic regressions to calculate the residuals $\boldsymbol{y}_i^\star$ and $\boldsymbol{x}_i^\star$. However, Wang (1985) showed that the added-variable plot may be constructed by following the logistic regression for the model

$\boldsymbol{y} \sim \boldsymbol{X}^{(-i)}$ with one weighted least-squares regression of $\boldsymbol{x}_i$ on $\boldsymbol{X}^{(-i)}$ to find the residual part, $\boldsymbol{x}_i^\star$, of $\boldsymbol{x}$ not predicted by the other regressors.

Let $\boldsymbol{r}$ be the vector of Pearson residuals from the initial logistic fit of $\boldsymbol{y}$ on the variables in $\boldsymbol{X}^{(-i)}$, and let $\boldsymbol{H}$ and $\boldsymbol{V} = \text{diag}\,[\hat{\boldsymbol{p}}(1-\hat{\boldsymbol{p}})]$ be the hat matrix and $\boldsymbol{V}$ matrix from this analysis. Then, the added-variable plot is a scatterplot of the residuals $\boldsymbol{r}$ against the $\boldsymbol{x}_i$-residuals,

$$\boldsymbol{x}_i^\star = (\boldsymbol{I} - \boldsymbol{H})\boldsymbol{V}^{1/2}\boldsymbol{x}\,.$$

There are several important uses of added-variable plots:

First, *marginal* plots of the response variable $\boldsymbol{y}$ against the predictor variables $\boldsymbol{x}_i$ can conceal or misrepresent the relationships in a model including several predictors together due to correlations or associations among the predictors. This problem is compounded by the fact that graphical methods for discrete responses (boxplots, mosaic plots) cannot easily show influential observations or nonlinear relationships. Added-variable plots solve this problem by plotting the residuals, $\boldsymbol{y}_i^\star = \boldsymbol{y} \,|\, \text{others}_i$, which are less discrete than the marginal responses in $\boldsymbol{y}$.

Second, the numerical measures and graphical methods for detecting influential observations described earlier in this section are based on the idea of *single-case deletion*, comparing coefficients or fitted values for the full data with those that result from deleting each case in turn. Yet it is well-known (Lawrance, 1995) that sets of two (or more) observations can have ***joint influence***, which greatly exceeds their individual influential. Similarly, the influence of one discrepant point can be offset by another influential point in an opposite direction, a phenomenon called ***masking***. The main cases of joint influence are illustrated in Figure 7.32. Added-variable plots, showing the partial regression for one predictor controlling all others, can make such cases visually apparent.

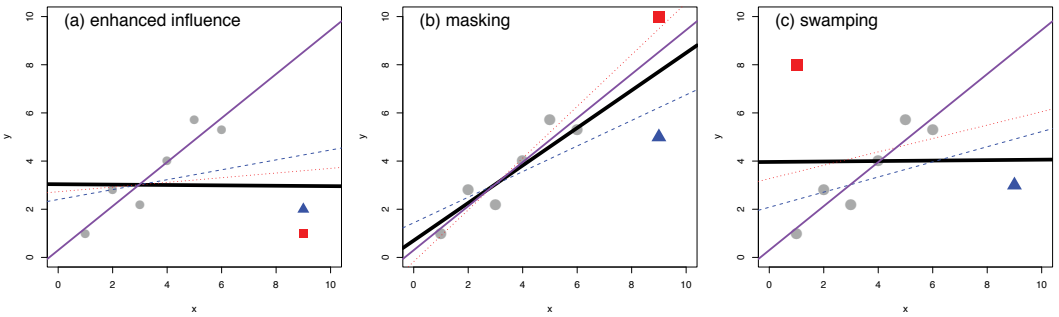


Figure 7.32: Jointly influential points in regression models. In each panel, the thick black line shows the regression of y on x using all the data points. The solid purple line shows the regression deleting *both* the red and blue points and the broken and dotted lines show the regression retaining only the point in its color in addition to the constant gray points. (a) Two points whose joint influence enhance each other; (b) two points where the influence of one is masked by that of the other; (c) two points whose combined influence greatly exceeds the effect of either one individually.

Finally, given a tentative model using predictors $\boldsymbol{x}$, the added-variable plot for another regressor, z, can provide a useful visual assessment of its additional contribution. An overall test could be based on the difference in G^2 for the enlarged model $\text{logit}(\boldsymbol{p}) = \boldsymbol{X}\boldsymbol{\beta} + \gamma\boldsymbol{z}$, compared to the reduced model $\text{logit}(\boldsymbol{p}) = \boldsymbol{X}\boldsymbol{\beta}$. But the added-variable plot shows whether the evidence for including z is spread throughout the sample or confined to a small subset of observations. The regressor z may be a new explanatory variable, or a higher-order term for variable(s) already in the model.

The car package implements these plots with the function `avPlot()` for a single term and `avPlots()` for all terms in a linear or generalized linear model, as shown in the example(s) below. See `http://www.datavis.ca/gallery/animation/duncanAV/` for an animated

graphic showing the transition between a marginal plot of the relationship of $\boldsymbol{y}$ to $\boldsymbol{x}$ and the added-variable plot of $\boldsymbol{y}^\star$ to $\boldsymbol{x}^\star$ for the case of multiple linear regression with a quantitative response.

EXAMPLE 7.16: Donner Party

The simple additive model `donner.mod1` for the Donner Party data can be used to illustrate some features of added-variable plots. In the call to `avPlots()` below, we use color for the plotting symbol to distinguish those who survived vs. died, shape to distinguish men from women.

```
> col <- ifelse(Donner$survived == "yes", "blue", "red")
> pch <- ifelse(Donner$sex == "Male", 16, 17)
> avPlots(donner.mod1, id.n = 2,
+         col = col, pch = pch, col.lines = "darkgreen")
```

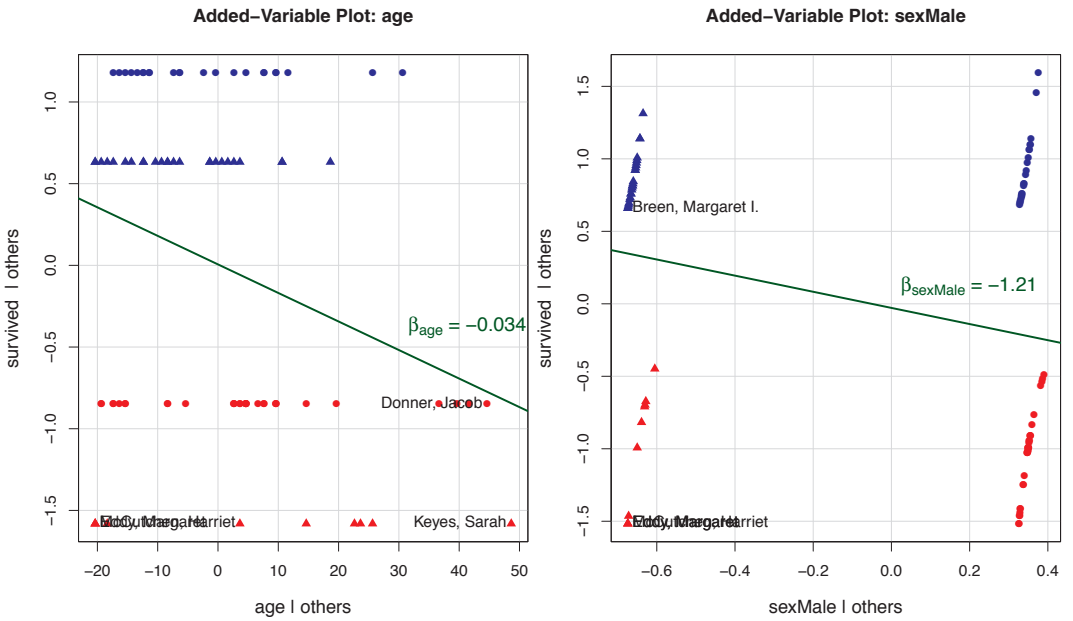


Figure 7.33: Added-variable plots for age (left) and sex (right) in the Donner Party main effects model. Those who survived are shown in blue; those who died in red. Men are plotted with filled circles; women with filled triangles.

These plots have the following properties:

1. The slope in the simple regression of $\boldsymbol{y}_i^\star$ on $\boldsymbol{x}_i^\star$ is the same as the partial coefficient β_i in the full multiple regression model including both predictors here (or all predictors in general).

2. The residuals from this regression line are the same as the residuals in the full model.

3. Because the response, `survived`, is binary, the vertical axis $\boldsymbol{y}^\star_{\text{age}}$ in the left panel for `age` is the part of the logit for survival that cannot be predicted from `sex`. Similarly, the vertical axis in the right panel is the part of survival that cannot be predicted from `age`. This property allows the clusters of points corresponding to discrete variables to be seen more readily, particularly if they are distinguished by visual attributes such as color and shape, as in Figure 7.33.

△

EXAMPLE 7.17: Death in the ICU

We illustrate some of the uses of added-variable plots using the main effects model, `icu.glm2`, predicting death in the ICU from the variables `age`, `cancer`, `admit`, and `uncons`.

To see why marginal plots of the discrete response against each predictor are often unrevealing for the purpose of model assessment, consider the collection of plots in Figure 7.34 showing the default plots (spineplots) for the factor response, `died`, against each predictor. These show the marginal distribution of each predictor by the widths of the bars, and highlight the proportion who died by color. Such plots are useful for some purposes, but not for assessing the adequacy of the fitted model.

```
> op <- par(mfrow = c(2, 2), mar = c(4, 4, 1, 2.5) + .1, cex.lab = 1.4)
> plot(died ~ age, data = ICU, col = c("lightblue", "pink"))
> plot(died ~ cancer, data = ICU, col = c("lightblue", "pink"))
> plot(died ~ admit, data = ICU, col = c("lightblue", "pink"))
> plot(died ~ uncons, data = ICU, col = c("lightblue", "pink"))
```

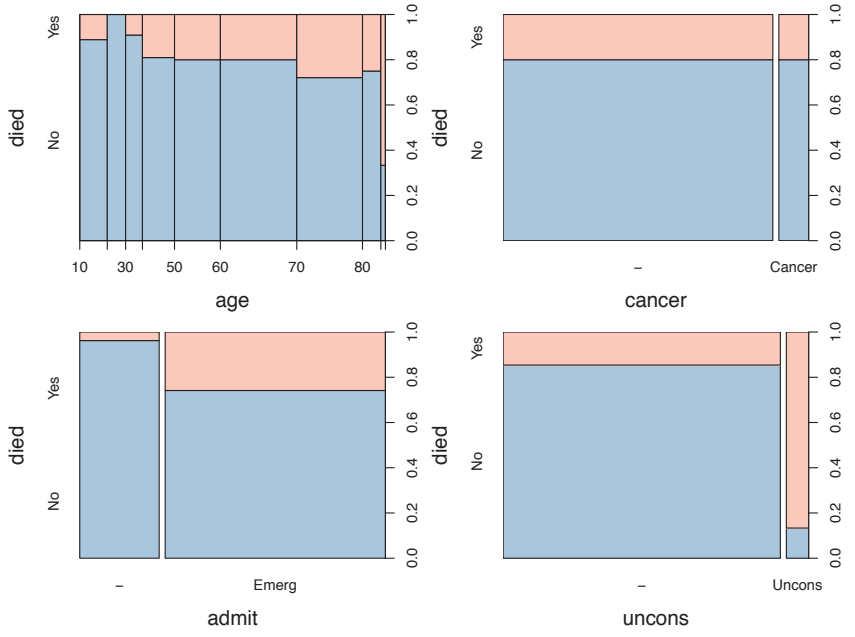


Figure 7.34: Marginal plots of the response `died` against each of the predictors in the model `icu.glm2` for the *ICU* data.

The added-variable plot for this model is shown in Figure 7.35. In each plot, the solid red line shows the partial slope, β_j for the focal predictor, controlling for all others.

```
> pch <- ifelse(ICU$died=="No", 1, 2)
> avPlots(icu.glm2, id.n=2, pch=pch, cex.lab=1.3)
```

The labeled points in each panel use the default `id.method` for `avPlots()`, selecting those with either large absolute model residuals or extreme $\boldsymbol{x}_i^\star$ residuals, given all other predictors. Cases 127 and 881, identified earlier as influential, stand out in all these plots.

Next, we illustrate the use of added-variable plots for checking the effect of influential observations on the decision to include an additional predictor in some given model. In the analysis of the *ICU* data using model selection methods, the variable `systolic` (systolic blood pressure at admission) was nominated by several different procedures. Here we take a closer look at the evidence

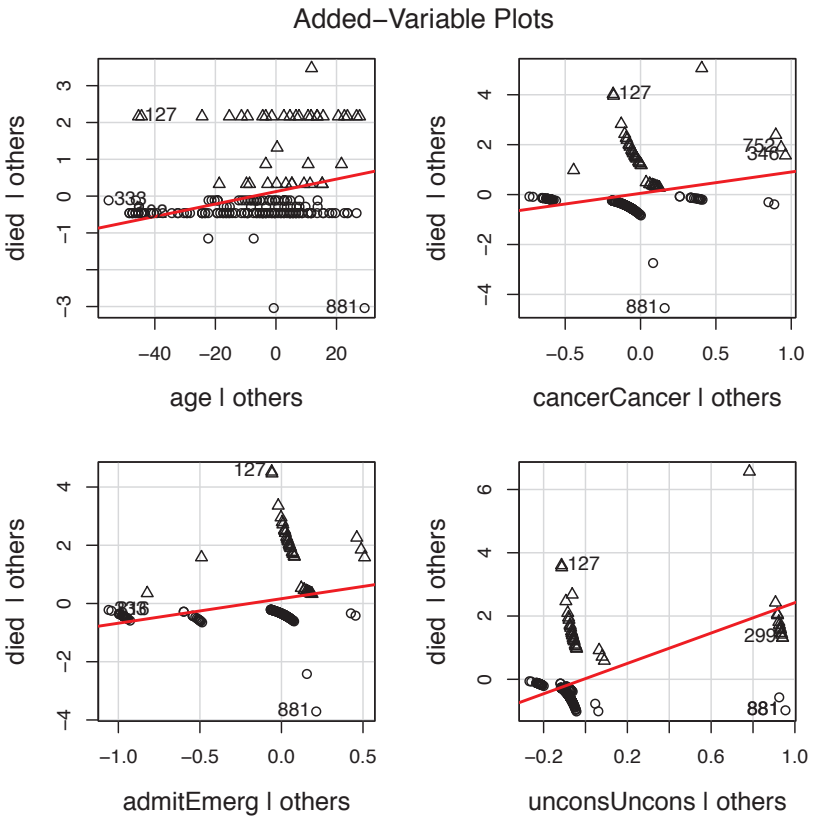


Figure 7.35: Added-variable plots for the predictors in the model for the ICU data. Those who died and survived are shown by triangles (△) and circles (◯), respectively.

for inclusion of this variable in a predictive model. We fit a new model adding `systolic` to the others and test the improvement with a likelihood ratio test:

```
> icu.glm2a <- glm(died ~ age + cancer  + admit + uncons + systolic,
+                  data = ICU, family = binomial)
> anova(icu.glm2, icu.glm2a, test = "Chisq")

Analysis of Deviance Table

Model 1: died ~ age + cancer + admit + uncons
Model 2: died ~ age + cancer + admit + uncons + systolic
  Resid. Df Resid. Dev Df Deviance Pr(>Chi)
1       195        139
2       194        136  1     3.52    0.061 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

So, the addition of systolic blood pressure is nearly significant at the conventional $\alpha = 0.05$ level. The added-variable plot for this variable in Figure 7.36 shows the strength of evidence for its contribution, above and beyond the other variables in the model, as well as the partial leverage and influence of individual points.

```
> avPlot(icu.glm2a, "systolic", id.n = 3, pch = pch)
```

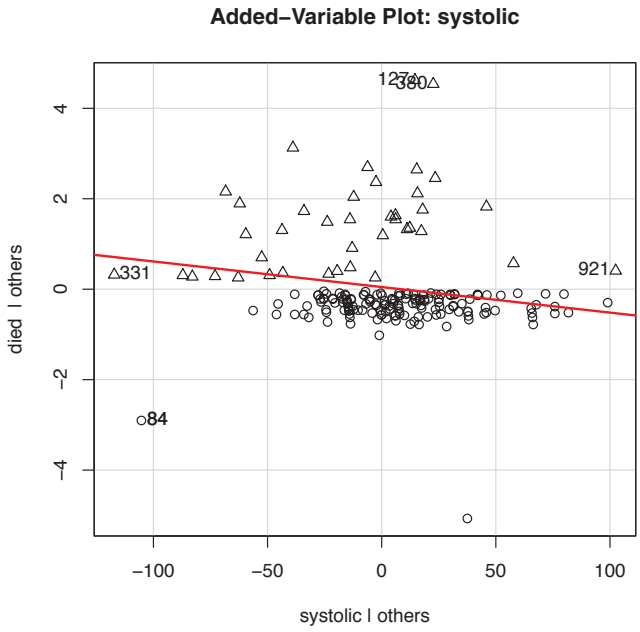


Figure 7.36: Added-variable plot for the effect of adding systolic blood pressure to the main effects model for the ICU data.

In this plot, cases 331 and 921 have high partial leverage, but they are not influential. Case 84, however, has high leverage and a large residual, so it is possibly influential on the evidence for inclusion of `systolic` in the model. Note also that the partial regression line in this plot nicely separates nearly all the patients who died from those who survived.

△

7.6 Chapter summary

- Model-based methods for categorical data provide confidence intervals for parameters and predicted values for observed and unobserved values of the explanatory variables. Graphical displays of predicted values help us to interpret the fitted relations by smoothing a discrete response.
- The logistic regression model (Section 7.2) describes the relationship between a categorical response variable, usually dichotomous, and a set of one or more quantitative or discrete explanatory variables (Section 7.3) It is conceptually convenient to specify this model as a linear model predicting the log odds (or logit) of the probability of a success from the explanatory variables.
- The relationship between a discrete response and a quantitative predictor may be explored graphically by plotting the binary observations against the predictor with some smoothed curve(s), either parametric or non-parametric, possibly stratified by other predictors.
- For both quantitative and discrete predictors, the results of a logistic regression are most easily interpreted from full-model plots of the fitted values against the predictors, either on the scale of

predicted probabilities or log odds (Section 7.3.2). In these plots, confidence intervals provide a visual indication of the precision of the predicted results.

- When there are multiple predictors and/or higher-order interaction terms, effect plots (Section 7.3.3) provide an important method for constructing simplified displays, focusing on the higher-order terms in a given model.
- Influence diagnostics (Section 7.5) assess the impact of individual cases or groups on the fitted model, predicted values, and the coefficients of individual predictors. Among other displays, plots of residuals against leverage showing Cook's D are often most useful.
- Other diagnostic plots (Section 7.5.3) include component-plus-residual plots, which are useful for detecting non-linear relationships for a quantitative predictor, and added-variable plots, which show the partial relations of the response to a given predictor, controlling or adjusting for all other predictors.

7.7 Lab exercises

Exercise 7.1 Arbuthnot's data on the sex ratio of births in London was examined in Example 3.1. Use a binomial logistic regression model to assess whether the proportion of male births varied with the variables `Year`, `Plague`, and `Mortality` in the *Arbuthnot* data set. Produce effect plots for the terms in this model. What do you conclude?

Exercise 7.2 For the Donner Party data in *Donner*, examine Grayson's 1990 claim that survival in the Donner Party was also mediated by the size of the family unit. This takes some care, because the `family` variable in the *Donner* data is a simplified grouping based on the person's name and known alliances among families from the historical record. Use the following code to compute a `family.size variable` from each individual's last name:

```
> data("Donner", package="vcdExtra")
> Donner$survived <- factor(Donner$survived, labels=c("no", "yes"))
> # use last name for family
> lame <- strsplit(rownames(Donner), ",")
> lame <- sapply(lame, function(x) x[[1]])
> Donner$family.size <- as.vector(table(lname)[lname])
```

(a) Choose one of the models (`donner.mod4`, `donner.mod6`) from Example 7.9 that include the interaction of age and sex and nonlinear terms in age. Fit a new model that adds a main effect of `family.size`. What do you conclude about Grayson's claim?
(b) Produce an effect plot for this model.
(c) Continue, by examining whether the effect of family size can be taken as linear, or whether a nonlinear term should be added.

Exercise 7.3 Use component+residual plots (Section 7.5.3) to examine the additive model for the *ICU* data given by

```
> icu.glm2 <- glm(died ~ age + cancer  + admit + uncons,
+                 data=ICU, family=binomial)
```

(a) What do you conclude about the linearity of the (partial) relationship between age and death in this model?

(b) An alternative strategy is to allow some nonlinear relation for age in the model using a quadratic (or cubic) term like `poly(age, 2)` (or `poly(age, 3)`) in the model formula. Do these models provide evidence for a nonlinear effect of age on death in the ICU?

Exercise 7.4 Explore the use of other marginal and conditional plots to display the relationships among the variables predicting death in the ICU in the model `icu.glm2`. For example, you might begin with a marginal `gpairs()` plot showing all bivariate marginal relations, something like this:

```
> library(gpairs)
> gpairs(ICU[,c("died", "age", "cancer", "admit", "uncons")],
+   diag.pars=list(fontsize=16, hist.color="lightgray"),
+   mosaic.pars=list(gp=shading_Friendly,
+                    gp_args=list(interpolate=1:4)))
```

Exercise 7.5 The data set *Caesar* in vcdExtra gives a 3×2^3 frequency table classifying 251 women who gave birth by Caesarian section by `Infection` (three levels: none, Type 1, Type2) and `Risk`, whether `Antibiotics` were used, and whether the Caesarian section was `Planned` or not. `Infection` is a natural response variable. In this exercise, consider only the binary outcome of infection vs. no infection.

```
> data("Caesar", package="vcdExtra")
> Caesar.df <- as.data.frame(Caesar)
> Caesar.df$Infect <- as.numeric(Caesar.df$Infection %in%
+                                  c("Type 1", "Type 2"))
```

(a) Fit the main-effects logit model for the binary response `Infect`. Note that with the data in the form of a frequency data frame you will need to use `weights=Freq` in the call to `glm()`. (It might also be convenient to reorder the levels of the factors so that `"No"` is the baseline level for each.)
(b) Use `summary()` or car::`Anova()` to test the terms in this model.
(c) Interpret the coefficients in the fitted model in terms of their effect on the odds of infection.
(d) Make one or more effects plots for this model, showing separate terms, or their combinations.

Exercise 7.6 The data set *birthwt* in the MASS package gives data on 189 babies born at Baystate Medical Center, Springfield, MA during 1986. The quantitative response is `bwt` (birth weight in grams), and this is also recorded as `low`, a binary variable corresponding to `bwt < 2500` (2.5 Kg). The goal is to study how this varies with the available predictor variables. The variables are all recorded as numeric, so in R it may be helpful to convert some of these into factors and possibly collapse some low frequency categories. The code below is just an example of how you might do this for some variables.

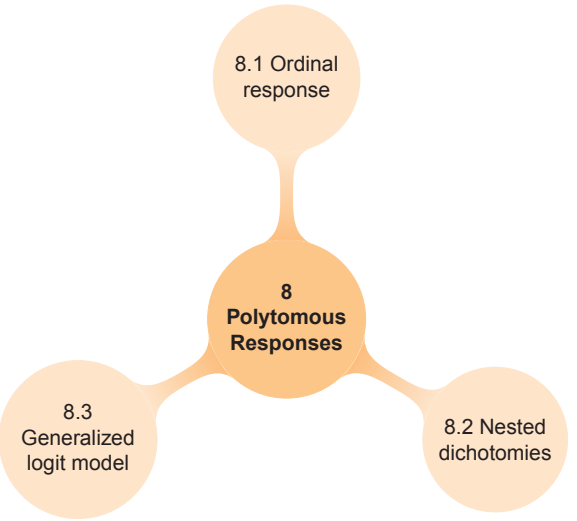
```
> data("birthwt", package="MASS")
> birthwt <- within(birthwt, {
+   race <- factor(race, labels = c("white", "black", "other"))
+         ptd <- factor(ptl > 0)   # premature labors
+         ftv <- factor(ftv)       # physician visits
+         levels(ftv)[-(1:2)] <- "2+"
+         smoke <- factor(smoke>0)
+         ht <- factor(ht>0)
+         ui <- factor(ui>0)
+   })
```

(a) Make some exploratory plots showing how low birth weight varies with each of the available predictors. In some cases, it will probably be helpful to add some sort of smoothed summary curves or lines.

(b) Fit several logistic regression models predicting low birth weight from these predictors, with the goal of explaining this phenomenon adequately, yet simply.
(c) Use some graphical displays to convey your findings.

Exercise 7.7 Refer to Exercise 5.9 for a description of the *Accident* data. The interest here is to model the probability that an accident resulted in death rather than injury from the predictors `age`, `mode`, and `gender`. With `glm()`, and the data in the form of a frequency table, you can use the argument `weight=Freq` to take cell frequency into account.

(a) Fit the main effects model, `result=="Died" ~ age + mode + gender`. Use car::`Anova()` to assess the model terms.
(b) Fit the model that allows all two-way interactions. Use `anova()` to test whether this model is significantly better than the main effects model.
(c) Fit the model that also allows the three-way interaction of all factors. Does this offer any improvement over the two-way model?
(d) Interpret the results of the analysis using effect plots for the two-way model, separately for each of the model terms. Describe verbally the nature of the `age*gender` effect. Which mode of transportation leads to greatest risk of death?



8

Models for Polytomous Responses

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

Ballerinas are often divided into three categories: jumpers, turners and balancers

Robert Gottlieb

Polytomous response data arise when the outcome variable, Y, takes on $m > 2$ discrete values. For example, (a) patients may record that their improvement after treatment is "none," "some" or "marked;" (b) high school students may choose a general, vocational, or academic program; (c) women's labor force participation may be recorded in a survey as not working outside the home, working part-time, or working full-time; (d) Canadian voters may express a preference for the Conservative, Liberal, NDP, or Green party. These response categories may be considered *ordered*, as in case (a), or simply *nominal*, as in case (d), and sometimes the response can arguably be treated in either way, as in cases (b) and (c).

In this situation, there are several different ways to model the response probabilities. Let $\pi_{ij} \equiv \pi_j(\boldsymbol{x}_i)$ be the probability of response j for case or group i, given the predictors $\boldsymbol{x}_i$. Because $\sum_j \pi_{ij} = 1$, only $m - 1$ of these probabilities are independent. The essential idea here is to construct a model for the polytomous (or multinomial) response composed of $m - 1$ logit comparisons among the response categories in a similar way to how factors are treated in the predictor variables.

The simplest approach uses the ***proportional odds model***, described in Section 8.1. This model applies *only* when the response is ordinal (as in improvement after therapy) *and* an additional assumption (the proportional odds assumption) holds. This model can be fit using `polr()` in the MASS package, `lrm()` in the rms package, and `vglm()` in VGAM (Yee, 2015).

However, if the response is purely nominal (e.g., vote Conservative, Liberal, NDP, Green), or if the proportional odds assumption is untenable, another particularly simple strategy is to fit separate models to a set of $m - 1$ ***nested dichotomies*** derived from the polytomous response (described in Section 8.2). This method allows you to resolve the differences among the m response categories into *independent* statistical questions (similar to orthogonal contrasts in ANOVA). For example, for women's labor force participation, it might be substantively interesting to contrast not working vs. part-time and full-time and then part-time vs. full-time for women who are working. You fit such nested dichotomies by running the $m - 1$ binary logit models and combining the statistical results.

The most general approach is the ***generalized logit model***, also called the ***multinomial logit model***, described in Section 8.3. This model fits *simultaneously* the $m - 1$ simple logit models against a baseline or reference category, for example, the last category, m. With a 3-category response, there are two generalized logits, $L_{i1} = \log(\pi_{i1}/\pi_{i3})$ and $L_{i2} = \log(\pi_{i2}/\pi_{i3})$, contrasting response categories 1 and 2 against category 3. In this approach, it doesn't matter which response category is chosen as the baseline, because all pairwise comparisons can be recovered from whatever is estimated. This model is conveniently fitted using `multinom()` in nnet.

8.1 Ordinal response: Proportional odds model

For an ordered response Y, with categories $j = 1, 2, \ldots, m$, the ordinal nature of the response can be taken into account by forming logits based on the $m - 1$ adjacent category cutpoints between successive categories. That is, if the cumulative probabilities are

$$\Pr(Y \leq j \,|\, \boldsymbol{x}) = \pi_1(\boldsymbol{x}) + \pi_2(\boldsymbol{x}) + \cdots \pi_j(\boldsymbol{x}) \,,$$

then the ***cumulative logit*** for category j is defined as

$$L_j \equiv \text{logit}[\Pr(Y \leq j \,|\, \boldsymbol{x})] = \log \frac{\Pr(Y \leq j \,|\, \boldsymbol{x})}{\Pr(Y > j \,|\, \boldsymbol{x})} = \log \frac{\Pr(Y \leq j \,|\, \boldsymbol{x})}{1 - \Pr(Y \leq j \,|\, \boldsymbol{x})} \tag{8.1}$$

for $j = 1, 2, \ldots m - 1$.

In our running example of responses to arthritis treatment, the actual response variable is `Improved`, with ordered levels `"None" < "Some" < "Marked"`. In this case, the cumulative logits would be defined as

$$L_1 = \log \frac{\pi_1(\boldsymbol{x})}{\pi_2(\boldsymbol{x}) + \pi_3(\boldsymbol{x})} = \text{logit (None vs. [Some or Marked])}$$

$$L_2 = \log \frac{\pi_1(\boldsymbol{x}) + \pi_2(\boldsymbol{x})}{\pi_3(\boldsymbol{x})} = \text{logit ([None or Some] vs. Marked) ,}$$

where $\boldsymbol{x}$ represents the predictors (sex, treatment and age).

The ***proportional odds model*** (PO) (McCullagh, 1980) proposes a simple and parsimonious account of these effects, where the predictors in (x) are constrained to have the same slopes for all cumulative logits,

$$L_j = \alpha_j + \boldsymbol{x}^\mathsf{T} \boldsymbol{\beta} \qquad j = 1, \ldots, m - 1 \,. \tag{8.2}$$

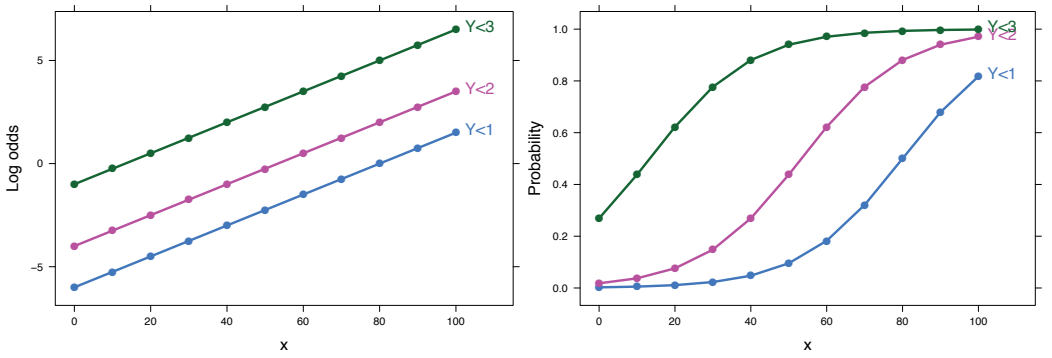


Figure 8.1: Proportional odds model for an ordinal response. The model assumes equal slopes for the cumulative response logits. Left: logit scale; right: probability scale.

That is, the effect of the predictor x_i is the same, β_i, for all cumulative logits. The cumulative logits differ only in their intercepts. In this formulation, the $\{\alpha_j\}$ increase with j, because $\Pr(Y \leq j \,|\, \boldsymbol{x})$ increases with j for fixed $\boldsymbol{x}$.[1] Figure 8.1 portrays the PO model for a single quantitative predictor x with $m = 4$ response categories.

The name "proportional odds" stems from the fact that under Eqn. (8.2), for fixed $\boldsymbol{x}$, the cumulative log odds (logits) for categories j and $j\prime$ are constant and their difference is $(\alpha_j - \alpha_{j\prime})$, so the odds have a constant ratio $\exp(\alpha_j - \alpha_{j\prime}) = \exp(\alpha_j)/\exp(\alpha_{j\prime})$, or are proportional. Similarly, the ratio of the cumulative odds of making a response $Y \leq j$ at values of the predictors $\boldsymbol{x} = \boldsymbol{x}_1$ are $\exp((\boldsymbol{x}_1 - \boldsymbol{x}_2)^{\mathsf{T}}\boldsymbol{\beta})$ times the odds of this response at $\boldsymbol{x} = \boldsymbol{x}_2$, so the log cumulative odds ratio is proportional to the difference between $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$.

8.1.1 Latent variable interpretation

For a binary response, an alternative motivation for logistic regression regards the relation of the observed Y as arising from a continuous, unobserved, (latent) response variable, ξ, representing the propensity for a "success" (1) rather than "failure" (0). The latent response is assumed to be linearly related to the predictors $\boldsymbol{x}$ according to

$$\xi_i = \alpha + \boldsymbol{x}_i^{\mathsf{T}}\beta + \epsilon_i = \alpha + \beta_1 x_{i1} + \cdots + \beta_p x_{ip} + \epsilon_i \,. \tag{8.3}$$

However, we can only observe $Y_i = 1$ when ξ_i passes some threshold, that with some convenient scaling can be taken as $\xi_i > 0 \Longrightarrow Y_i = 1$.[2]

The latent variable motivation extends directly to an ordinal response under the PO model. We now assume that there is a set of $m - 1$ thresholds, $\alpha_1 < \alpha_2 < \cdots < \alpha_{m-1}$ for the latent variable ξ_i in Eqn. (8.3), and we observe

$$Y_i = j \quad \text{if} \quad \alpha_{j-1} < \xi_i \leq \alpha_j \,,$$

[1] Some authors and some software describe the PO model in terms of $\text{logit}[\Pr(Y > j \,|\, \boldsymbol{x})]$, so the signs and order of the intercepts, α_j, are reversed.

[2] The latent variable derivation of logistic regression (and the related probit model) was fundamental in the history of statistical methods for discrete response outcomes. An early example was Thurstone's (1927) *Law of comparative judgment* designed to account for psychological preference by assuming an underlying latent continuum of "hedonic values." Similarly, the probit model arose from dose-response studies in toxicology (Bliss, 1934, Finney, 1947) where the number killed by some chemical agent was related to its type, dose, or concentration. The idea of a latent variable was also at the heart of the development of factor analysis (Bentler, 1980), and latent class analysis (Lazarsfeld, 1950, 1954) was developed to treat the problem of classifying individuals into discrete latent classes from fallible measurements. See Bollen (2002) for a useful overview of latent variable models in the social sciences.

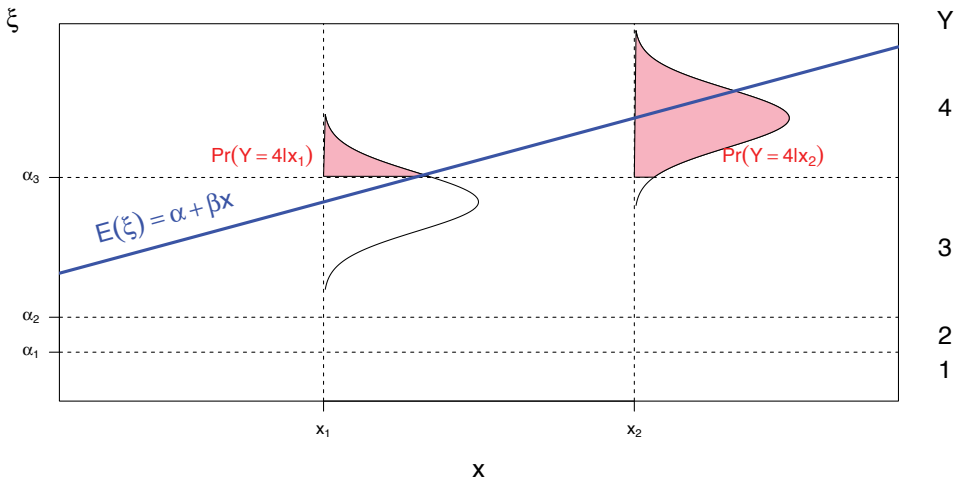


Figure 8.2: Latent variable representation of the proportional odds model for $m = 4$ response categories and a single quantitative predictor, x. *Source*: Adapted from Fox (2008, Fig 14.10), using code provided by John Fox.

with appropriate modifications to the inequalities at the end points.

This is illustrated in Figure 8.2 for a response with $m = 4$ ordered categories and a single quantitative predictor, x. The observable response Y categories are shown on the right vertical axis, and the corresponding latent continuous variable ξ on the left axis together with the thresholds $\alpha_1, \alpha_2, \alpha_3$. The (conditional) logistic distribution of ξ is shown at two values of x, and the shaded areas under the curve give the conditional probabilities $\Pr(Y = 4 \,|\, x_i)$ for the two values x_1 and x_2.

8.1.2 Fitting the proportional odds model

As mentioned earlier, there are a number of different R packages that provide facilities for fitting the PO model. These have somewhat different capabilities for reporting results, testing hypotheses, and plotting, so we generally use `polr()` in the MASS package, except where other packages offer greater convenience.

Unless the response variable has numeric values, it is important to ensure that it has been defined as an *ordered* factor (using `ordered()`). In the *Arthritis* data, the response `Improved` was set up this way, as we can check by printing some of the values.[3]

```
> data("Arthritis", package = "vcd")
> head(Arthritis$Improved, 8)

[1] Some   None   None   Marked Marked Marked None   Marked
Levels: None < Some < Marked
```

We fit the main effects model for the ordinal response using `polr()` as shown below. We also specify `Hess=TRUE` to have the function return the observed information matrix (called the Hessian), that is used in other operations to calculate standard errors.

```
> library(MASS)
> arth.polr <- polr(Improved ~ Sex + Treatment + Age,
+                   data = Arthritis, Hess = TRUE)
```

[3] As an unordered factor, the levels would be treated as ordered alphabetically, i.e., `Marked`, `None`, `Some`.

```
> summary(arth.polr)

Call:
polr(formula = Improved ~ Sex + Treatment + Age, data = Arthritis,
    Hess = TRUE)

Coefficients:
                   Value Std. Error t value
SexMale          -1.2517     0.5464   -2.29
TreatmentTreated  1.7453     0.4759    3.67
Age               0.0382     0.0184    2.07

Intercepts:
            Value  Std. Error t value
None|Some    2.532  1.057       2.395
Some|Marked  3.431  1.091       3.144

Residual Deviance: 145.46
AIC: 155.46
```

The output from the `summary()` method, shown above, gives the estimated coefficients ($\boldsymbol{\beta}$) and intercepts (α_j) labeled by the cutpoint on the ordinal response. It provides standard errors and t-values ($\beta_i / SE(\beta_i)$), but no significance tests or p-values. The car::`Anova()` method gives the appropriate tests.

```
> library(car)
> Anova(arth.polr)

Analysis of Deviance Table (Type II tests)

Response: Improved
          LR Chisq Df Pr(>Chisq)
Sex           5.69  1    0.01708 *
Treatment    14.71  1    0.00013 ***
Age           4.57  1    0.03251 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

8.1.3 Testing the proportional odds assumption

The simplicity of the PO model is achieved only when the proportional odds model holds for a given data set. In essence, a test of this assumption involves a contrast between the PO model and a generalized logit NPO model that allows different effects (slopes) of the predictors across the response categories:

$$\text{PO}: \quad L_j = \alpha_j + \boldsymbol{x}^{\mathsf{T}}\boldsymbol{\beta} \qquad j = 1, \ldots, m-1 \tag{8.4}$$

$$\text{NPO}: \quad L_j = \alpha_j + \boldsymbol{x}^{\mathsf{T}}\boldsymbol{\beta}_j \qquad j = 1, \ldots, m-1 \tag{8.5}$$

The most general test involves fitting both models and testing the difference in the residual deviance by a likelihood ratio test or using some other measure (such as AIC) for model comparison. The PO model (Eqn. (8.4)) has $(m-1)+p$ parameters, while the NPO model (Eqn. (8.5)) has $(m-1)(1+p) = m(1+p)$ parameters, which may be difficult to fit if this is large relative to the number of observations. An intermediate model, the ***partial proportional odds model*** (Peterson and Harrell, 1990), allows one subset of predictors, $\boldsymbol{x}_{po}$, to satisfy the proportional odds assumption (equal slopes), while the remaining predictors $\boldsymbol{x}_{npo}$ have slopes varying with the response level:

$$\text{PPO}: \quad L_j = \alpha_j + \boldsymbol{x}_{po}^{\mathsf{T}}\boldsymbol{\beta} + \boldsymbol{x}_{npo}^{\mathsf{T}}\boldsymbol{\beta}_j \qquad j = 1, \ldots, m-1\,. \tag{8.6}$$

In R, the PO and NPO models can be readily contrasted by fitting them both using `vglm()` in the VGAM package. This defines the `cumulative` family of models and allows a `parallel` option. With `parallel=TRUE`, this is equivalent to the `polr()` model, except that the signs of the coefficients are reversed.

```
> library(VGAM)
> arth.po <- vglm(Improved ~ Sex + Treatment + Age, data = Arthritis,
+                 family = cumulative(parallel = TRUE))
> arth.po

Call:
vglm(formula = Improved ~ Sex + Treatment + Age, family = cumulative(parallel = TRUE),
    data = Arthritis)

Coefficients:
   (Intercept):1    (Intercept):2          SexMale
        2.531990         3.430988         1.251671
TreatmentTreated              Age
       -1.745304        -0.038163

Degrees of Freedom: 168 Total; 163 Residual
Residual deviance: 145.46
Log-likelihood: -72.729
```

The more general NPO model can be fit using `parallel=FALSE`.

```
> arth.npo <- vglm(Improved ~ Sex + Treatment + Age, data = Arthritis,
+                  family = cumulative(parallel = FALSE))
> arth.npo

Call:
vglm(formula = Improved ~ Sex + Treatment + Age, family = cumulative(parallel = FALSE),
    data = Arthritis)

Coefficients:
     (Intercept):1      (Intercept):2          SexMale:1
         2.618539           3.431175           1.509827
         SexMale:2 TreatmentTreated:1 TreatmentTreated:2
         0.866434          -1.836929          -1.704011
             Age:1              Age:2
        -0.040866          -0.037294

Degrees of Freedom: 168 Total; 160 Residual
Residual deviance: 143.57
Log-likelihood: -71.787
```

The VGAM package defines a `coef()` method that can print the coefficients in a more readable matrix form giving the category cutpoints:

```
> coef(arth.po, matrix = TRUE)

                  logit(P[Y<=1]) logit(P[Y<=2])
(Intercept)             2.531990       3.430988
SexMale                 1.251671       1.251671
TreatmentTreated       -1.745304      -1.745304
Age                    -0.038163      -0.038163

> coef(arth.npo, matrix = TRUE)

                  logit(P[Y<=1]) logit(P[Y<=2])
(Intercept)             2.618539       3.431175
SexMale                 1.509827       0.866434
TreatmentTreated       -1.836929      -1.704011
Age                    -0.040866      -0.037294
```

In most cases, nested models can be tested using an `anova()` method, but the VGAM package has not implemented this for "vglm" objects. Instead, it provides an analogous function, `lrtest()`:

```
> VGAM::lrtest(arth.npo, arth.po)

Likelihood ratio test

Model 1: Improved ~ Sex + Treatment + Age
Model 2: Improved ~ Sex + Treatment + Age
  #Df LogLik Df Chisq Pr(>Chisq)
1 160  -71.8
2 163  -72.7  3  1.88        0.6
```

The LR test can be also calculated "manually" as shown below using the difference in residual deviance for the two models.

```
> tab <- cbind(
+   Deviance = c(deviance(arth.npo), deviance(arth.po)),
+         df = c(df.residual(arth.npo), df.residual(arth.po))
+         )
> tab <- rbind(tab, diff(tab))
> rownames(tab) <- c("GenLogit", "PropOdds", "LR test")
> tab <- cbind(tab, pvalue=1-pchisq(tab[,1], tab[,2]))
> tab

          Deviance  df  pvalue
GenLogit 143.5741 160 0.81966
PropOdds 145.4579 163 0.83435
LR test    1.8838   3 0.59686
```

The `vglm()` can also fit partial proportional odds models, by specifying a formula giving the terms for which the PO assumption should be taken as `TRUE` or `FALSE`. Here we illustrate this using `parallel=FALSE ~ Sex`, to fit separate slopes for males and females, but parallel lines for the other predictors. The same model would be fit using `parallel=TRUE ~ Treatment + Age`.

```
> arth.ppo <- vglm(Improved ~ Sex + Treatment + Age, data = Arthritis,
+   family = cumulative(parallel = FALSE ~ Sex))
> coef(arth.ppo, matrix = TRUE)

                  logit(P[Y<=1]) logit(P[Y<=2])
(Intercept)             2.542452       3.615561
SexMale                 1.483336       0.867362
TreatmentTreated       -1.775742      -1.775742
Age                    -0.039622      -0.039622
```

8.1.4 Graphical assessment of proportional odds

There are several graphical methods for visual assessment of the proportional odds assumption. These are all *marginal* methods, in that they treat the predictors one at a time. However, that provides one means to determine if a partial proportional odds model might be more appropriate. Harrell's work *Regression Modeling Strategies* (2001, Chapters 13–14) and the corresponding rms package provide an authoritative treatment and methods in R.

One simple idea is to plot the conditional mean or expected value $E(X \mid Y)$ of a given predictor, X, at each level of the ordered response Y. If the response behaves ordinally in relation to X, these means should be strictly increasing or decreasing with Y. For comparison, one can also plot the estimated conditional means $\widehat{E}(X \mid Y = j)$ under the fitted PO model X as the only predictor. If the PO assumption holds for this X, the model-mean curve should be close to the data mean curve.

```
> library(rms)
> arth.po2 <- lrm(Improved ~ Sex + Treatment + Age, data = Arthritis)
> arth.po2


Logistic Regression Model

lrm(formula = Improved ~ Sex + Treatment + Age, data = Arthritis)
                      Model Likelihood     Discrimination    Rank Discrim.
                         Ratio Test           Indexes          Indexes
Obs             84    LR chi2      24.46    R2       0.291    C       0.750
 None           42    d.f.             3    g        1.335    Dxy     0.500
 Some           14    Pr(> chi2) <0.0001    gr       3.801    gamma   0.503
 Marked         28                          gp       0.280    tau-a   0.309
max |deriv| 1e-07                          Brier    0.187

                   Coef    S.E.   Wald Z Pr(>|Z|)
y>=Some           -2.5320 1.0570 -2.40  0.0166
y>=Marked         -3.4310 1.0911 -3.14  0.0017
Sex=Male          -1.2517 0.5464 -2.29  0.0220
Treatment=Treated  1.7453 0.4759  3.67  0.0002
Age                0.0382 0.0184  2.07  0.0382
```

The plot of conditional X means is produced using the `plot.xmean.ordinaly()` as shown below. It produces one marginal panel for each predictor in the model. For categorical predictors, it plots only the overall most frequent category. The resulting plot is shown in Figure 8.3.

```
> op <- par(mfrow=c(1,3))
> plot.xmean.ordinaly(Improved ~ Sex + Treatment + Age, data=Arthritis,
+                     lwd=2, pch=16, subn=FALSE)
> par(op)
```

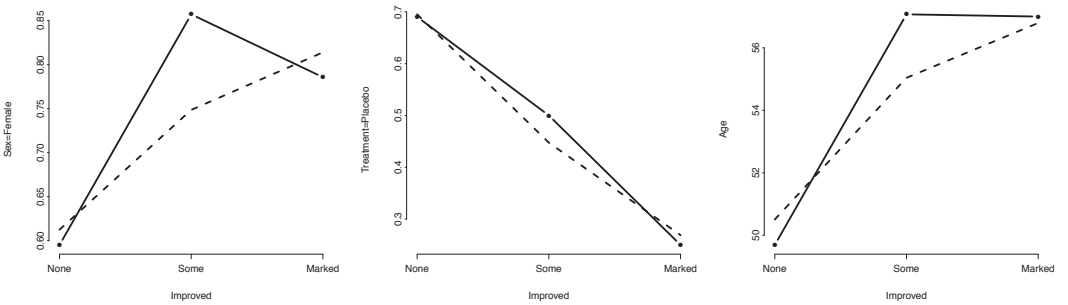


Figure 8.3: Visual assessment of ordinality and the proportional odds assumption for predictors in the Arthritis data. Solid lines connect the stratified means of X given Y. Dashed lines show the estimated expected value of X given Y=j if the proportional odds model holds for X.

In Figure 8.3, there is some evidence that the effect of `Sex` is non-monotonic and the means differ from their model-implied values under the PO assumption. The effect of `Treatment` looks good by this method, and the effect of `Age` hints that the upper two categories may not be well-distinguished as an ordinal response.

Of course, this example has only a modest total sample size, and this method only examines the marginal effects of the predictors. Nevertheless, it is a useful supplement to the statistical methods described earlier.

8.1.5 Visualizing results for the proportional odds model

Results from the PO model (and other models for polytomous responses) can be graphed using the same ideas and methods shown earlier for a binary or binomial response. In particular, full-model plots (described earlier in Section 7.3.2) and effect plots (Section 7.3.3) are still very helpful.

But now there is the additional complication that the response variable has $m > 2$ levels and so needs to be represented by $m - 1$ curves or panels in addition to those related to the predictor variables.

8.1.5.1 Full-model plots

For full-model plots, we continue the idea of appending the fitted response probabilities (or logits) to the data frame and plotting these in relation to the predictors. The `predict()` method returns the highest probability category label by default (with `type="class"`), so to get the fitted probabilities you have to ask for `type="probs"`, as shown below.

```
> arth.fitp <- cbind(Arthritis,
+                    predict(arth.polr, type = "probs"))
> head(arth.fitp)

  ID Treatment  Sex Age Improved    None    Some  Marked
1 57   Treated Male  27     Some 0.73262 0.13806 0.12932
2 46   Treated Male  29     None 0.71740 0.14443 0.13816
3 77   Treated Male  30     None 0.70960 0.14763 0.14277
4 17   Treated Male  32   Marked 0.69363 0.15400 0.15237
5 36   Treated Male  46   Marked 0.57025 0.19504 0.23471
6 23   Treated Male  58   Marked 0.45634 0.21713 0.32653
```

For plotting, it is most convenient to reshape these from wide to long format using `melt()` in the reshape2 (Wickham, 2014b) package. The response category is named `Level`.

```
> library(reshape2)
> plotdat <- melt(arth.fitp,
+                 id.vars = c("Sex", "Treatment", "Age", "Improved"),
+                 measure.vars = c("None", "Some", "Marked"),
+                 variable.name = "Level",
+                 value.name = "Probability")
> ## view first few rows
> head(plotdat)

   Sex Treatment Age Improved Level Probability
1 Male   Treated  27     Some  None     0.73262
2 Male   Treated  29     None  None     0.71740
3 Male   Treated  30     None  None     0.70960
4 Male   Treated  32   Marked  None     0.69363
5 Male   Treated  46   Marked  None     0.57025
6 Male   Treated  58   Marked  None     0.45634
```

We can now plot `Probability` against `Age`, using `Level` to assign different colors to the lines for the response categories, as seen in Figure 8.4. Here, `facet_grid()` is used to split the plot into separate panels by `Sex` and `Treatment`. In this example, the directlabels package is also used replace the default legend created by `ggplot()` with category labels on the curves themselves, which is easier to read.

```
> library(ggplot2)
> library(directlabels)
> gg <- ggplot(plotdat, aes(x = Age, y = Probability, colour = Level)) +
+     geom_line(size = 2.5) + theme_bw() + xlim(10, 80) +
+     geom_point(color = "black", size = 1.5) +
```

```
+      facet_grid(Sex ~ Treatment,
+                 labeller = function(x, y) sprintf("%s = %s", x, y)
+                 )
> direct.label(gg)
```

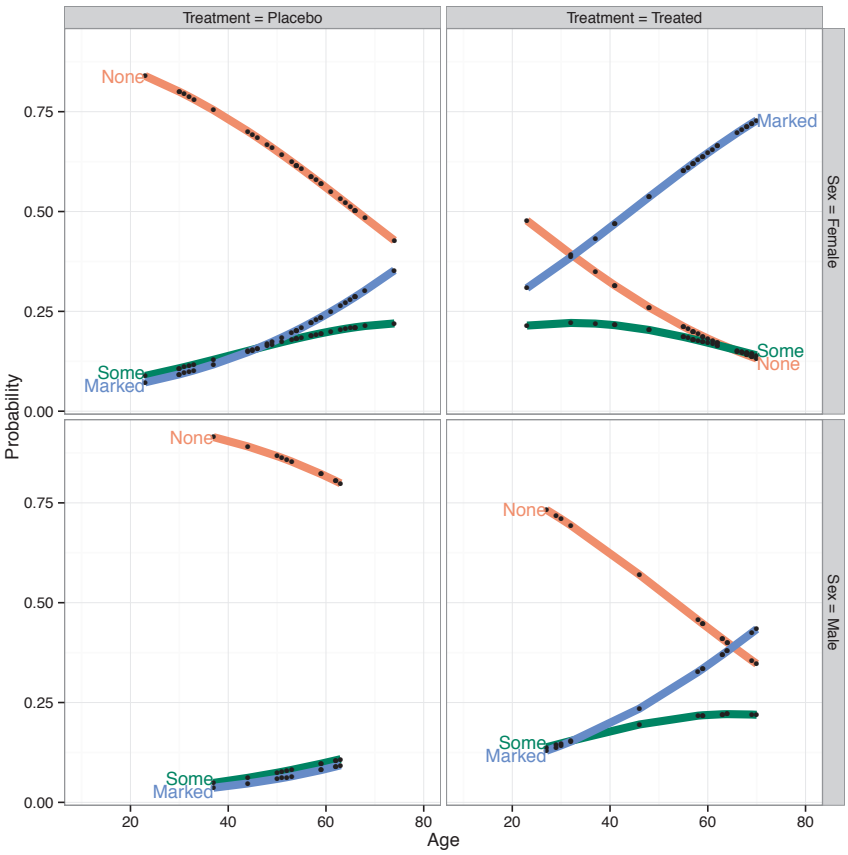


Figure 8.4: Predicted probabilities for the proportional odds model fit to the Arthritis data.

Although we now have three response curves in each panel in Figure 8.4, this plot is relatively easy to understand: (a) In each panel, the probability of no improvement decreases with age, while that for marked improvement increases. (b) It is easy to compare the placebo and treated groups in each row, showing that no improvement decreases, while marked improvement increases with the active treatment. (On the other hand, this layout makes it harder to compare panels vertically for males and females in each condition.) (c) The points show where the observations are located in each panel; so, we can see that the data is quite thin for males given the placebo.[4]

8.1.5.2 Effect plots

For PO models fit using `polr()`, the effects package provides two different styles for plotting a given effect. By default, curves are plotted in separate panels for the different response levels of a given effect, together with confidence bands for predicted probabilities. This form provides confidence bands and rug plots for the observations, but the default vertical arrangement of the

[4]One way to improve (pun intended) this graph would be to show the points on the lines only for the actual level of `Improve` for each observation.

panels makes it harder to compare the trends for the different response levels. The alternative *stacked* format shows the changes in response level more directly, but doesn't provide confidence bands.

Figure 8.5 shows these two styles for the main effect of `Age` in the proportional odds model, `arth.polr` fit earlier.

```
> library(effects)
> plot(Effect("Age", arth.polr))
> plot(Effect("Age", arth.polr), style = "stacked",
+       key.args = list(x = .55, y = .9))
```

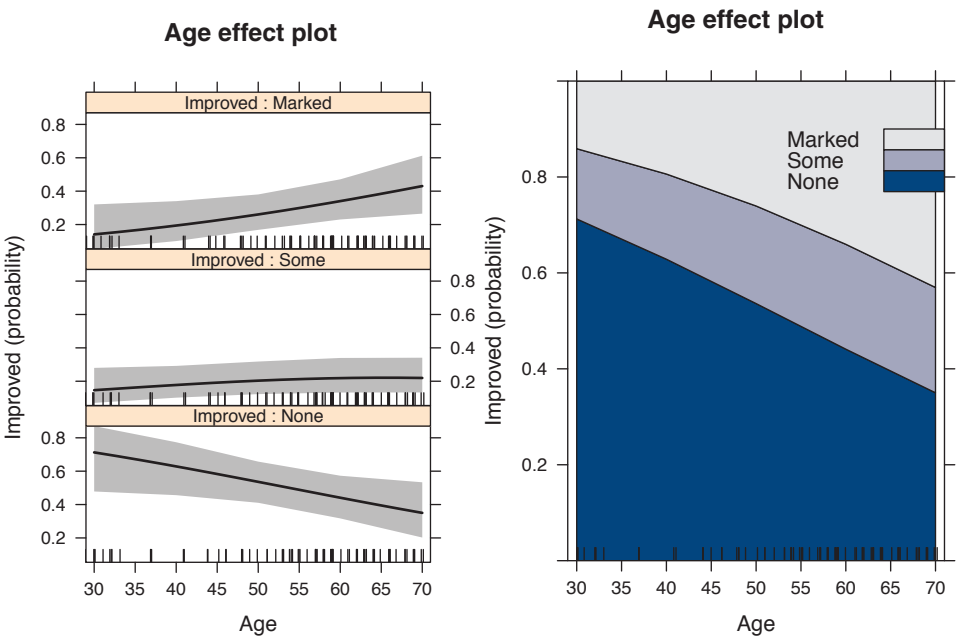


Figure 8.5: Effect plots for the effect of Age in the proportional odds model for the Arthritis data. Left: responses shown in separate panels. Right: responses shown in stacked format.

Even though this model includes only main effects, you can still plot the higher-order effects for more focal predictors in a coherent display. Figure 8.6 shows the predicted probabilities for all three predictors together. Again, visual comparison is easier horizontally for placebo versus treated groups, but you can also see that the prevalence of marked improvement is greater for females than for males.

```
> plot(Effect(c("Treatment", "Sex", "Age"), arth.polr),
+       style = "stacked", key.arg = list(x = .8, y = .9))
```

Finally, the latent variable interpretation of the PO model provides for simpler plots on the logit scale. Figure 8.7 shows this plot for the effects of `Treatment` and `Age` (collapsed over `Sex`) produced with the argument `latent=TRUE` to `Effect()`. In this plot, there is a single line in each panel for the effect (slope) of `Age` on the log odds. The dashed horizontal lines give the thresholds between the adjacent response categories corresponding to the intercepts.

```
> plot(Effect(c("Treatment", "Age"), arth.polr, latent = TRUE), lwd = 3)
```

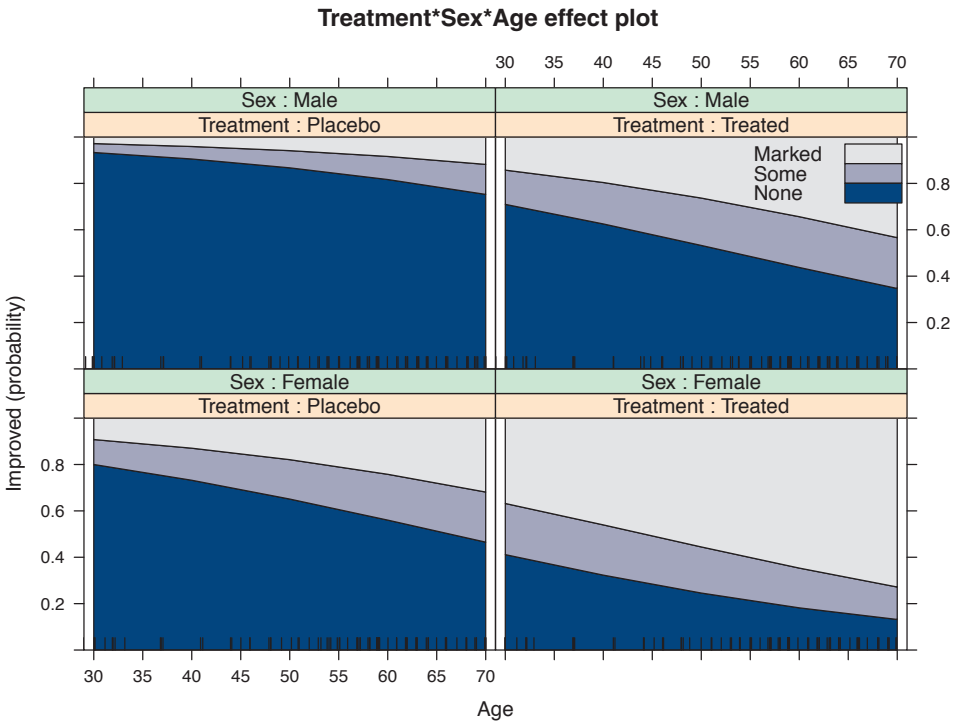


Figure 8.6: Effect plot for the effects of Treatment, Sex, and Age in the Arthritis data.

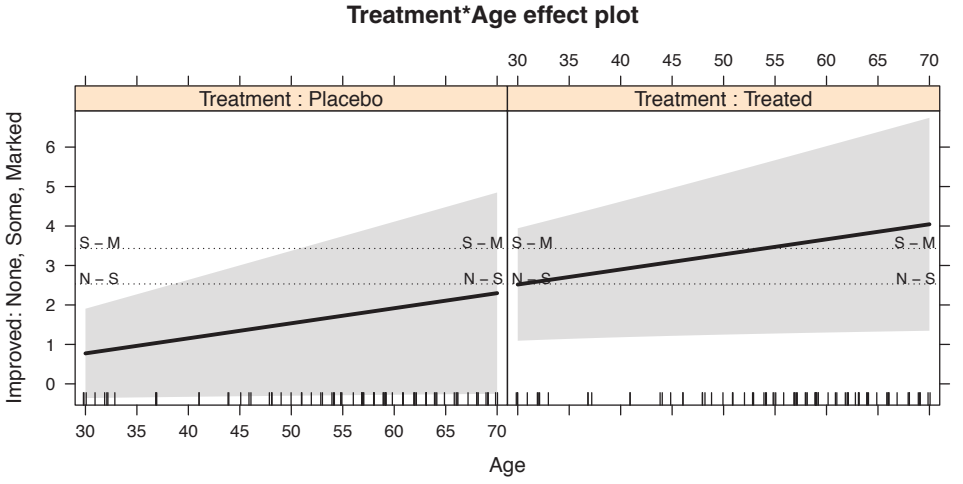


Figure 8.7: Latent variable effect plot for the effects of Treatment and Age in the Arthritis data.

8.2 Nested dichotomies

The method of ***nested dichotomies*** provides another simple way to analyze a polytomous response in the framework of logistic regression (or other generalized linear models). This method does not require an ordinal response or special software. Instead, it uses the familiar binary logistic model and fits $m - 1$ separate models for each of a hierarchically nested set of comparisons among the response categories.

Taken together, this set of models for the dichotomies comprises a complete model for the polytomous response. As well, these models are statistically independent, so test statistics such as G^2 or Wald tests can be added to give overall tests for the full polytomy.

For example, the response categories $Y = \{1,2,3,4\}$ could be divided first as $\{1,2\}$ vs. $\{3,4\}$, as shown in the left side of Figure 8.8. Then these two dichotomies could be divided as $\{1\}$ vs. $\{2\}$, and $\{3\}$ vs. $\{4\}$. Alternatively, these response categories could be divided as shown in the right side of Figure 8.8: first, $\{1\}$ vs. $\{2,3,4\}$, then $\{2\}$ vs $\{3,4\}$, and finally $\{3\}$ vs. $\{4\}$.

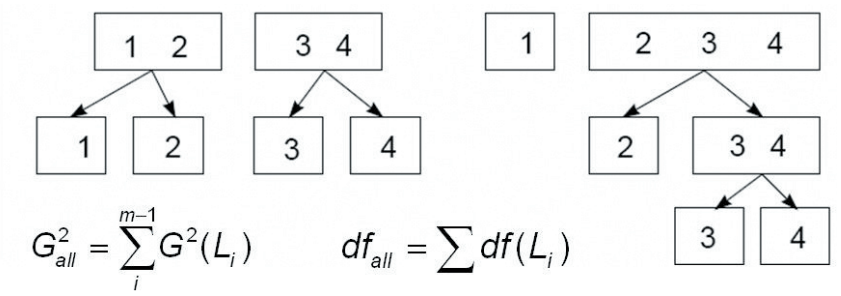


Figure 8.8: Nested dichotomies. The boxes show two different ways a four-category response can be represented as three nested dichotomies. Adapted from Fox (2008).

Such models make the most sense when there are substantive reasons for considering the response categories in terms of such dichotomies. Two examples are shown in Figure 8.9.

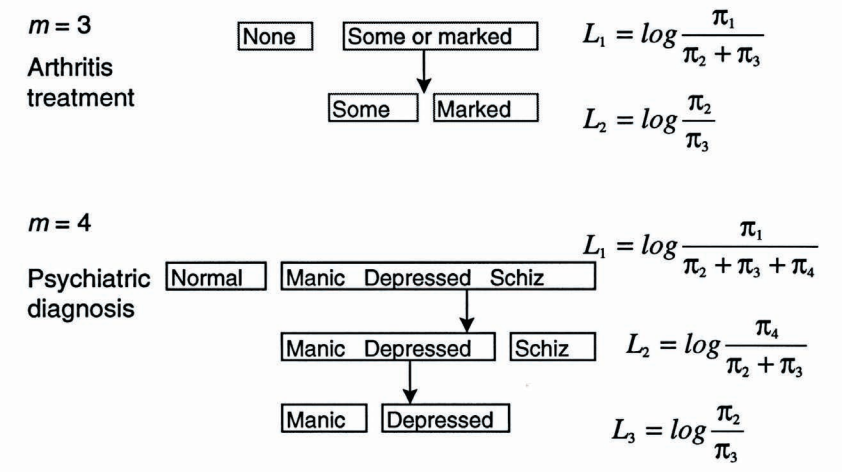


Figure 8.9: Examples of nested dichotomies and the corresponding logits.

- For the `Arthritis` data, it is sensible to consider one dichotomy ("better"), with logit L_1, between the categories of `"None"` compared to `"Some"` or `"Marked"`. A second dichotomy, with logit L_2, would then distinguish between the some and marked response categories.
- For a second case where patients are classified into $m = 4$ psychiatric diagnostic categories, the first dichotomy, with logit L_1, distinguishes those considered normal from all others given a clinical diagnosis. Two other dichotomies are defined to further divide the non-normal categories.

Then, consider the separate logit models for these $m - 1$ dichotomies, with different intercepts α_j and slopes $\boldsymbol{\beta}_j$ for each dichotomy,

$$\begin{aligned} L_1 &= \alpha_1 + \boldsymbol{x}^\mathsf{T}\boldsymbol{\beta}_1 \\ L_2 &= \alpha_2 + \boldsymbol{x}^\mathsf{T}\boldsymbol{\beta}_2 \\ \vdots &= \vdots \\ L_{m-1} &= \alpha_{m-1} + \boldsymbol{x}^\mathsf{T}\boldsymbol{\beta}_{m-1} \, . \end{aligned}$$

EXAMPLE 8.1: Women's labor force participation

The data set `Womenlf` in the car package gives the result of a 1977 Canadian survey. It contains data for 263 married women of age 21–30 who indicated their working status (outside the home) as not working, working part time, or working full time, together with their husband's income and a binary indicator of whether they had one or more young children in their household. (Another variable, region of Canada, had no effects in these analyses, and is not examined here.) This example follows Fox and Weisberg (2011a, Section 5.8).

```
> library(car)    # for data and Anova()
> data("Womenlf", package = "car")
> some(Womenlf)

      partic hincome children  region
7   not.work      15  present Ontario
29  not.work      17  present Prairie
45  parttime       5  present Ontario
82  parttime      15  present Ontario
91  not.work      35   absent Ontario
97  not.work      17  present Ontario
129 parttime      13  present Prairie
138 not.work      13  present Ontario
175 fulltime       9   absent Ontario
200 fulltime      11   absent  Quebec
```

In this example, it makes sense to consider a first dichotomy (`working`) between women who are not working vs. those who are (full time or part time). A second dichotomy (`fulltime`) contrasts full time work vs. part time work, among those women who are working at least part time. These two binary variables are created in the data frame using the `recode()` function from the car package.

```
> # create dichotomies
> Womenlf <- within(Womenlf,{
+   working <-  recode(partic, " 'not.work' = 'no'; else = 'yes' ")
+   fulltime <- recode(partic,
+     " 'fulltime' = 'yes'; 'parttime' = 'no'; 'not.work' = NA")})
> some(Womenlf)

      partic hincome children  region fulltime working
81  fulltime      13   absent Ontario      yes     yes
```

```
96  not.work      17  present Ontario     <NA>      no
97  not.work      17  present Ontario     <NA>      no
115 parttime      13  present Prairie       no     yes
123 fulltime       9   absent Ontario      yes     yes
131 parttime      19  present Ontario       no     yes
153 not.work       5   absent      BC     <NA>      no
190 not.work      23  present      BC     <NA>      no
248 not.work      23   absent  Quebec     <NA>      no
255 fulltime      11   absent  Quebec      yes     yes
```

The tables below show how the response `partic` relates to the recoded binary variables, `working` and `fulltime`. Note that the `fulltime` variable is recoded to `NA` for women who are not working.

```
> with(Womenlf, table(partic, working))

          working
partic      no yes
  fulltime   0  66
  not.work 155   0
  parttime   0  42

> with(Womenlf, table(partic, fulltime, useNA = "ifany"))

          fulltime
partic      no yes <NA>
  fulltime   0  66    0
  not.work   0   0  155
  parttime  42   0    0
```

We proceed to fit two separate binary logistic regression models for the derived dichotomous variables. For the `working` dichotomy, we get the following results:

```
> mod.working <- glm(working ~ hincome + children, family = binomial,
+                    data = Womenlf)
> summary(mod.working)


Call:
glm(formula = working ~ hincome + children, family = binomial,
    data = Womenlf)

Deviance Residuals:
   Min      1Q  Median      3Q     Max
-1.677  -0.865  -0.777   0.929   1.997

Coefficients:
                Estimate Std. Error z value Pr(>|z|)
(Intercept)       1.3358     0.3838    3.48   0.0005 ***
hincome          -0.0423     0.0198   -2.14   0.0324 *
childrenpresent  -1.5756     0.2923   -5.39    7e-08 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 356.15  on 262  degrees of freedom
Residual deviance: 319.73  on 260  degrees of freedom
AIC: 325.7

Number of Fisher Scoring iterations: 4
```

And, similarly for the `fulltime` dichotomy:

```
> mod.fulltime <- glm(fulltime ~ hincome + children, family = binomial,
+                     data = Womenlf)
> summary(mod.fulltime)


Call:
glm(formula = fulltime ~ hincome + children, family = binomial,
    data = Womenlf)

Deviance Residuals:
   Min      1Q  Median      3Q     Max
-2.405  -0.868   0.395   0.621   1.764

Coefficients:
                Estimate Std. Error z value Pr(>|z|)
(Intercept)       3.4778     0.7671    4.53  5.8e-06 ***
hincome          -0.1073     0.0392   -2.74   0.0061 **
childrenpresent  -2.6515     0.5411   -4.90  9.6e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 144.34  on 107  degrees of freedom
Residual deviance: 104.49  on 105  degrees of freedom
  (155 observations deleted due to missingness)
AIC: 110.5

Number of Fisher Scoring iterations: 5
```

Although these were fit separately, we can view this as a combined model for the three-level response, with the following coefficients:

```
> cbind(working = coef(mod.working), fulltime = coef(mod.fulltime))

                  working fulltime
(Intercept)      1.335830  3.47777
hincome         -0.042308 -0.10727
childrenpresent -1.575648 -2.65146
```

Writing these out as equations for the logits, we have:

$$L_1 = \log \frac{\Pr(\text{working})}{\Pr(\text{notworking})} = 1.336 - 0.042\,\text{hincome} - 1.576\,\text{children} \tag{8.7}$$

$$L_2 = \log \frac{\Pr(\text{fulltime})}{\Pr(\text{parttime})} = 3.478 - 0.1072\,\text{hincome} - 2.652\,\text{children} \tag{8.8}$$

For both dichotomies, increasing income of the husband and the presence of young children decrease the log odds of a greater level of work. However, for those women who are working, the effects of husband's income and and children are greater on the choice between full time and part time work than they are for all women on the choice between working and not working.

As we mentioned above, the use of nested dichotomies implies that the models fit to the separate dichotomies are statistically independent. Thus, we can additively combine χ^2 statistics and degrees of freedom to give overall tests for the polytomous response.

For example, here we define a function, `LRtest()`, to calculate the likelihood ratio test of the hypothesis $H_0 : \boldsymbol{\beta} = \mathbf{0}$ for all predictors simultaneously. We then use this to display these tests for each sub-model, as well as the combined model based on the sums of the test statistic and degrees of freedom.

```
> LRtest <- function(model)
+   c(LRchisq = model$null.deviance - model$deviance,
+     df = model$df.null - model$df.residual)
>
> tab <- rbind(working = LRtest(mod.working),
+              fulltime = LRtest(mod.fulltime))
> tab <- rbind(tab, All = colSums(tab))
> tab <- cbind(tab, pvalue = 1- pchisq(tab[,1], tab[,2]))
> tab

         LRchisq df     pvalue
working   36.418  2 1.2355e-08
fulltime  39.847  2 2.2252e-09
All       76.265  4 1.1102e-15
```

Similarly, you can carry out tests of individual predictors, $H_0 : \boldsymbol{\beta}_i = \mathbf{0}$, for the polytomy by adding the separate χ^2s from `Anova()`.

```
> Anova(mod.working)

Analysis of Deviance Table (Type II tests)

Response: working
         LR Chisq Df Pr(>Chisq)
hincome   4.82637  1   0.028028 *
children 31.32288  1 2.1849e-08 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

> Anova(mod.fulltime)

Analysis of Deviance Table (Type II tests)

Response: fulltime
         LR Chisq Df Pr(>Chisq)
hincome    8.9813  1  0.0027275 **
children  32.1363  1 1.4373e-08 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

For example, the test for husband's income gives $\chi^2 = 4.826 + 8.981 = 13.807$ with 2 df.

As before, you can plot the fitted values from such models, either on the logit scale (for the separate logit equations) or in terms of probabilities for the various responses. The general idea is the same: obtain the fitted values from `predict()` using data frame containing the values of the predictors. However, now we have to combine these for each of the sub-models.

We calculate these values below, on both the logit scale and the response scale of probabilities. The `newdata` argument to `predict()` is constructed as the combinations of values for `hincome` and `children`.[5]

```
> predictors <- expand.grid(hincome = 1 : 50,
+                           children =c('absent', 'present'))
> fit <- data.frame(predictors,
+     p.working = predict(mod.working, predictors, type = "response"),
+     p.fulltime = predict(mod.fulltime, predictors, type = "response"),
+     l.working = predict(mod.working, predictors, type = "link"),
+     l.fulltime = predict(mod.fulltime, predictors, type = "link")
+ )
> print(some(fit, 5), digits = 3)
```

[5] Alternatively, using the predictor values in the *Womenlf* data would give the fitted values for the cases in the data, and allow a more data-centric plot as shown in Figure 8.4.

```
   hincome children p.working p.fulltime l.working l.fulltime
6        6   absent     0.747     0.9445     1.082      2.834
8        8   absent     0.731     0.9321     0.997      2.620
39      39   absent     0.422     0.3306    -0.314     -0.706
68      18  present     0.269     0.2489    -1.001     -1.105
88      38  present     0.136     0.0373    -1.848     -3.250
```

One wrinkle here is that the probabilities for working full time and part time are conditional on working. We calculate the unconditional probabilities as shown below and choose to display the probability of *not* working as the complement of working.

```
> fit <- within(fit, {
+   `full-time` <- p.working * p.fulltime
+   `part-time` <- p.working * (1 - p.fulltime)
+   `not working` <- 1 - p.working
+   })
```

To plot these fitted values, we will again create a conditional plot using ggplot2. Since this requires having all probabilities in one column, together with an additional grouping variable identifying the working status, we need to reshape `fit` from "wide" to "long" format, yet again using the `melt()` from the reshape2 package:

```
> fit2 <- melt(fit,
+              measure.vars = c("full-time", "part-time", "not working"),
+              variable.name = "Participation",
+              value.name = "Probability")
```

The lines below give the plot shown in Figure 8.10:

```
> gg <- ggplot(fit2,
+              aes(x = hincome, y = Probability, colour= Participation)) +
+         facet_grid(~ children,
+                    labeller = function(x, y) sprintf("%s = %s", x, y)) +
+         geom_line(size = 2) + theme_bw() +
+         scale_x_continuous(limits = c(-3, 55)) +
+         scale_y_continuous(limits = c(0, 1))
>
> direct.label(gg, list("top.bumptwice", dl.trans(y = y + 0.2)))
```

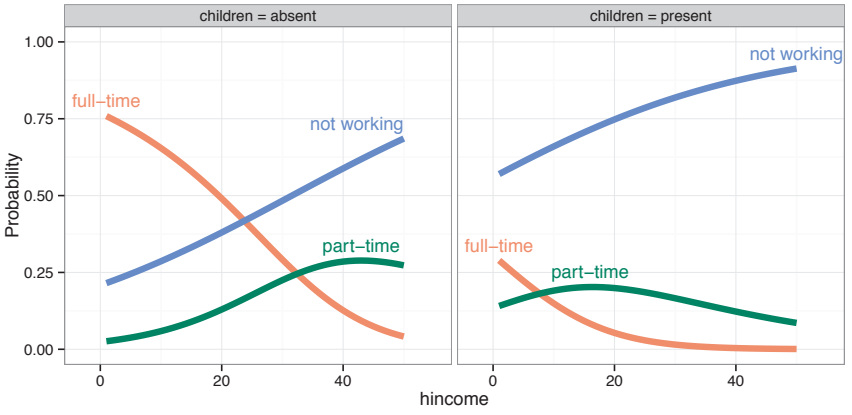


Figure 8.10: Fitted probabilities from the models for nested dichotomies fit to the data on women's labor force participation.

(Note the extension of the axes to avoid label clipping.)

We can see that the decision not to work outside the home increases strongly with husband's income, and is higher when there are children present. As well, among working women, the decision to work full time as opposed to part time decreases strongly with husband's income, and is less likely with young children.

Similarly, we plot the fitted logits for the two dichotomies in `l.working` and `l.fulltime` as shown below, giving Figure 8.11.

```
> fit3 <- melt(fit,
+              measure.vars = c("l.working", "l.fulltime"),
+              variable.name = "Participation",
+              value.name = "LogOdds")
> levels(fit3$Participation) <- c("working", "full-time")
>
> gg <- ggplot(fit3,
+              aes(x = hincome, y = LogOdds, colour = Participation)) +
+         facet_grid(~ children,
+                    labeller = function(x, y) sprintf("%s = %s", x, y)) +
+         geom_line(size = 2) + theme_bw() +
+         scale_x_continuous(limits = c(-3, 50)) +
+         scale_y_continuous(limits = c(-5, 4))
>
> direct.label(gg, list("top.bumptwice", dl.trans(y = y + 0.2)))
```

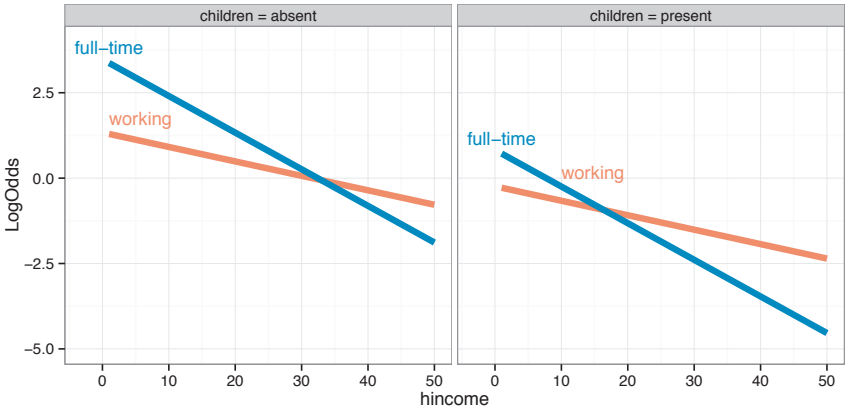


Figure 8.11: Fitted log odds from the models for nested dichotomies fit to the data on women's labor force participation.

This is essentially a graph of the fitted equations for L_1 and L_2 shown in Eqn. (8.7). It shows how the choice of full time work as opposed to part time depends more strongly on husband's income among women who are working than does the choice of working at all among all women. It also illustrates why the proportional odds assumption would not be reasonable for this data: that would require equal slopes for the two lines within each panel. △

8.3 Generalized logit model

The generalized logit (or multinomial logit) approach models the probabilities of the m response categories directly as a set of $m - 1$ logits. These compare each of the first $m - 1$ categories to

the last category, which serves as the baseline.[6] The logits for any other pair of categories can be retrieved from the $m-1$ fitted ones.

When there are p predictors, $x_1, x_2, \ldots, x_p$, which may be quantitative or categorical, the generalized logit model expresses the logits as

$$\begin{aligned} L_{jm} \equiv \log \frac{\pi_{ij}}{\pi_{im}} &= \beta_{0j} + \beta_{1j}\, x_{i1} + \beta_{2j}\, x_{i2} + \cdots + \beta_{kj}\, x_{ip} \quad j = 1, \ldots, m-1 \\ &= \boldsymbol{x}_i^{\mathsf{T}} \boldsymbol{\beta}_j \,. \end{aligned} \tag{8.9}$$

Thus, there is one set of fitted coefficients, $\boldsymbol{\beta}_j$, for each response category except the last. Each coefficient, β_{hj}, gives the effect, for a unit change in the predictor x_h, on the log odds that an observation had a response in category $Y = j$, as opposed to category $Y = m$.

The probabilities themselves can be expressed as

$$\begin{aligned} \pi_{ij} &= \frac{\exp(\boldsymbol{x}_i^{\mathsf{T}} \boldsymbol{\beta}_j)}{1 + \sum_{\ell=1}^{m-1} \exp(\boldsymbol{x}_i^{\mathsf{T}} \boldsymbol{\beta}_j)} \qquad j = 1, 2, \ldots m-1 \,, \\ \pi_{im} &= 1 - \sum_{i=1}^{m-1} \pi_{ij} \qquad \text{for } Y = m \,. \end{aligned}$$

Parameters in the $m-1$ equations Eqn. (8.9) can be used to determine the probabilities or the predicted log odds for any pair of response categories by subtraction. For instance, for an arbitrary pair of categories, a and b, and two predictors, x_1 and x_2,

$$\begin{aligned} L_{ab} &= \log \frac{\pi_{ia}/\pi_{im}}{\pi_{ib}/\pi_{im}} \\ &= \log \frac{\pi_{ia}}{\pi_{im}} - \log \frac{\pi_{ib}}{\pi_{im}} \\ &= (\beta_{0a} - \beta_{0b}) + (\beta_{1a} - \beta_{1b}) x_{i1} + (\beta_{2a} - \beta_{2b}) x_{i2} \,. \end{aligned}$$

For example, the coefficient for x_{i1} in L_{ab} is just $(\beta_{1a} - \beta_{1b})$. Similarly, the predicted logit for any pair of categories can be calculated as

$$\hat{L}_{ab} = \hat{L}_{am} - \hat{L}_{bm} \,.$$

The generalized logit model can be fit most conveniently in R using the function `multinom()` in the nnet package, and the effects package has a set of methods for "multinom" models. These models can also be fit using VGAM and the mlogit (Croissant, 2013) package.

EXAMPLE 8.2: Women's labor force participation

To illustrate this method, we fit the generalized logit model to the women's labor force participation data as explained below. The response, `partic`, is a character factor, and, by default `multinom()` treats these in alphabetical order and uses the *first* level as the baseline category.

```
> levels(Womenlf$partic)

[1] "fulltime" "not.work" "parttime"
```

Although the multinomial model does not depend on the baseline category, it makes interpretation easier to choose `"not.work"` as the reference level, which we do with `relevel()`.[7]

[6] When the response is a factor, any category can be selected as the baseline level using `relevel()`.

[7] Alternatively, we could declare `partic` an *ordered* factor, using `ordered()`.

```
> # choose not working as baseline category
> Womenlf$partic <- relevel(Womenlf$partic, ref = "not.work")
```

We fit the main effects model for husband's income and children as follows. As we did with `polr()` (Section 8.1), specifying `Hess=TRUE` saves the Hessian and facilitates calculation of standard errors and hypothesis tests.

```
> library(nnet)
> wlf.multinom <- multinom(partic ~ hincome + children,
+                          data = Womenlf, Hess = TRUE)

# weights:  12 (6 variable)
initial  value 288.935032
iter  10 value 211.454772
final  value 211.440963
converged
```

The `summary()` method for "multinom" objects doesn't calculate test statistics for the estimated coefficients by default. The option `Wald=TRUE` produces Wald z-test statistics, calculated as $z = \beta/SE(\beta)$.

```
> summary(wlf.multinom, Wald = TRUE)

Call:
multinom(formula = partic ~ hincome + children, data = Womenlf,
    Hess = TRUE)

Coefficients:
          (Intercept)    hincome childrenpresent
fulltime       1.9828 -0.0972321       -2.558605
parttime      -1.4323  0.0068938        0.021456

Std. Errors:
          (Intercept)  hincome childrenpresent
fulltime      0.48418 0.028096         0.36220
parttime      0.59246 0.023455         0.46904

Value/SE (Wald statistics):
          (Intercept)  hincome childrenpresent
fulltime       4.0953 -3.46071       -7.064070
parttime      -2.4176  0.29392        0.045744

Residual Deviance: 422.88
AIC: 434.88
```

Notice that the coefficients, their standard errors and the Wald test z values are printed in separate tables. The first line in each table pertains to the logit comparing full time work with the not working reference level; the second line compares part time work against not working.

For those who like p-values for significance tests, you can calculate these from the results returned by the `summary()` method in the `Wald.ratios` component, using the standard normal asymptotic approximation:

```
> stats <- summary(wlf.multinom, Wald = TRUE)
> z <- stats$Wald.ratios
> p <- 2 * (1 - pnorm(abs(z)))
> zapsmall(p)

          (Intercept) hincome childrenpresent
fulltime      0.00004 0.00054         0.00000
parttime      0.01562 0.76882         0.96351
```

The interpretation of these tests is that both husband's income and presence of children have highly significant effects on the comparison of working full time as opposed to not working, while neither of these predictors are significant for the comparison of working part time vs. not working.

So far, we have assumed that the effects of husband's income and presence of young children are additive on the log odds scale. We can test this assumption by allowing an interaction of those effects and testing it for significance.

```
> wlf.multinom2 <- multinom(partic ~ hincome * children,
+                           data = Womenlf, Hess = TRUE)

# weights:  15 (8 variable)
initial  value 288.935032
iter  10 value 210.797079
final  value 210.714841
converged

> Anova(wlf.multinom2)

Analysis of Deviance Table (Type II tests)

Response: partic
                 LR Chisq Df Pr(>Chisq)
hincome              15.2  2    0.00051 ***
children             63.6  2    1.6e-14 ***
hincome:children      1.5  2    0.48378
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The test for the interaction term, `hincome:children`, is not significant, so we can abandon this model.

Full model plots of the fitted values can be plotted as shown earlier in Example 8.1: obtain the fitted values over a grid of the predictors and plot these.

```
> predictors <- expand.grid(hincome = 1 : 50,
+                           children = c("absent", "present"))
> fit <- data.frame(predictors,
+                   predict(wlf.multinom, predictors, type = "probs")
+                   )
```

Plotting these fitted values gives the plot shown in Figure 8.12.

```
> fit2 <- melt(fit,
+              measure.vars = c("not.work", "fulltime", "parttime"),
+              variable.name = "Participation",
+              value.name = "Probability")
> levels(fit2$Participation) <- c("not working", "full-time", "part-time")
>
> gg <- ggplot(fit2,
+              aes(x = hincome, y = Probability, colour = Participation)) +
+         facet_grid(~ children,
+                    labeller = function(x, y) sprintf("%s = %s", x, y)) +
+         geom_line(size = 2) + theme_bw() +
+         scale_x_continuous(limits = c(-3, 50)) +
+         scale_y_continuous(limits = c(0, 0.9))
>
> direct.label(gg, list("top.bumptwice", dl.trans(y = y + 0.2)))
```

The results shown in this plot are roughly similar to those obtained from the nested dichotomy models, graphed in Figure 8.10. However, the predicted probabilities of not working under the generalized logit model rise more steeply with husband's income for women with no children and level off sooner for women with young children.

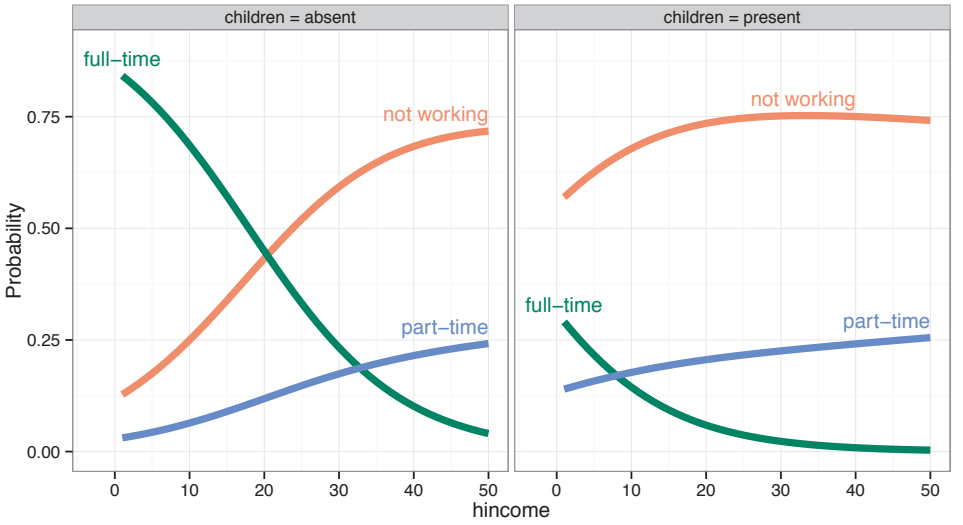


Figure 8.12: Fitted probabilities from the generalized logit model fit to the data on women's labor force participation.

The effects package has special methods for "multinom" models. It treats the response levels in the order given by `levels()`, so before plotting we use `ordered()` to arrange levels in their natural order. The `update()` method provides a simple way to get a new fitted model; in the call, the model formula `. ~ .` means to fit the same model as before, i.e., `partic ~ hincome + children`.

```
> levels(Womenlf$partic)

[1] "not.work" "fulltime" "parttime"

> Womenlf$partic <- ordered(Womenlf$partic,
+                           levels=c("not.work", "parttime", "fulltime"))
> wlf.multinom <- update(wlf.multinom, . ~ .)

# weights:  12 (6 variable)
initial  value 288.935032
iter  10 value 211.454772
final  value 211.440963
converged
```

As illustrated earlier, you can use `plot(allEffects(model), ...)` to plot all the high-order terms in the model, either with separate curves for each response level (`style="lines"`) or as cumulative filled polygons (`style="stacked"`). Here, we simply plot the effects for the combinations of husband's income and children in stacked style, giving a plot (Figure 8.13) that is analogous to the full-model plot shown in Figure 8.12.

```
> plot(Effect(c("hincome", "children"), wlf.multinom),
+       style = "stacked", key.args = list(x = .05, y = .9))
```

△

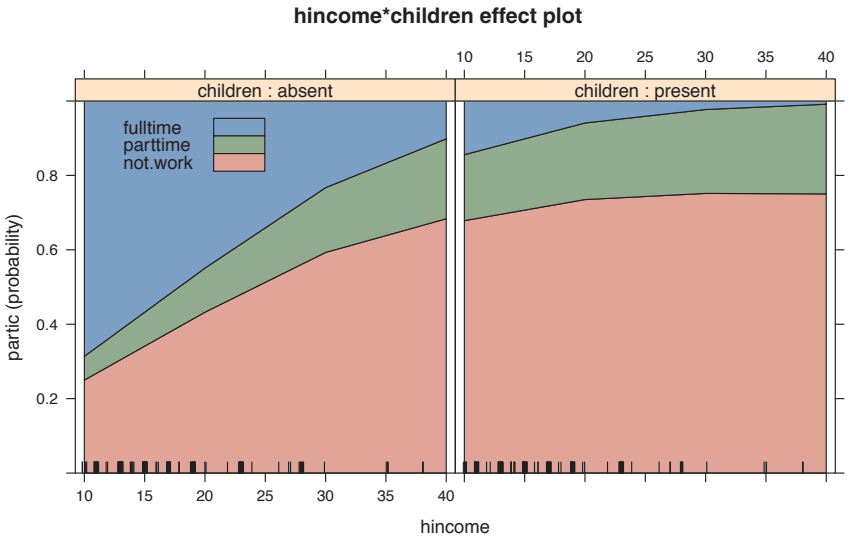


Figure 8.13: Effect plot for the probabilities of not working and working part time and full time from the generalized logit model fit to the women's labor force data.

8.4 Chapter summary

- Polytomous responses may be handled in several ways as extensions of binary logistic regression. These methods require different fitting functions in R; however, the graphical methods for plotting results are relatively straightforward extensions of those used for binary responses.
- The *proportional odds model* (Section 8.1) is simple and convenient, but its validity depends on an assumption of equal slopes for adjacent-category logits.
- *Nested dichotomies* (Section 8.2) among the response categories give a set of statistically independent binary logistic submodels. These may be regarded as a single combined model for the polytomous response.
- *Generalized logit models* (Section 8.3) provide the most general approach. These may be used to construct submodels comparing any pair of categories.

8.5 Lab exercises

Exercise 8.1 For the women's labor force participation data (*Womenlf*), the response variable, `partic`, can be treated as ordinal by using

```
> Womenlf$partic <- ordered(Womenlf$partic,
+                           levels=c('not.work', 'parttime', 'fulltime'))
```

Use the methods in Section 8.1 to test whether the proportional odds model holds for these data.

Exercise 8.2 The data set *housing* in the MASS package gives a $3 \times 3 \times 4 \times 2$ table in frequency form relating (a) satisfaction (`Sat`) of residents with their housing (High, Medium, Low), (b) perceived degree of influence (`Infl`) they have on the management of the property (High, Medium, Low), (c) `Type` of rental (Tower, Atrium, Apartment, Terrace), and (d) contact (`Cont`) residents have with other residents (Low, High). Consider satisfaction as the ordinal response variable.

(a) Fit the proportional odds model with additive (main) effects of housing type, influence in management, and contact with neighbors to this data. (Hint: Using `polr()`, with the data in frequency form, you need to use the `weights` argument to supply the `Freq` variable.)
(b) Investigate whether any of the two-factor interactions among `Infl`, `Type`, and `Cont` add substantially to goodness of fit of this model. (Hint: use `stepAIC()`, with the scope formula `~ .^2` and `direction="forward"`.)
(c) For your chosen model from the previous step, use the methods of Section 8.1.5 to plot the probabilities of the categories of satisfaction.
(d) Write a brief summary of these analyses, interpreting *how* satisfaction with housing depends on the predictor variables.

Exercise 8.3 The data *TV* on television viewing was analyzed using correspondence analysis in Example 6.4, ignoring the variable `Time`, and extended in Exercise 6.9. Treating `Network` as a three-level response variable, fit a generalized logit model (Section 8.3) to explain the variation in viewing in relation to `Day` and `Time`. The *TV* data is a three-way table, so you will need to convert it to a frequency data frame first.

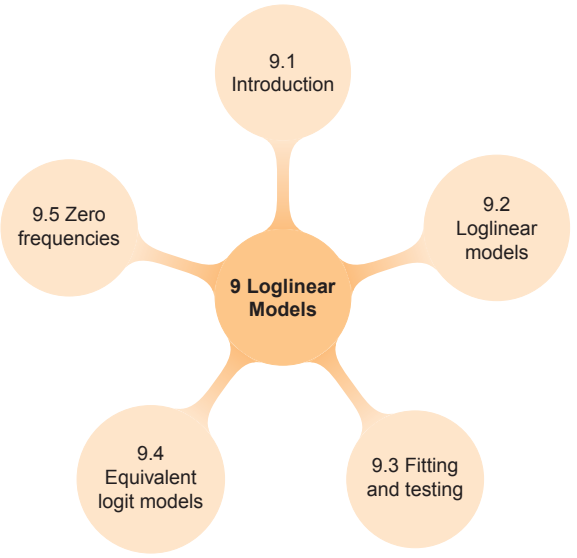
```
> data("TV", package="vcdExtra")
> TV.df <- as.data.frame.table(TV)
```

(a) Fit the main-effects model, `Network ~ Day + Time`, with `multinom()`. Note that you will have to supply the `weights` argument because each row of `TV.df` represents the number of viewers in the `Freq` variable.
(b) Prepare an effects plot for the fitted probabilities in this model.
(c) Interpret these results in comparison to the correspondence analysis in Example 6.4.

Exercise 8.4 * Refer to Exercise 5.10 for a description of the *Vietnam* data set in vcdExtra. The goal here is to fit models for the polytomous `response` varialble in relation to `year` and `sex`.

(a) Fit the proportional odds model to these data, allowing an interaction of `year` and `sex`.
(b) Is there evidence that the proportional odds assumption does not hold for this data set? Use the methods described in Section 8.1 to assess this.
(c) Fit the multinomial logistic model, also allowing an interaction. Use car::`Anova()` to assess the model terms.
(d) Produce an effect plot for this model and describe the nature of the interaction.
(e) Fit the simpler multinomial model in which there is no effect of year for females and the effect of year is linear for males (on the logit scale). Test whether this model is significantly worse than the general multinomial model with interaction.

This page intentionally left blank



9

Loglinear and Logit Models for Contingency Tables

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

> Numbers have an important story to tell. They rely on you to give them a clear and convincing voice
>
> Stephen Few

9.1 Introduction

The chapter continues the modeling framework begun in Chapter 7, and takes up the case of loglinear models for contingency tables of frequencies, when all variables are discrete, another special case of generalized linear models. These models provide a comprehensive scheme to describe and understand the associations among two or more categorical variables. Whereas logistic regression models focus on the prediction of one response factor, loglinear models treat all variables symmetrically, and attempt to model all important associations among them.

In this sense, loglinear models are analogous to a correlation analysis of continuous variables, where the goal is to determine the patterns of dependence and independence among a set of variables. When one variable is a response and the others are explanatory, certain loglinear models

are equivalent to logistic models for that response. Such models are also particularly useful when there are two or more response variables, a case that would require a multivariate version of the generalized linear model, for which the current theory and implementations are thin at best.

Chapter 5 and Chapter 6 introduced some basic aspects of loglinear models in connection with mosaic displays and correspondence analysis. In this chapter, the focus is on fitting and interpreting loglinear models. The usual analyses with `loglm()` and `glm()` present the results in terms of tables of parameter estimates. Particularly for larger tables, it becomes difficult to understand the nature of these associations from tables of parameter estimates. Instead, we emphasize plots of observed and predicted frequencies, probabilities or log odds (when there are one or more response variables), as well as mosaic and other displays for interpreting a given model. We also illustrate how mosaic displays and correspondence analysis plots may be used in a complementary way to the usual numerical summaries, to provide additional insights into the data.

Section 9.2 gives a brief overview of loglinear models in relation to the more familiar ANOVA and regression models for quantitative data. Methods and software for fitting these models are discussed in Section 9.3. When one variable is a response, logit models for that response provide a simpler, but equivalent means for interpreting and graphing results of loglinear models, as we describe in Section 9.4. In Section 9.5 we consider problems that arise in sparce contingency tables containing cells with frequencies of zero.

9.2 Loglinear models for frequencies

Loglinear models have been developed from two formally distinct, but related perspectives. The first is a discrete analog of familiar ANOVA models for quantitative data, where the multiplicative relations among joint and marginal probabilities are transformed into an additive one by transforming the counts to logarithms. The second is an analog of regression models, where the log of the cell frequency is modeled as a linear function of discrete predictors, with a random component often taken as the Poisson distribution and called ***Poisson regression***; this approach is treated in more detail as generalized linear models for count data in Chapter 11.

9.2.1 Loglinear models as ANOVA models for frequencies

For two discrete variables, A and B, suppose we have a multinomial sample of n_{ij} observations in each cell i, j of an $I \times J$ contingency table. To ease notation, we replace a subscript by $+$ to represent summation over that dimension, so that $n_{i+} = \Sigma_j n_{ij}$, $n_{+j} = \Sigma_i n_{ij}$, and $n_{++} = \Sigma_{ij} n_{ij}$.

Let π_{ij} be the joint probabilities in the table, and let $m_{ij} = n_{++}\pi_{ij}$ be the expected cell frequencies under any model. Conditional on the observed total count, n_{++}, each count has a Poisson distribution, with mean m_{ij}. Any loglinear model may be expressed as a linear model for the $\log m_{ij}$. For example, the hypothesis of independence means that the expected frequencies, m_{ij}, obey

$$m_{ij} = \frac{m_{i+}\, m_{+j}}{m_{++}} \, .$$

This multiplicative model can be transformed to an additive (linear) model by taking logarithms of both sides:

$$\log(m_{ij}) = \log(m_{i+}) + \log(m_{+j}) - \log(m_{++}) \, ,$$

which is usually expressed in an equivalent form in terms of model parameters,

$$\log(m_{ij}) = \mu + \lambda_i^A + \lambda_j^B \tag{9.1}$$

where μ is a function of the total sample size, λ_i^A is the "main effect" for variable A, $\lambda_i^A = \log \pi_{i+} - \sum_k (\log \pi_{k+})/I$, and λ_j^B is the "main effect" for variable B, $\lambda_j^B = \log \pi_{+j} - \sum_k (\log \pi_{+k})/J$. Model Eqn. (9.1) is called the ***loglinear independence model*** for a two-way table.

In this model, there are $1+I+J$ parameters, but only $(I-1)+(J-1)$ are separately estimable. Hence, the typical ANOVA sum-to-zero restrictions are usually applied to the parameters:

$$\sum_i^I \lambda_i^A = \sum_j^J \lambda_j^B = 0 \, .$$

These "main effects" in loglinear models pertain to differences among the marginal probabilities of a variable (which are usually not of direct interest).

Other restrictions to make the parameters identifiable are also used. Setting the first values, λ_1^A and λ_1^B, to zero (the default in `glm()`), defines $\lambda_i^A = \log \pi_{i+} - \log \pi_{1+}$, and $\lambda_j^B = \log \pi_{+j} - \log \pi_{+1}$, as deviations from the first, reference category, but these parameterizations are otherwise identical. For modeling functions in R (`lm()`, `glm()`, etc.) the reference category parameterization is obtained using `contr.treatment()`, while the sum-to-zero constraints are obtained with `contr.sum()`.

Model Eqn. (9.1) asserts that the row and column variables are independent. For a two-way table, a model that allows an arbitrary association between the variables is the ***saturated model***, including an additional term, λ_{ij}^{AB}:

$$\log(m_{ij}) = \mu + \lambda_i^A + \lambda_j^B + \lambda_{ij}^{AB} \, , \tag{9.2}$$

where again, restrictions must be imposed for estimation:

$$\sum_i^I \lambda_i^A = 0, \quad \sum_j^J \lambda_j^B = 0, \quad \sum_i^I \lambda_{ij}^{AB} = \sum_j^J \lambda_{ij}^{AB} = 0 \, . \tag{9.3}$$

There are thus $I-1$ linearly independent λ_i^A row parameters, $J-1$ linearly independent λ_j^B column parameters, and $(I-1)(J-1)$ linearly independent λ_{ij}^{AB} association parameters. This model is called the *saturated model* because the number of parameters in μ, λ_i^A, λ_j^B, and λ_{ij}^{AB} is equal to the number of frequencies in the two-way table,

$$\underset{(\mu)}{1} + \underset{(\lambda_i^A)}{(I-1)} + \underset{(\lambda_j^B)}{(J-1)} + \underset{(\lambda_{ij}^{AB})}{(I-1)(J-1)} = \underset{(n_{ij})}{IJ} \, .$$

The association parameters λ_{ij}^{AB} express the departures from independence, so large absolute values pertain to cells that differ from the independence model.

Except for the difference in notation, model Eqn. (9.2) is formally the same as a two-factor ANOVA model with an interaction, typically expressed as $E(y_{ij}) = \mu + \alpha_i + \beta_j + (\alpha\beta)_{ij}$. Hence, associations between variables in loglinear models are analogous to interactions in ANOVA models. The use of superscripted symbols, $\lambda_i^A, \lambda_j^B, \lambda_{ij}^{AB}$, rather than separate Greek letters is a convention in loglinear models, and useful mainly for multiway tables.

Models such as Eqn. (9.1) and Eqn. (9.2) are examples of ***hierarchical models***. This means that the model must contain all lower-order terms contained within any high-order term in the model. Thus, the saturated model in Eqn. (9.2) contains λ_{ij}^{AB}, and therefore *must* contain λ_i^A and λ_j^B. As a result, hierarchical models may be identified by the shorthand notation that lists only the high-order terms: model Eqn. (9.2) is denoted $[AB]$, while model Eqn. (9.1) is $[A][B]$.

9.2.2 Loglinear models for three-way tables

Loglinear models for three-way contingency tables were described briefly in Section 5.4.2. Each type of model allows associations among different sets of variables and each has a different independence interpretation, as illustrated in Table 5.2.

For a three-way table, the saturated model, denoted $[ABC]$ is

$$\log m_{ijk} = \mu + \lambda_i^A + \lambda_j^B + \lambda_k^C + \lambda_{ij}^{AB} + \lambda_{ik}^{AC} + \lambda_{jk}^{BC} + \lambda_{ijk}^{ABC} \; . \tag{9.4}$$

This model allows all variables to be associated; Eqn. (9.4) fits the data perfectly because the number of independent parameters equals the number of table cells. Two-way terms, such as λ_{ij}^{AB}, pertain to the *conditional association* between pairs of factors, controlling for the remaining variable. The presence of the three-way term, λ_{ijk}^{ABC}, means that the partial association (conditional odds ratio) between any pair varies over the levels of the third variable.

Omitting the three-way term in Model Eqn. (9.4) gives the model $[AB][AC][BC]$,

$$\log m_{ijk} = \mu + \lambda_i^A + \lambda_j^B + \lambda_k^C + \lambda_{ij}^{AB} + \lambda_{ik}^{AC} + \lambda_{jk}^{BC} \; , \tag{9.5}$$

in which all pairs are conditionally dependent given the remaining one. For any pair, the conditional odds ratios are the *same* at all levels of the remaining variable, so this model is often called the ***homogeneous association model***.

The interpretation of terms in this model may be illustrated using the Berkeley admissions data (Example 4.11 and Example 4.15), for which the factors are Admit, Gender, and Department, in a $2 \times 2 \times 6$ table. In the homogeneous association model,

$$\log m_{ijk} = \mu + \lambda_i^A + \lambda_j^D + \lambda_k^G + \lambda_{ij}^{AD} + \lambda_{ik}^{AG} + \lambda_{jk}^{DG} \; , \tag{9.6}$$

the λ-parameters have the following interpretations:

- The main effects, λ_i^A, λ_j^D, and λ_k^G pertain to differences in the one-way marginal probabilities. Thus λ_j^D relates to differences in the total number of applicants to these departments, while λ_k^G relates to the differences in the overall numbers of men and women applicants.
- λ_{ij}^{AD} describes the conditional association between admission and department, that is, different admission rates across departments (controlling for gender).
- λ_{ik}^{AG} relates to the conditional association between admission and gender, controlling for department. This term, if significant, might be interpreted as indicating gender-bias in admissions.
- λ_{jk}^{DG}, the association between department and gender, indicates whether males and females apply differentially across departments.

As we discussed earlier (Section 5.4), loglinear models for three-way (and larger) tables often have an interpretation in terms of various types of independence relations, as illustrated in Table 5.2. The model Eqn. (9.5) has no such interpretation. However the smaller model $[AC][BC]$ can be interpreted as asserting that A and B are (conditionally) independent controlling for C; this independence interpretation is symbolized as $A \perp B \,|\, C$. Similarly, the model $[AB][C]$ asserts that A and B are jointly independent of C: $(A, B) \perp C$, while the model $[A][B][C]$ is the model of mutual (complete) independence, $A \perp B \perp C$.

9.2.3 Loglinear models as GLMs for frequencies

In the GLM approach, a loglinear model may be cast in the form of a regression model for $\log \boldsymbol{m}$, where the table cells are reshaped to a column vector. One advantage is that models for tables of any size and structure may be expressed in a compact form.

For a contingency table of variables $A, B, C, \cdots$, with $N = I \times J \times K \times \cdots$ cells, let $\boldsymbol{n}$ denote a column vector of the observed counts arranged in standard order, and let $\boldsymbol{m}$ denote a similar vector of the expected frequencies under some model. Then *any* loglinear model may be expressed in the form

$$\log \boldsymbol{m} = \boldsymbol{X}\boldsymbol{\beta} \, ,$$

where $\boldsymbol{X}$ is a known design or ***model matrix*** and $\boldsymbol{\beta}$ is a column vector containing the unknown λ parameters.

For example, for a 2×2 table, the saturated model Eqn. (9.2) with the usual zero-sum constraints Eqn. (9.3) can be represented as

$$\log \begin{pmatrix} m_{11} \\ m_{12} \\ m_{21} \\ m_{22} \end{pmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & 1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix} \begin{pmatrix} \mu \\ \lambda_1^A \\ \lambda_1^B \\ \lambda_{11}^{AB} \end{pmatrix} .$$

Note that only the linearly independent parameters are represented here. $\lambda_2^A = -\lambda_1^A$, because $\lambda_1^A + \lambda_2^A = 0$, and $\lambda_2^B = -\lambda_1^B$, because $\lambda_1^B + \lambda_2^B = 0$, and so forth.

An additional substantial advantage of the GLM formulation is that it makes it easier to express models with ordinal or quantitative variables. `glm()`, with a model formula of the form `Freq ~ .` involving factors $A, B, \ldots$ and quantitative variables $x_1, x_2, \ldots$, constructs the model matrix $\boldsymbol{X}$ from the terms given in the formula. A factor with K levels gives rise to $K - 1$ columns for its main effect and sets of $K - 1$ columns in each interaction effect. A quantitative predictor, say, x_1 (with a linear effect) creates a single column with its values, and interactions with other terms are calculated at the products of the columns for the main effects.

The parameterization for factors is controlled by the contrasts assigned to a given factor (if any), or by the general `contrasts` option, that gives the contrast functions used for unordered and ordered factors:

```
> options("contrasts")

$contrasts
        unordered           ordered
"contr.treatment"      "contr.poly"
```

This says that, by default, unordered factors use the baseline (first) reference-level parameterization, while ordered factors are given a parameterization based on orthogonal polynomials, allowing linear, quadratic, ... effects, assuming integer-spacing of the factor levels.

9.3 Fitting and testing loglinear models

For a given table, possible loglinear models range from the baseline model of mutual independence, $[A][B][C][\ldots]$ to the saturated model, $[ABC\ldots]$ that fits the observed frequencies perfectly, but offers no simpler description or interpretation than the data itself.

Fitting a loglinear model is usually a process of deciding which association terms are large enough ("significantly different from zero") to warrant inclusion in a model to explain the observed frequencies. Terms that are excluded from the model go into the residual or error term, which reflects the overall badness-of-fit of the model. The usual goal of loglinear modeling is to find a small model (few association terms), which nonetheless achieves a reasonable fit (small residuals).

9.3.1 Model fitting functions

In R, the most basic function for fitting loglinear models is `loglin()` in the stats package. This uses the classical iterative proportional fitting (IPF) algorithm described in Haberman (1972) and

Fienberg (1980, Section 3.4). It is designed to work with the frequency data in table form, and a model specified in terms of the (high-order) table margins to be fitted. For example, the model Eqn. (9.5) of homogenous association for a three-way table is specified as

```
> loglin(mytable, margin = list(c(1, 2), c(1, 3), c(2, 3)))
```

The variables are represented by their margin index; margins combined in the same vector represent an interaction term. The function `loglm()` in MASS provides a more convenient front-end to `loglin()` to allow loglinear models to be specified using a model formula. With table variables `A`, `B`, and `C`, the same model can be fit using `loglm()` as

```
> loglm(~ (A + B + C)^2, data = mytable)
```

(Note that the formula expression expands to `A*A + A*B + A*C + B*A + B*B + B*C + C*A + C*B + C*C`. Since terms like `A*A` become `A` and duplicate terms are ignored, this eventually yields `A + B + C + A*B + A*C + B*C`—all second-order terms and the corresponding main effects.)

When the data is a frequency data frame with frequencies in `Freq`, for example, the result of `mydf <- as.data.frame(mytable)`, you can also use a two-sided formula:

```
> loglm(Freq ~ (A + B + C)^2, data = mydf)
```

As implied in Section 9.2.3, loglinear models can also be fit using `glm()`, using `family=poisson`, which constructs the model for `log(Freq)`. The same model is fit with `glm()` as:

```
> glm(Freq ~ (A + B + C)^2, data = mydf, family = poisson)
```

While all of these fit equivalent models, the details of the printed output, model objects, and available methods differ, as indicated in some of the examples that follow.

It should be noted that both the `loglin()`/`loglm()` methods based on iterative proportional fitting, and the `glm()` approach using the Poisson model for log frequency, give maximum likelihood estimates, $\widehat{\boldsymbol{m}}$, of the expected frequencies, as long as all observed frequencies $\boldsymbol{n}$ are *all* positive. Some special considerations when there are cells with zero frequencies are described in Section 9.5.

9.3.2 Goodness-of-fit tests

For an n-way table, global goodness-of-fit tests for a loglinear model attempt to answer the question, "How well does the model reproduce the observed frequencies?" That is, how close are the fitted frequencies estimated under the model to those of the saturated model or the data?

To avoid multiple subscripts for an n-way table, let $\boldsymbol{n} = (n_1, n_2, \ldots, n_N)$ denote the observed frequencies in a table with N cells, and corresponding fitted frequencies $\widehat{\boldsymbol{m}} = (\widehat{m}_1, \widehat{m}_2, \ldots, \widehat{m}_N)$ according to a particular loglinear model. The standard goodness-of-fit statistics are sums over the cells of measures of the difference between the $\boldsymbol{n}$ and $\widehat{\boldsymbol{m}}$.

The most commonly used are the familiar Pearson chi-square,

$$X^2 = \sum_i^N \frac{(n_i - \widehat{m}_i)^2}{\widehat{m}_i} \,, \tag{9.7}$$

and the likelihood-ratio G^2 or ***deviance*** statistic,

$$G^2 = 2 \sum_i^N n_i \, \log \left(\frac{n_i}{\widehat{m}_i} \right) \,. \tag{9.8}$$

Both of these statistics have asymptotic χ^2 distributions (as $\Sigma \boldsymbol{n} \rightarrow \infty$), reasonably well-approximated when all expected frequencies are large.[1] The (residual) degrees of freedom are the number of cells (N) minus the number of estimated parameters. The likelihood-ratio test can also be expressed as twice the difference in log-likelihoods under saturated and fitted models,

$$G^2 = 2 \log \left[\frac{\mathcal{L}(\boldsymbol{n}; \boldsymbol{n})}{\mathcal{L}(\widehat{\boldsymbol{m}}; \boldsymbol{n})} \right] = 2[\log \mathcal{L}(\boldsymbol{n}; \boldsymbol{n}) - \log \mathcal{L}(\widehat{\boldsymbol{m}}; \boldsymbol{n})] \, ,$$

where $\mathcal{L}(\boldsymbol{n}; \boldsymbol{n})$ is the likelihood for the saturated model and $\mathcal{L}(\widehat{\boldsymbol{m}}; \boldsymbol{n})$ is the corresponding maximized likelihood for the fitted model.

In practice such global tests are less useful for comparing competing models. You may find that several different models have an acceptable fit or, sadly, that none do (usually because you are "blessed" with a large sample size). It is then helpful to compare competing models *directly*, and two strategies are particularly useful in these cases.

First, the likelihood-ratio G^2 statistic has the property in that one can compare two ***nested models*** by their difference in G^2 statistics, which has a χ^2 distribution on the difference in degrees of freedom. Two models, M_1 and M_2, are nested when one, say, M_2, is a special case of the other. That is, model M_2 (with ν_2 residual df) contains a subset of the parameters of M_1 (with ν_1 residual df), the remaining ones being effectively set to zero. Model M_2 is therefore more restrictive and cannot fit the data better than the more general model M_1, i.e., $G^2(M_2) \geq G^2(M_1)$. The least restrictive of all models, with $G^2 = 0$ and $\nu = 0$ df, is the saturated model for which $\widehat{\boldsymbol{m}} = \boldsymbol{n}$.

Assuming that the less restrictive model M_1 fits, the difference in G^2,

$$\begin{aligned} \Delta G^2 \equiv G^2(M_2 \,|\, M_1) &= G^2(M_2) - G^2(M_1) && (9.9) \\ &= 2 \sum_i n_i \, \log(\widehat{m}_{i1} / \widehat{m}_{i2}) && (9.10) \end{aligned}$$

has a chi-squared distribution with df = $\nu_2 - \nu_1$. The last equality, Eqn. (9.10), follows from substituting in Eqn. (9.8).

Rearranging terms in Eqn. (9.9), we see that we can partition the $G^2(M_2)$ into two terms,

$$G^2(M_2) = G^2(M_1) + G^2(M_2 \,|\, M_1) \, .$$

The first term measures the difference between the data and the more general model M_1. If this model fits, the second term measures the additional lack of fit imposed by the more restrictive model. In addition to providing a more focused test, $G^2(M_2 \,|\, M_1)$ also follows the chi-squared distribution more closely when some $\{m_i\}$ are small (Agresti, 2013, Section 10.6.3).

Alternatively, a second strategy uses other measures that combine goodness-of-fit with model parsimony and may also be used to compare non-nested models. The statistics described below are all cast in the form of badness-of-fit relative to degrees of freedom, so that smaller values reflect "better" models.

The simplest idea (Goodman, 1971) is to use G^2/df (or χ^2/df), which has an asymptotic expected value of 1 for a good-fitting model. This type of measure is not routinely reported by R software, but is easy to calculate from output.

The ***Akaike Information Criterion*** (AIC) statistic (Akaike, 1973) is a very general criterion for model selection with maximum likelihood estimation, based on the idea of maximizing the information provided by a fitted model. AIC is defined generally as

$$\text{AIC} = -2 \log \mathcal{L} + 2k \, ,$$

[1] Except in bizarre or borderline cases, these tests provide the same conclusions when expected frequencies are at least moderate (all $\widehat{\boldsymbol{m}} > 5$). However, G^2 approaches the theoretical chi-squared distribution more slowly than does χ^2, and the approximation may be poor when the average cell frequency is less than 5.

where $\log \mathcal{L}$ is the maximized log likelihood and k is the number of parameters estimated in the model. Better models correspond to *smaller* AIC. For loglinear models, minimizing AIC is equivalent to minimizing

$$\text{AIC}^\star = G^2 - 2\,\nu\,,$$

where ν is the residual df, but the values of AIC and AIC* differ by an arbitrary constant. This form is easier to calculate by hand from the output of any modeling function if AIC is not reported, or an `AIC()` method is not available.

A third statistic of this type is the ***Bayesian Information Criterion*** (BIC) due to Schwartz (1978) and Raftery (1986),

$$\text{BIC} = G^2 - \log(n)\,\nu\,,$$

where n is the total sample size. Both AIC and BIC penalize the fit statistic for increasing number of parameters. BIC also penalizes the fit directly with (log) sample size, and so expresses a preference for less complex models than AIC as the sample size increases.

9.3.3 Residuals for loglinear models

Test statistics such as G^2 can determine whether a model has significant lack of fit, and model comparison tests using $\Delta G^2 = G^2(M_2 \,|\, M_1)$ can assess whether the extra term(s) in model M_1 significantly improves the model fit. Beyond these tests, the pattern of residuals for individual cells offers important clues regarding the nature of lack of fit and can suggest associations that could be accounted for better.

As with logistic regression models (Section 7.5.1), several types of residuals are available for loglinear models. For cell i in the vector form of the contingency table, the ***raw residual*** is simply the difference between the observed and fitted frequencies, $e_i = n_i - \widehat{m}_i$.

The ***Pearson residual*** is the square root of the contribution of the cell to the Pearson χ^2,

$$r_i = \frac{n_i - \widehat{m}_i}{\sqrt{\widehat{m}_i}}\,. \tag{9.11}$$

Similarly, the ***deviance residual*** can be defined as

$$g_i = \text{sign}(n_i - \widehat{m}_i)\sqrt{2n_i \log(n_i/\widehat{m}_i) - 2(n_i - \widehat{m}_i)}\,. \tag{9.12}$$

Both of these attempt to standardize the distribution of the residuals to a standard normal, $N(0, 1)$ form. However, as pointed out by Haberman (1973), the asymptotic variance of these is less than one (with average value df/N) but, worse—the variance decreases with $\widehat{m}_i$. That is, residuals for cells with small expected frequencies have larger sampling variance as might be expected.

Consequently, Haberman suggested dividing the Pearson residual by its estimated standard error, giving what are often called ***adjusted residual***s. When loglinear models are fit using the GLM approach, the adjustment may be calculated using the leverage ("hat value"), h_i to give appropriately standardized residuals,

$$\begin{aligned} r_i^\star &= r_i/\sqrt{1 - h_i}\,, \\ g_i^\star &= g_i/\sqrt{1 - h_i}\,. \end{aligned}$$

These standardized versions are generally preferable, particularly for visualizing model lack of fit using mosaic displays. The reason for preferring adjusted residuals is illustrated in Figure 9.1, a plot of the factors, $\sqrt{1 - h_i}$, determining the standard errors of the residuals against the fitted values, $\widehat{m}_i$, in the model for the `UCBAdmissions` data described in Example 9.2 below. The values shown in this plot are calculated as:

```
> berkeley <- as.data.frame(UCBAdmissions)
> berk.glm1 <- glm(Freq ~ Dept * (Gender + Admit), data = berkeley,
+                  family = "poisson")
> fit <- fitted(berk.glm1)
> hat <- hatvalues(berk.glm1)
> stderr <- sqrt(1 - hat)
```

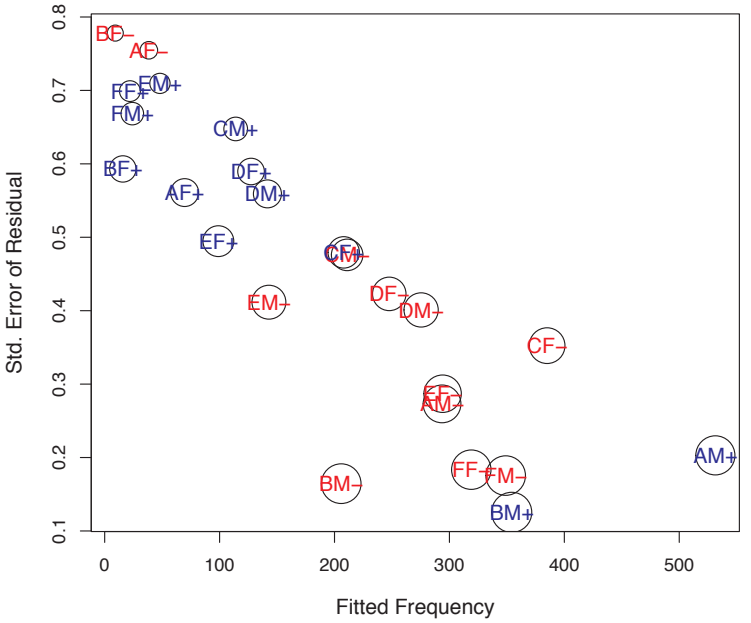


Figure 9.1: Standard errors of residuals, $\sqrt{1-h_i}$ decrease with expected frequencies. This plot shows why ordinary Pearson and deviance residuals may be misleading. The symbol size in the plot is proportional to leverage, h_i. Labels abbreviate Department, Gender, and Admit, colored by Admit.

In R, raw, Pearson and deviance residuals may be obtained using `residuals(model, type=)`, where `type` is one of `"raw"`, `"pearson"`, and `"deviance"`. Standardized (adjusted) residuals can be calculated using `rstandard(model, type=)`, for `type="pearson"` and `type="deviance"` versions.

9.3.4 Using `loglm()`

Here we illustrate the basics of fitting loglinear models using `loglm()`. As indicated in Section 9.3.1, the model to be fitted is specified by a model formula involving the table variables. The MASS package provides a `coef()` method for "loglm" objects that extracts the estimated parameters and a `residuals()` method that calculates various types of residuals according to a `type` argument, one of `"deviance"`, `"pearson"`, `"response"`. vcd and vcdExtra provide a variety of plotting methods, including `assoc()`, `sieve()`, `mosaic()`, and `mosaic3d()` for "loglm" objects.

EXAMPLE 9.1: Berkeley admissions

The *UCBAdmissions* on admissions to the six largest graduate departments at U.C. Berkeley was examined using graphical methods in Chapter 4 (Example 4.15) and in Chapter 5 (Example 5.14). We can fit and compare several loglinear models as shown below.

The model of mutual independence, $[A][D][G]$, is not substantively reasonable here, because the association of `Dept` and `Gender` should be taken into account to control for these variables, but we show it here to illustrate the form of the printed output, giving the Pearson χ^2 and likelihood-ratio G^2 tests of goodness of fit, as well as some optional arguments for saving additional components in the result.

```
> data("UCBAdmissions")
> library(MASS)
> berk.loglm0 <- loglm(~ Dept + Gender + Admit, data = UCBAdmissions,
+                      param = TRUE, fitted = TRUE)
> berk.loglm0

Call:
loglm(formula = ~Dept + Gender + Admit, data = UCBAdmissions,
    param = TRUE, fitted = TRUE)

Statistics:
                     X^2 df P(> X^2)
Likelihood Ratio 2097.7 16        0
Pearson          2000.3 16        0
```

The argument `param = TRUE` stores the estimated parameters in the loglinear model and `fitted = TRUE` stores the fitted frequencies $\widehat{m}_{ijk}$. The fitted frequencies can be extracted from the model object using `fitted()`.

```
> structable(Dept ~ Admit + Gender, fitted(berk.loglm0))

                Dept      A      B      C      D      E      F
Admit    Gender
Admitted Male        215.10 134.87 211.64 182.59 134.64 164.61
         Female      146.68  91.97 144.32 124.51  91.81 112.25
Rejected Male        339.63 212.95 334.17 288.30 212.59 259.91
         Female      231.59 145.21 227.87 196.59 144.96 177.23
```

Similarly, you can extract the estimated parameters with `coef(berk.loglm0)`, and the Pearson residuals with `residuals(berk.loglm0, type = "pearson")`.

Next, consider the model of conditional independence of gender and admission given department, $[AD][GD]$, which allows associations of admission with department and gender with department.

```
> # conditional independence in UCB admissions data
> berk.loglm1 <- loglm(~ Dept * (Gender + Admit), data = UCBAdmissions)
> berk.loglm1

Call:
loglm(formula = ~Dept * (Gender + Admit), data = UCBAdmissions)

Statistics:
                     X^2 df   P(> X^2)
Likelihood Ratio 21.736  6 0.0013520
Pearson          19.938  6 0.0028402
```

Finally, for this example, the model of homogeneous association, $[AD][AG][GD]$, can be fit as follows.[2]

[2]It is useful to note here that the added term $[AG]$ allows a general association of admission with gender (controlling for department). A significance test for this term, or for model `berk.loglm2` against `berk.loglm1`, is a proper test for the assertion of gender bias in admissions.

```
> berk.loglm2 <-loglm(~ (Admit + Dept + Gender)^2, data = UCBAdmissions)
> berk.loglm2

Call:
loglm(formula = ~(Admit + Dept + Gender)^2, data = UCBAdmissions)

Statistics:
                     X^2 df  P(> X^2)
Likelihood Ratio 20.204  5 0.0011441
Pearson          18.823  5 0.0020740
```

Neither of these models fits particularly well, as judged by the goodness-of-fit Pearson χ^2 and likelihood-ratio G^2 test against the saturated model. The `anova()` method for a nested collection of "loglm" models gives a series of likelihood-ratio tests of the difference, ΔG^2, between each sequential pair of models, according to Eqn. (9.9).

```
> anova(berk.loglm0, berk.loglm1, berk.loglm2, test = "Chisq")

LR tests for hierarchical log-linear models

Model 1:
 ~Dept + Gender + Admit
Model 2:
 ~Dept * (Gender + Admit)
Model 3:
 ~(Admit + Dept + Gender)^2

          Deviance df Delta(Dev) Delta(df) P(> Delta(Dev)
Model 1   2097.671 16
Model 2     21.736  6  2075.9357        10        0.00000
Model 3     20.204  5     1.5312         1        0.21593
Saturated    0.000  0    20.2043         5        0.00114
```

The conclusion from these results is that the model `berk.loglm1` is not much worse than model `berk.loglm2`, but there is still significant lack of fit. The next example, using `glm()`, shows how to visualize the lack of fit and account for it.

△

9.3.5 Using `glm()`

Loglinear models fit with `glm()` require the data in a data frame in frequency form, for example as produced by `as.data.frame()` from a table. The model formula expresses the model for the frequency variable, and uses `family = poisson` to specify the error distribution. More general distributions for frequency data are discussed in Chapter 11.

EXAMPLE 9.2: Berkeley admissions

For the $2 \times 2 \times 6$ *UCBAdmissions* table, first transform this to a frequency data frame:

```
> berkeley <- as.data.frame(UCBAdmissions)
> head(berkeley)

     Admit Gender Dept Freq
1 Admitted   Male    A  512
2 Rejected   Male    A  313
3 Admitted Female    A   89
4 Rejected Female    A   19
5 Admitted   Male    B  353
6 Rejected   Male    B  207
```

Then, the model of conditional independence corresponding to `berk.loglm1` can be fit using `glm()` as shown below.

```
> berk.glm1 <- glm(Freq ~ Dept * (Gender + Admit),
+                  data = berkeley, family = "poisson")
```

Similarly, the all two-way model of homogeneous association is fit using

```
> berk.glm2 <- glm(Freq ~ (Dept + Gender + Admit)^2,
+                  data = berkeley, family = "poisson")
```

These models are equivalent to those fit using `loglm()` in Example 9.1. We get the same residual G^2 as before, and the likelihood-ratio test of ΔG^2 given by `anova()` gives the same result, that the model `berk.glm2` offers no significant improvement over model `berk.glm1`.

```
> anova(berk.glm1, berk.glm2, test = "Chisq")

Analysis of Deviance Table

Model 1: Freq ~ Dept * (Gender + Admit)
Model 2: Freq ~ (Dept + Gender + Admit)^2
  Resid. Df Resid. Dev Df Deviance Pr(>Chi)
1         6       21.7
2         5       20.2  1     1.53     0.22
```

Among other advantages of using `glm()` as opposed to `loglm()` is that an `anova()` method is available for *individual* "glm" models, giving significance tests of the contributions of each *term* in the model, as opposed to the tests for individual coefficients provided by `summary()`.[3]

```
> anova(berk.glm1, test = "Chisq")

Analysis of Deviance Table

Model: poisson, link: log

Response: Freq

Terms added sequentially (first to last)


            Df Deviance Resid. Df Resid. Dev Pr(>Chi)
NULL                           23       2650
Dept         5      160        18       2491   <2e-16 ***
Gender       1      163        17       2328   <2e-16 ***
Admit        1      230        16       2098   <2e-16 ***
Dept:Gender  5     1221        11        877   <2e-16 ***
Dept:Admit   5      855         6         22   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

We proceed to consider what is wrong with these models and how they can be improved. A mosaic display can help diagnose the reason(s) for lack of fit of these models. We focus here on the model $[AD][GD]$ that allows an association between gender and department (i.e., men and women apply at different rates to departments).

[3] Unfortunately, in the historical development of R, the `anova()` methods for linear and generalized linear models provide only *sequential* ("Type I") tests that are computationally easy, but useful only under special circumstances. The car package provides an analogous `Anova()` method that gives more generally useful *partial* ("Type II") tests for the additional contribution of each term beyond the others, taking marginal relations into account.

The `mosaic()` method for "glm" objects in vcdExtra provides a `residuals_type` argument, allowing `residuals_type = "rstandard"` for standardized residuals. The `formula` argument here pertains to the order of the variables in the mosaic, not a model formula.

```
> library(vcdExtra)
> mosaic(berk.glm1, shade = TRUE,
+         formula = ~ Dept + Admit + Gender, split = TRUE,
+         residuals_type = "rstandard",
+         main = "Model: [AdmitDept][GenderDept]",
+         labeling = labeling_residuals,
+         abbreviate_labs = c(Gender = TRUE),
+         keep_aspect_ratio = FALSE)
```

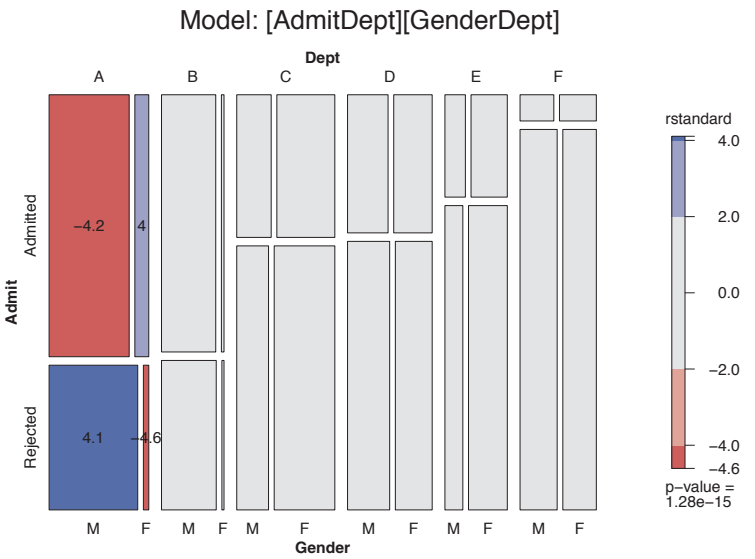


Figure 9.2: Mosaic display for the model [AD][GD], showing standardized residuals for the cell contributions to G^2.

The mosaic display, shown in Figure 9.2, indicates that this model fits well (residuals are small) except in Department A. This suggests a model that allows an association between Admission and Gender in Department A only,

$$\log m_{ijk} = \mu + \lambda_i^A + \lambda_j^D + \lambda_k^G + \lambda_{ij}^{AD} + \lambda_{jk}^{DG} + I(j=1)\lambda_{ik}^{AG} \,, \tag{9.13}$$

where the indicator function $I(j = 1)$ equals 1 for Department A ($j = 1$) and is zero otherwise. This model asserts that Admission and Gender are conditionally independent, given Department, except in Department A. It has one more parameter than the conditional independence model, $[AD][GD]$, and forces perfect fit in the four cells for Department A.

Model Eqn. (9.13) may be fit with `glm()` by constructing a variable equal to the interaction of `gender` and `admit` with a dummy variable having the value 1 for Department A and 0 for other departments.

```
> berkeley <- within(berkeley,
+                    dept1AG <- (Dept == "A") *
+                      (Gender == "Female") *
+                      (Admit == "Admitted"))
> head(berkeley)
```

```
     Admit Gender Dept Freq dept1AG
1 Admitted   Male    A  512       0
2 Rejected   Male    A  313       0
3 Admitted Female    A   89       1
4 Rejected Female    A   19       0
5 Admitted   Male    B  353       0
6 Rejected   Male    B  207       0
```

Fitting this model with the extra term `dept1AG` gives `berk.glm3`

```
> berk.glm3 <- glm(Freq ~ Dept * (Gender + Admit) + dept1AG,
+                  data = berkeley, family = "poisson")
```

This model does indeed fit well, and represents a substantial improvement over model `berk.glm1`:

```
> LRstats(berk.glm3)

Likelihood summary table:
          AIC BIC LR Chisq Df Pr(>Chisq)
berk.glm3 200 222     2.68  5       0.75

> anova(berk.glm1, berk.glm3, test = "Chisq")

Analysis of Deviance Table

Model 1: Freq ~ Dept * (Gender + Admit)
Model 2: Freq ~ Dept * (Gender + Admit) + dept1AG
  Resid. Df Resid. Dev Df Deviance Pr(>Chi)
1         6      21.74
2         5       2.68  1     19.1  1.3e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The parameter estimate for the `dept1AG` term, $\widehat{\lambda}_{ik}^{AG} = 1.052$, may be interpreted as the log odds ratio of admission for females as compared to males in Dept. A. The odds ratio is $\exp(1.052) = 2.86$, the same as the value calculated from the raw data (see Section 4.4.2).

```
> coef(berk.glm3)[["dept1AG"]]

[1] 1.0521

> exp(coef(berk.glm3)[["dept1AG"]])

[1] 2.8636
```

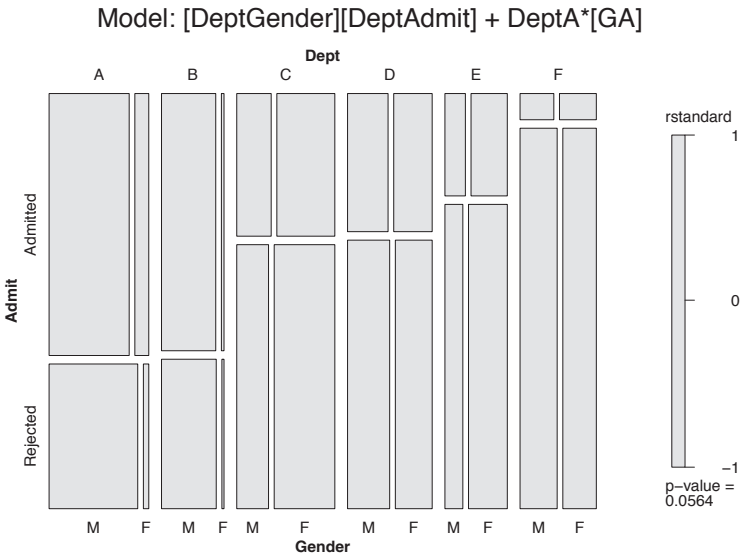


Figure 9.3: Mosaic display for the model `berk.glm3`, allowing an association of gender and admission in Department A. This model now fits the data well.

Finally, Figure 9.3 shows the mosaic for this revised model. The absence of shading indicates a well-fitting model.

```
> mosaic(berk.glm3, shade = TRUE,
+         formula = ~ Dept + Admit + Gender, split = TRUE,
+         residuals_type = "rstandard",
+         main = "Model: [DeptGender][DeptAdmit] + DeptA*[GA]",
+         labeling = labeling_residuals,
+         abbreviate_labs = c(Gender = TRUE),
+         keep_aspect_ratio = FALSE)
```

△

9.4 Equivalent logit models

Because loglinear models are formulated as models for the log (expected) frequency, they make no distinction between response and explanatory variables. In effect, they treat all variables as responses and describe their associations.

Logit (logistic regression) models, on the other hand, describe how the log odds for one variable depends on other, explanatory variables. There is a close connection between the two: When there is a response variable, each logit model for that response is equivalent to a loglinear model.

This relationship often provides a simpler way to formulate and test the model, and to plot and interpret the fitted results. Even when there is no response variable, the logit representation for one variable helps to interpret a loglinear model in terms of odds ratios. The price paid for this simplicity is that associations among the explanatory variables are not expressed in the model.

Consider, for example, the model of homogeneous association, $[AB][AC][BC]$, Eqn. (9.5), for a three-way table, and let variable C be a binary response. Under this model, the logit for variable C is

$$L_{ij} = \log\left(\frac{\pi_{ij|1}}{\pi_{ij|2}}\right) \quad = \quad \log\left(\frac{m_{ij1}}{m_{ij2}}\right)$$

$$= \log(m_{ij1}) - \log(m_{ij2}) .$$

Substituting from Eqn. (9.5), all terms that do not involve variable C cancel, and we are left with

$$\begin{aligned} L_{ij} = \log(m_{ij1}/m_{ij2}) &= (\lambda_1^C - \lambda_2^C) + (\lambda_{i1}^{AC} - \lambda_{i2}^{AC}) + (\lambda_{j1}^{BC} - \lambda_{j2}^{BC}) \\ &= 2\lambda_1^C + 2\lambda_{i1}^{AC} + 2\lambda_{j1}^{BC} , \end{aligned} \tag{9.14}$$

because all λ terms sum to zero. We are interested in how these logits depend on A and B, so we can simplify the notation by replacing the λ parameters with more familiar ones, $\alpha = 2\lambda_1^C$, $\beta_i^A = 2\lambda_{i1}^{AC}$, etc., which express this relation more directly,

$$L_{ij} = \alpha + \beta_i^A + \beta_j^B . \tag{9.15}$$

In the logit model Eqn. (9.15), the response, C, is affected by both A and B, which have additive effects on the log odds of response category C_1 compared to C_2. The terms β_i^A and β_j^B correspond directly to $[AC]$ and $[BC]$ in the loglinear model Eqn. (9.5). The association among the explanatory variables, $[AB]$, is assumed in the logit model, but this model provides no explicit representation of that association. The logit model Eqn. (9.14) is equivalent to the loglinear model $[AB][AC][BC]$ in goodness-of-fit and fitted values, and parameters in the two models correspond directly.

Table 9.1: Equivalent loglinear and logit models for a three-way table, with C as a binary response variable

Loglinear model	Logit model	Logit formula
$[AB][C]$	α	`C ~ 1`
$[AB][AC]$	$\alpha + \beta_i^A$	`C ~ A`
$[AB][BC]$	$\alpha + \beta_j^B$	`C ~ B`
$[AB][AC][BC]$	$\alpha + \beta_i^A + \beta_j^B$	`C ~ A + B`
$[ABC]$	$\alpha + \beta_i^A + \beta_j^B + \beta_{ij}^{AB}$	`C ~ A * B`

Table 9.1 shows the equivalent relationships between all loglinear and logit models for a three-way table when variable C is a binary response. Each model necessarily includes the $[AB]$ association involving the predictor variables. The most basic model, $[AB][C]$, is the intercept-only model, asserting constant odds for variable C. The saturated loglinear model, $[ABC]$, allows an interaction in the effects of A and B on C, meaning that the AC association or odds ratio varies with B.

More generally, when there is a binary response variable, say R, and one or more explanatory variables, $A, B, C, \ldots$, any logit model for R has an equivalent loglinear form. Every term in the logit model, such as β_{ik}^{AC}, corresponds to an association of those factors with R, that is, $[ACR]$ in the equivalent loglinear model.

The equivalent loglinear model must also include all associations among the explanatory factors, the term $[ABC\ldots]$. Conversely, any loglinear model that includes all associations among the explanatory variables has an equivalent logit form. When the response factor has more than two categories, models for generalized logits (Section 8.3) also have an equivalent loglinear form.

EXAMPLE 9.3: Berkeley admissions

The homogeneous association model, $[AD][AG][DG]$, did not fit the `UCBAdmissions` data very well, and we saw that the term $[AG]$ was unnecessary. Nevertheless, it is instructive to consider the equivalent logit model. We illustrate the features of the logit model that lead to the same conclusions and simplified interpretation from graphical displays.

Because Admission is a binary response variable, model Eqn. (9.6) is equivalent to the logit model,

$$L_{ij} = \log\left(\frac{m_{\text{Admit}(ij)}}{m_{\text{Reject}(ij)}}\right) = \alpha + \beta_i^{\text{Dept}} + \beta_j^{\text{Gender}} \,. \tag{9.16}$$

That is, the logit model Eqn. (9.16) asserts that department and gender have additive effects on the log odds of admission. A significance test for the term β_j^{Gender} here is equivalent to the test of the $[AG]$ term for gender bias in the loglinear model. The observed log odds of admission here can be calculated as:

```
> (obs <- log(UCBAdmissions[1,,] / UCBAdmissions[2,,]))

        Dept
Gender        A      B       C      D      E      F
  Male   0.4921 0.5337 -0.5355 -0.704 -0.957 -2.770
  Female 1.5442 0.7538 -0.6604 -0.622 -1.157 -2.581
```

With the data in the form of the frequency data frame `berkeley` we used in Example 9.2, the logit model Eqn. (9.16) can be fit using `glm()` as shown below. In the model formula, the binary response is `Admit == "Admitted"`. The `weights` argument gives the frequency, `Freq` in each table cell.[4]

```
> berk.logit2 <- glm(Admit == "Admitted" ~ Dept + Gender,
+                    data = berkeley, weights = Freq, family = "binomial")
> summary(berk.logit2)


Call:
glm(formula = Admit == "Admitted" ~ Dept + Gender, family = "binomial",
    data = berkeley, weights = Freq)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-25.342  -13.058   -0.163   16.017   21.320

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)    0.5821     0.0690    8.44   <2e-16 ***
DeptB         -0.0434     0.1098   -0.40     0.69
DeptC         -1.2626     0.1066  -11.84   <2e-16 ***
DeptD         -1.2946     0.1058  -12.23   <2e-16 ***
DeptE         -1.7393     0.1261  -13.79   <2e-16 ***
DeptF         -3.3065     0.1700  -19.45   <2e-16 ***
GenderFemale   0.0999     0.0808    1.24     0.22
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 6044.3  on 23  degrees of freedom
Residual deviance: 5187.5  on 17  degrees of freedom
AIC: 5201

Number of Fisher Scoring iterations: 6
```

As in logistic regression models, parameter estimates may be interpreted as increments in the log odds, or $\exp(\beta)$ may be interpreted as the multiple of the odds associated with the explanatory categories. Because `glm()` uses a baseline category parameterization (by default), the coefficients of the first category of `Dept` and `Gender` are set to zero. You can see from the `summary()`

[4] Using weights gives the same fitted values, but not the same LR tests for model fit.

output that the coefficients for the departments decline steadily from A–F.[5] The coefficient $\beta_F^{\text{Gender}} = 0.0999$ for females indicates that, overall, women were $\exp(0.0999) = 1.105$ times as likely as male applicants to be admitted to graduate school at U.C. Berkeley, a 10% advantage.

Similarly, the logit model equivalent of the loglinear model Eqn. (9.13) `berk.glm3` containing the extra 1 df term for an effect of gender in Department A is

$$L_{ij} = \alpha + \beta_i^{\text{Dept}} + I(j = 1)\beta^{\text{Gender}} . \tag{9.17}$$

This model can be fit as follows:

```
> berkeley <- within(berkeley,
+                    dept1AG <- (Dept == "A") * (Gender == "Female"))
> berk.logit3 <- glm(Admit == "Admitted" ~ Dept + Gender + dept1AG,
+                    data = berkeley, weights = Freq, family = "binomial")
```

In contrast to the tests for individual coefficients, the `Anova()` method in the car package gives likelihood-ratio tests of the terms in a model. As mentioned earlier, this provides *partial* ("Type II") tests for the additional contribution of each term beyond all others.

```
> library(car)
> Anova(berk.logit2)

Analysis of Deviance Table (Type II tests)

Response: Admit == "Admitted"
       LR Chisq Df Pr(>Chisq)
Dept      763.4  5     <2e-16 ***
Gender      1.5  1      0.216
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

> Anova(berk.logit3)

Analysis of Deviance Table (Type II tests)

Response: Admit == "Admitted"
        LR Chisq Df Pr(>Chisq)
Dept       646.7  5    < 2e-16 ***
Gender       0.1  1      0.724
dept1AG     17.6  1   2.66e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Plotting logit models

Logit models are easier to interpret than the corresponding loglinear models because there are fewer parameters, and because these parameters pertain to the odds of a response category rather than to cell frequency. Nevertheless, interpretation is often easier still from a graph than from the parameter values.

The simple interpretation of these logit models can be seen by plotting the logits for a given model. To do that, it is necessary to construct a data frame containing the observed (`obs`) and fitted (`fit`) for the combinations of gender and department.

```
> pred2 <- cbind(berkeley[,1:3], fit = predict(berk.logit2))
> pred2 <- cbind(subset(pred2, Admit == "Admitted"), obs = as.vector(obs))
> head(pred2)
```

[5] In fact, the departments were labeled A–F in decreasing order of rate of admission.

```
      Admit Gender Dept      fit      obs
1  Admitted   Male    A  0.58205  0.49212
3  Admitted Female    A  0.68192  1.54420
5  Admitted   Male    B  0.53865  0.53375
7  Admitted Female    B  0.63852  0.75377
9  Admitted   Male    C -0.68055 -0.53552
11 Admitted Female    C -0.58068 -0.66044
```

In this form, these results can be plotted as a line plot of the fitted logits vs. department, with separate curves for males and females, and adding points to show the observed values. Here, we use ggplot2 as shown below, with the `aes()` arguments `group = Gender, color = Gender`. This produces the left panel in Figure 9.4. The same steps for the model `berk.logit3` gives the right panel in this figure. The observed logits, of course, are the same in both plots.

```
> library(ggplot2)
> ggplot(pred2, aes(x = Dept, y = fit, group = Gender, color = Gender)) +
+   geom_line(size = 1.2) +
+   geom_point(aes(x = Dept, y = obs, group = Gender, color = Gender),
+              size = 4) +
+   ylab("Log odds (Admitted)") + theme_bw() +
+   theme(legend.position = c(.8, .9),
+         legend.title = element_text(size = 14),
+         legend.text = element_text(size = 14))
```

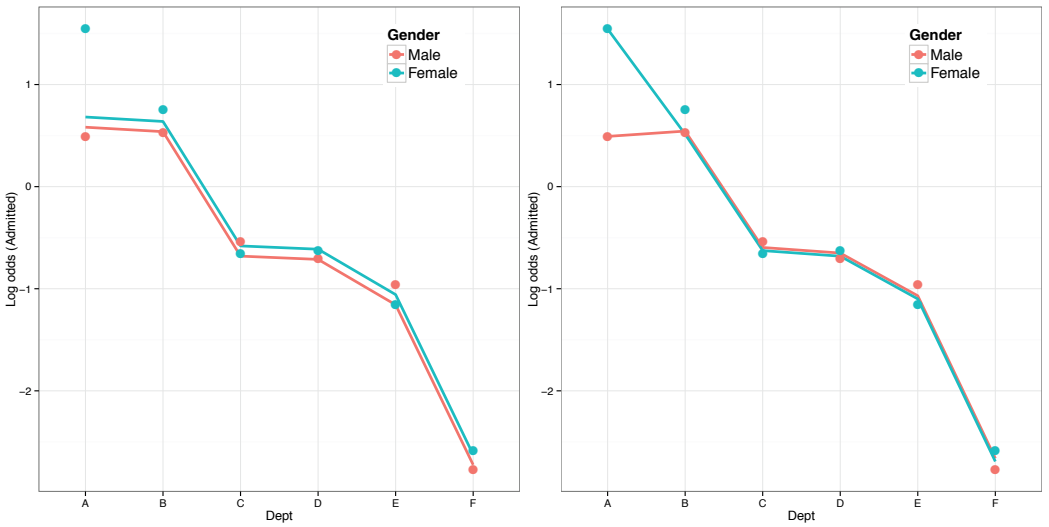


Figure 9.4: Observed (points) and fitted (lines) log odds of admissions in the logit models for the *UCBAdmissions* data. Left: the logit model Eqn. (9.16) corresponding to the loglinear model [AD] [AG] [DG]. Right: the logit model Eqn. (9.17), allowing only a 1 df term for Department A.

The effects seen in our earlier analyses (Examples 5.14, 5.15, and 9.2) may all be observed in these plots. In the left panel of Figure 9.4, corresponding to the loglinear model $[AD][AG][DG]$, the effect of gender, β_j^{Gender}, in the equivalent logit model is shown by the constant separation between the two curves. From the plot we see that this effect is very small (and nonsignificant). In the right panel, corresponding to the logit model Eqn. (9.17), there is no effect of gender on admission, except in department A, where the extra parameter allows perfect fit.

△

9.5 Zero frequencies

Cells with frequencies of zero create problems for loglinear and logit models. For loglinear models, most of the derivations of expected frequencies by maximum likelihood and other quantities that depend on these (e.g., G^2 tests) assume that all $n_{ijk\ldots} > 0$. In analogous logit models, the observed log odds (e.g., for a three-way table), $\log(n_{ij1}/n_{ij2})$, will be undefined if either frequency is zero.

Zero frequencies may occur in contingency tables for two different reasons:

- ***structural zeros*** (also called *fixed zeros*) will occur when it is impossible to observe values for some combinations of the variables. For these cases we should have $\widehat{m}_i = 0$ wherever $n_i = 0$. For example, suppose we have three different methods of contacting people at risk for some obscure genetically inherited disease: newspaper advertisement, telephone campaign, and radio appeal. If each person contacted in any way is classified dichotomously by the three methods of contact, there can never be a non-zero frequency in the 'No-No-No' cell.[6] Similarly, in a tabulation of seniors by gender and health concerns, there can never be males citing menopause or females citing prostate cancer. Square tables, such as wins and losses for sporting teams often have structural zeros in the main diagonal.

- ***sampling zeros*** (also called *random zeros*) occur when the total size of the sample is not large enough in relation to the probabilities in each of the cells to assure that someone will be observed in every cell. Here, it is permissible to have $\widehat{m}_i > 0$ when $n_i = 0$. This problem increases with the number of table variables. For example, in a European survey of religious affiliation, gender, and occupation, we may not happen to observe any female Muslim vineyard-workers in France, although such individuals surely exist in the population. Even when zero frequencies do not occur, tables with many cells relative to the total frequency tend to produce small expected frequencies in at least some cells, which tends to make the G^2 statistics for model fit and likelihood-ratio statistics for individual terms unreliable.

Following Birch (1963b), Haberman (1974) and many others (e.g., Bishop et al., 1975) identified conditions under which the maximum likelihood estimate for a given loglinear model does not exist, meaning that the algorithms used in `loglin()` and `glm()` do not converge to a solution. The problem depends on the number and locations of the zero cells, but not on the size of the frequencies in the remaining cells. Fienberg and Rinaldo (2007) give a historical overview of the problem and current approaches, and Agresti (2013, Section 10.6) gives a compact summary.

In R, the mechanism to handle structural zeros in the IPF approach of `loglin()` and `loglm()` is to supply the argument `start`, giving a table conforming to the data, containing values of 0 in the locations of the zero cells, and non-zero elsewhere.[7] In the `glm()` approach, the argument `subset=Freq > 0` can be used to remove the cells with zero frequencies from the data; alternatively, zero frequencies can be set to `NA`. This usually provides the correct degrees of freedom; however, some estimated coefficients may be infinite.

For a complete table, the residual degrees of freedom are determined as

$$df = \#\text{ of cells} - \#\text{ of fitted parameters}\ .$$

For tables with structural zeros, an analogous general formula is

$$df = (\#\text{ cells} - \#\text{ of parameters}) - (\#\text{ zero cells} - \#\text{ of NA parameters})\ , \tag{9.18}$$

[6] Yet, if we fit an unsaturated model, expected frequencies may be estimated for all cells, and provide a means to estimate the total number at risk in the population. See Lindsey (1995, Section 5.4).

[7] If structural zeros are present, the calculation of degrees of freedom may not be correct. `loglm()` deducts one degree of freedom for each structural zero, but cannot make allowance for patterns of zeros based on the fitted margins that lead to gains in degrees of freedom due to smaller dimension in the parameter space. `loglin()` makes no such correction.

where NA parameters refers to parameters that cannot be estimated due to zero marginal totals in the model formula.

In contrast, sampling zeros are often handled by some modification of the data frequencies to ensure all non-zero cells. Some suggestions are:

- Add a small positive quantity (0.5 is often recommended) to *every* cell in the contingency table (Goodman, 1970), as is often done in calculating empirical log odds (Example 10.10); this simple approach over-smooths the data for unsaturated models, and should be deprecated, although it is widely used in practice.
- Replace sampling zeros by some small number, typically 10^{-10} or smaller (Agresti, 1990).
- Add a small quantity, like 0.1, to *all* zero cells, sampling or structural (Evers and Namboordiri, 1977).

In complex, sparse tables, a sensitivity analysis, comparing different approaches, can help determine if the substantive conclusions vary with the approach to zero cells.

EXAMPLE 9.4: Health concerns of teenagers

Fienberg (1980, Table 8-3) presented a classic example of structural zeros in the analysis of the $4 \times 2 \times 2$ table shown in Table 9.2. The data come from a survey of health concerns among teenagers, originally from Brunswick (1971). Among the health concerns, the two zero entries for menstrual problems among males are clearly structural zeros and therefore one structural zero in the concern-by-gender marginal table. As usual, we abbreviate the table variables concern, age, and gender by their initial letters, C, A, G below.

Table 9.2: Results from a survey of teenagers, regarding their health concerns. *Note*: Two cells with structural zeros are highlighted. *Source:* Fienberg (1980, Table 8-3)

Health	Gender:	Male		Female	
Concerns	Age:	12–15	16–17	12–15	16–17
sex, reproduction		4	2	9	7
menstrual problems		**0**	**0**	4	8
how healthy I am		42	7	19	10
nothing		57	20	71	21

Note: Two cells with structural zeros are highlighted. *Source:* Fienberg (1980, Table 8-3)

The `Health` data is created as a frequency data frame as follows.

```
> Health <- expand.grid(concerns = c("sex", "menstrual",
+                                    "healthy", "nothing"),
+                       age      = c("12-15", "16-17"),
+                       gender   = c("M", "F"))
> Health$Freq <- c(4, 0, 42, 57, 2, 0, 7, 20,
+                  9, 4, 19, 71, 7, 8, 10, 21)
```

In this form, we first use `glm()` to fit two small models, neither of which involves the $\{CG\}$ margin. Model `health.glm0` is the model of mutual independence, $[C][A][G]$. Model `health.glm1` is the model of joint independence, $[C][AG]$, allowing an association between age and gender, but neither with concern. As noted above, the argument `subset = (Freq>0)` is used to eliminate the structural zero cells.

```
> health.glm0 <- glm(Freq ~ concerns + age + gender, data = Health,
+                    subset = (Freq > 0), family = poisson)
> health.glm1 <- glm(Freq ~ concerns + age * gender, data = Health,
+                    subset = (Freq > 0), family = poisson)
```

Neither of these fits the data well. To conserve space, we show only the results of the G^2 tests for model fit.

```
> LRstats(health.glm0, health.glm1)

Likelihood summary table:
              AIC BIC LR Chisq Df Pr(>Chisq)
health.glm0 100.7 105     27.7  8    0.00053 ***
health.glm1  99.9 104     24.9  7    0.00080 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

To see why, Figure 9.5 shows the mosaic display for model `health.glm1`, $[C][AG]$. Note that `mosaic()` takes care to make cells of zero frequency more visible by marking them with a small "0," as these have an area of zero.

```
> mosaic(health.glm1, ~ concerns + age + gender,
+        residuals_type = "rstandard",
+        rot_labels = 0,
+        just_labels = c(left = "right"),
+        margin = c(left = 5))
```

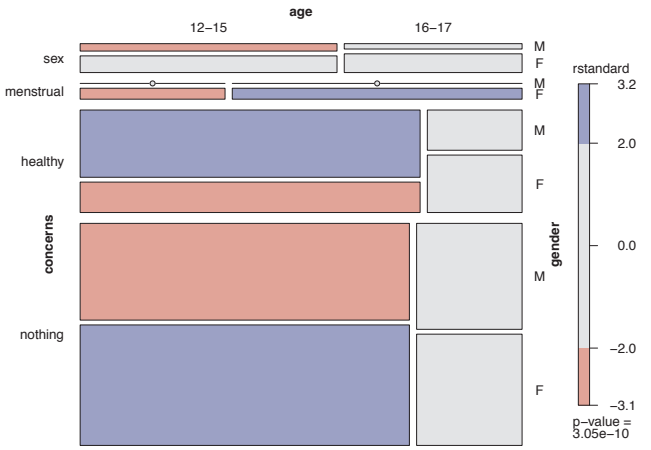


Figure 9.5: Mosaic display for the Health data, model `health.glm1`.

This suggests that there are important associations at least between concern and gender ($[CG]$) and between concern and age ($[CA]$). These are incorporated into the next model:

```
> health.glm2 <- glm(Freq ~ concerns*gender + concerns*age, data = Health,
+                    subset = (Freq > 0), family = poisson)
> LRstats(health.glm2)

Likelihood summary table:
             AIC  BIC LR Chisq Df Pr(>Chisq)
health.glm2 87.7 94.7     4.66  3        0.2
```

The degrees of freedom are correct here. Eqn. (9.18), with 2 zero cells and 1 NA parameter due to the zero in the $\{CG\}$ margin, gives $df = (16 - 12) - (2 - 1) = 3$. The loss of one estimable parameter can be seen in the output from `summary`.

```
> summary(health.glm2)


Call:
glm(formula = Freq ~ concerns * gender + concerns * age, family = poisson,
    data = Health, subset = (Freq > 0))

Deviance Residuals:
     1       3       4       5       7       8       9      10      11      12
 0.236   0.585  -0.173  -0.300  -1.202   0.302  -0.149   0.000  -0.795   0.158
    13      14      15      16
 0.176   0.000   1.348  -0.282

Coefficients: (1 not defined because of singularities)
                          Estimate Std. Error z value Pr(>|z|)
(Intercept)                  1.266      0.445    2.84   0.0045 **
concernsmenstrual           -0.860      0.586   -1.47   0.1425
concernshealthy              2.380      0.471    5.05  4.4e-07 ***
concernsnothing              2.800      0.462    6.07  1.3e-09 ***
genderF                      0.981      0.479    2.05   0.0405 *
age16-17                    -0.368      0.434   -0.85   0.3964
concernsmenstrual:genderF       NA         NA      NA       NA
concernshealthy:genderF     -1.505      0.533   -2.82   0.0047 **
concernsnothing:genderF     -0.803      0.503   -1.60   0.1105
concernsmenstrual:age16-17   1.061      0.750    1.41   0.1574
concernshealthy:age16-17    -0.910      0.513   -1.77   0.0761 .
concernsnothing:age16-17    -0.771      0.469   -1.64   0.1005
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 252.4670  on 13  degrees of freedom
Residual deviance:   4.6611  on  3  degrees of freedom
AIC: 87.66

Number of Fisher Scoring iterations: 4
```

In contrast, `loglm()` reports the degrees of freedom incorrectly for models containing zeros in any fitted margin. For use with `loglm()`, we convert it to a $4 \times 2 \times$ table.

```
> health.tab <- xtabs(Freq ~ concerns + age + gender, data = Health)
```

The same three models are fitted with `loglm()` as shown below. The locations of the positive frequencies are marked in the array `nonzeros` and supplied as the value of the `start` argument.

```
> nonzeros <- ifelse(health.tab>0, 1, 0)
> health.loglm0 <- loglm(~ concerns + age + gender,
+                        data = health.tab, start = nonzeros)
> health.loglm1 <- loglm(~ concerns + age * gender,
+                        data = health.tab, start = nonzeros)
> # df is wrong
> health.loglm2 <- loglm(~ concerns*gender + concerns*age,
+                        data = health.tab, start = nonzeros)
> LRstats(health.loglm0, health.loglm1, health.loglm2)

Likelihood summary table:
               AIC BIC LR Chisq Df Pr(>Chisq)
health.loglm0 104.7 111    27.74  8    0.00053 ***
health.loglm1 103.9 111    24.89  7    0.00080 ***
health.loglm2  93.7 104     4.66  2    0.09724 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The results agree with those of `glm()`, except for the degrees of freedom for the last model.

△

9.6 Chapter summary

- Loglinear models provide a comprehensive scheme to describe and understand the associations among two or more categorical variables. It is helpful to think of these as discrete analogs of ANOVA models, or of regression models, where the log of cell frequency is modelled as a linear function of predictors.
- Loglinear models typically make no distinction between response and explanatory variables. When one variable *is* a response, however, any logit model for that response has an equivalent loglinear model. The logit form is usually simpler to formulate and test, and plots of the observed and fitted logits are easier to interpret.
- In all these cases, the interplay between graphing and fitting is important in arriving at an understanding of the relationships among variables, and an adequate descriptive model that is faithful to the details of the data.
- Cells with zero frequencies create problems for estimation and testing hypotheses in loglinear models. Different methods are available to handle *structural zeros* and *sampling zeros*.

9.7 Lab exercises

Exercise 9.1 Consider the data set `DaytonSurvey` (described in Example 2.6), giving results of a survey of use of alcohol (A), cigarettes (C), and marijuana (M) among high school seniors. For this exercise, ignore the variables `sex` and `race`, by working with the marginal table `Dayton.ACM`, a $2 \times 2 \times 2$ table in frequency data frame form.

```
> Dayton.ACM <- aggregate(Freq ~ cigarette + alcohol + marijuana,
+                         data=DaytonSurvey, FUN=sum)
```

(a) Use `loglm()` to fit the model of mutual independence, [A][C][M].
(b) Prepare mosaic display(s) for associations among these variables. Give a verbal description of the association between cigarette and alcohol use.
(c) Use `fourfold()` to produce fourfold plots for each pair of variables, AC, AM, and CM, stratified by the remaining one. Describe these associations verbally.

Exercise 9.2 Continue the analysis of the `DaytonSurvey` data by fitting the following models:

(a) Joint independence, [AC][M]
(b) Conditional independence, [AM][CM]
(c) Homogeneous association, [AC][AM][CM]
(d) Prepare a table giving the goodness-of-fit tests for these models, as well as the model of mutual independence, [A][C][M], and the saturated model, [ACM]. *Hint*: `anova()` and `LRstats()` are useful here. Which model appears to give the most reasonable fit?

Exercise 9.3 The data set `Caesar` in vcdExtra gives a 3×2^3 frequency table classifying 251 women who gave birth by Caesarian section by `Infection` (three levels: none, Type 1, Type2) and `Risk`, whether `Antibiotics` were used, and whether the Caesarian section was `Planned` or not. `Infection` is a natural response variable, but the table has quite a few zeros.

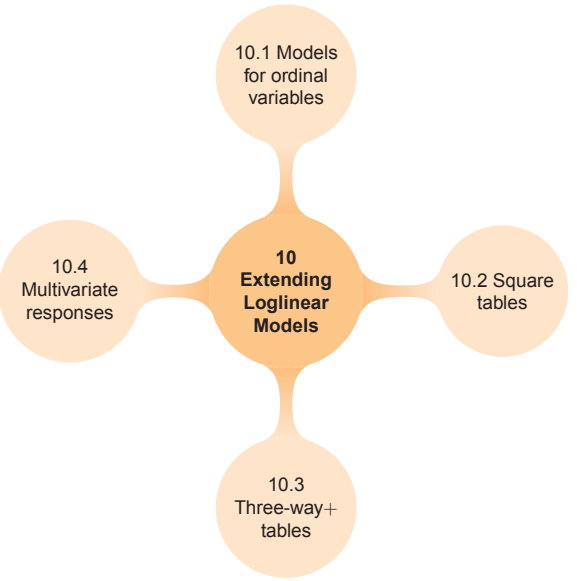
(a) Use `structable()` and `mosaic()` to see the locations of the zero cells in this table.
(b) Use `loglm()` to fit the baseline model [I][RAP]. Is there any problem due to zero cells indicated in the output?

(c) For the purpose of this excercise, treat all the zero cells as *sampling zeros* by adding 0.5 to all cells, e.g., `Caesar1 <- Caesar + 0.5`. Refit the baseline model.
(d) Now fit a "main effects" model [IR][IA][IP][RAP] that allows associations of `Infection` with each of the predictors.

Exercise 9.4 The *`Detergent`* in vcdExtra gives a $2^3 \times 3$ table classifying a sample of 1,008 consumers according to their preference for (a) expressed `Preference` for Brand "X" or Brand "M" in a blind trial, (b) `Temperature` of laundry water used, (c) previous use (`M_user`) of detergent Brand "M," and (d) the softness (`Water_softness`) of the laundry water used.

(a) Make some mosaic displays to visualize the associations among the table variables. Try using different orderings of the table variables to make associations related to `Preference` more apparent.
(b) Use a `doubledecker()` plot to visualize how `Preference` relates to the other factors.
(c) Use `loglm()` to fit the baseline model [P][TMW] for `Preference` as the response variable. Use a mosaic display to visualize the lack of fit for this model.

This page intentionally left blank



10

Extending Loglinear Models

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

> The universe is built on a plan the profound symmetry of which is somehow present in the inner structure of our intellect.
>
> Paul Valery, 1871–1945

This chapter extends the analysis of loglinear models to some important special cases allowing us to represent additional structure in the variables in contingency tables in a way that provides a more parsimonious description of associations than is available from models for general association. One class of such simplified models (Section 10.1) occurs when one or more of the explanatory variables are ordinal, and discrete levels might be replaced by numerical values.

Models for square tables (Section 10.2) with the same row and column categories comprise another special case giving simpler descriptions than the saturated model of general association. These important special cases are extended to three-way and higher-dimensional tables in Section 10.3.

Finally, Section 10.4 describes some methods for dealing with situations where there are several response variables, and it is useful to understand both the marginal relations of the responses with the predictors as well as how their association varies with the predictors.

10.1 Models for ordinal variables

Standard loglinear models treat all classification variables as nominal, unordered factors. In these models, all statistical tests are identical and parameter estimates are equivalent if the categories of any of the table variable are reordered. Yet we have seen that the ordering of categories often provides important information about the nature of associations, and we showed (Section 4.2.4) that non-parametric tests which take into account the ordered nature of a factor are more powerful.

Correspondence analysis plots (Chapter 6) make it easy to see the relationships between ordinal variables, because the method assigns quantitative scores to the table variables that maximally account for their association. As we saw for the hair–eye color data (Figure 6.1) and the mental impairment data (Figure 6.2), an association can be interpreted in terms of ordered categories when the points for two factors are ordered similarly, usually along the first CA dimension.

Similarly, in a mosaic display, an ordered associative effect is seen when the residuals have an opposite-corner pattern of positive and negative signs and magnitudes (e.g., for the hair–eye color data, Figure 5.4). In these cases loglinear and logit models that use the ordered nature of the factors offer several advantages:

- Because they are more focused, tests that use the ordinal structure of the table variables are more powerful when the association varies systematically with the ordered values of a factor.
- Because they consume fewer degrees of freedom, we can fit unsaturated models where the corresponding model for nominal factors would be saturated. In a two-way table, for example, a variety of models for ordinal factors may be proposed that are intermediate between the independence model and the saturated model.
- Parameter estimates from these models are fewer in number, are easier to interpret, and quantify the nature of effects better than corresponding quantities in models for nominal factors. Estimating fewer parameters typically gives smaller standard errors.

These advantages are analogous to the use of tests for trends or polynomial contrasts in ANOVA models. More importantly, in some research areas in the social sciences (where categorical data is commonplace), models for ordinal variables have proved crucial in theory construction and debates, giving more precise tests of hypotheses than are available from less focused or descriptive methods (Agresti, 1984).

10.1.1 Loglinear models for ordinal variables

For a two-way table, when either the row variable or the column variable, or both, are ordinal, one simplification comes from assigning ordered scores, $\boldsymbol{a} = (a_i), a_1 \leq a_2 \leq \cdots a_I$, and/or $\boldsymbol{b} = (b_j), b_1 \leq b_2 \leq \cdots b_J$, to the categories so that the ordinal relations are necessarily included in the model. Typically, equally spaced scores are used; for example, integer scores, $a_i = i$, or the zero-sum equivalent, $a_i = i - (I+1)/2$ (e.g., $(a_i) = (-1, 0, 1)$ for $I = 3$).

Using such scores gives simple interpretations of the association parameters in terms of *local odds ratios* for adjacent 2×2 subtables,

$$\theta_{ij} = \frac{m_{ij}\, m_{i+1,j+1}}{m_{i,j+1}\, m_{i+1,j}} \,, \tag{10.1}$$

which is the odds ratio for pairs of adjacent rows and adjacent columns.

When both variables are assigned scores, this gives the ***linear-by-linear model*** $(L \times L)$

$$\log(m_{ij}) = \mu + \lambda_i^A + \lambda_j^B + \gamma\, a_i b_j \,. \tag{10.2}$$

Because the scores $\boldsymbol{a}$ and $\boldsymbol{b}$ are fixed, this model has only one extra parameter, γ, compared to the independence model, which is the special case, $\gamma = 0$. In contrast, the saturated model, allowing general association λ_{ij}^{AB}, uses $(I-1)(J-1)$ additional parameters.

The terms $\gamma a_i b_j$ in Eqn. (10.2) describe a pattern of association where deviations from independence increase linearly with a_i and b_j in opposite directions towards the opposite corners of the table, as we have often observed in mosaic displays.

In the linear-by-linear association model, the local log odds ratios are

$$\log(\theta_{ij}) = \gamma(a_{i+1} - a_i)(b_{j+1} - b_j) \; ,$$

which reduces to

$$\log(\theta_{ij}) = \gamma$$

for integer-spaced scores, so γ is the common local log odds ratio. As a result, the linear-by-linear model is sometimes called the ***uniform association model*** (Goodman, 1979).

Generalizations of the linear-by-linear model result when only one variable is assigned scores. In the ***row effects model*** (R), the row variable, A, is treated as nominal, while the column variable, B, is assigned ordered scores (b_j). The loglinear model is then

$$\log(m_{ij}) = \mu + \lambda_i^A + \lambda_j^B + \alpha_i b_j \; , \tag{10.3}$$

where the α_i parameters are the *row effects*. An additional constraint, $\sum_i \alpha_i = 0$ or $\alpha_1 = 0$, is imposed, so that model Eqn. (10.3) has only $(I-1)$ more parameters than the independence model. The linear-by-linear model is the special case where the row effects are equally spaced, and the independence model is the special case where all $\alpha_i = 0$.

The row-effects model Eqn. (10.3) also has a simple odds ratio interpretation. The local log odds ratio for adjacent pairs of rows and columns is

$$\log(\theta_{ij}) = \alpha_{i+1} - \alpha_i \; ,$$

which is constant for all pairs of adjacent columns. Plots of the local log odds ratio against i would appear as a set of coincident curves.

In the analogous ***column effects model*** (C), $(J-1)$ linearly independent column effect parameters β_j are estimated for the column variable, while fixed scores $\{a_i\}$ are assigned to the row variable. It is also possible to fit a ***row plus column effects model*** (R+C), which assigns specified scores to both the rows and column variables. Plots of the local odds ratios for the R+C model appear as parallel curves.

Nesting relationships among these models and others (RC(1) and RC(2)) described in Section 10.1.3 are shown in Figure 10.1. Any set of models connected by a path can be directly compared with likelihood-ratio tests of the form $G^2(M_2|M_1)$.

In R, the $L \times L$, row effects, and column effects models can all be fit using `glm()` simply by replacing the appropriate table factor variable(s) with their `as.numeric()` equivalents.

EXAMPLE 10.1: Mental impairment and parents' SES

The `Mental` data on the mental health status of young New York residents in relation to their parents' socioeconomic status was examined in Example 4.7 using CMH tests for ordinal association and in Example 6.2 using correspondence analysis. Figure 6.2 showed that nearly all of the association in the table was accounted for by a single dimension along which both factors were ordered, consistent with the view that mental health increased in relation to parents' SES.

Because these models provide their interpretations in terms of local odds ratios, Eqn. (10.1), it is helpful to see these values for the observed data, corresponding to the saturated model. The values $\log(\theta_{ij})$ are calculated by `loddsratio()` in vcd, with the data in table form.

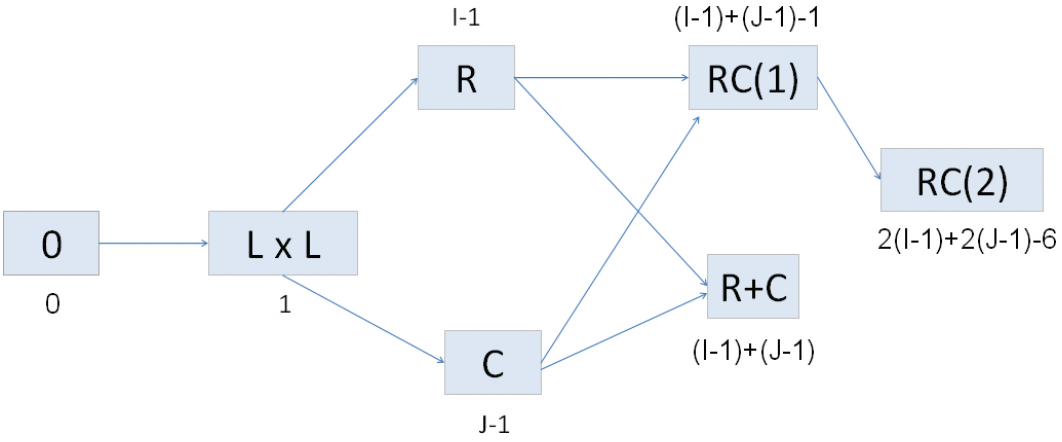


Figure 10.1: Nesting relationships among some association models for an $I \times J$ table specifying the association parameters, λij^{AB}. Model **0** is the independence model. Formulas near the boxes give the number of identifiable association parameters. Arrows point from one nested model to another that is a more general version.

```
> library(vcd)
> data("Mental", package = "vcdExtra")
> (mental.tab <- xtabs(Freq ~ mental + ses, data=Mental))

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

> (LMT <- loddsratio(mental.tab))

log odds ratios for mental and ses

                   ses
mental                 1:2    2:3     3:4    4:5    5:6
  Well:Mild          0.1158 0.1107  0.0612 0.3191  0.227
  Mild:Moderate     -0.0715 0.0747 -0.1254 0.0192  0.312
  Moderate:Impaired -0.0683 0.2201  0.2795 0.1682 -0.094
```

A simple plot of these values, using area- and color-proportional shaded squares, is shown in Figure 10.2. This plot is drawn using the corrplot (Wei, 2013) package. It is easy to see that most of the local odds ratios are mildly positive.[1]

```
> library(corrplot)
> corrplot(as.matrix(LMT), method = "square", is.corr = FALSE,
+          tl.col = "black", tl.srt = 0, tl.offset = 1)
```

For comparison with the $L \times L$ model fitted below, the mean local log odds ratio is 0.103.

[1] Using `plot(loddsratio(mental.tab))` would give a line plot of the odds ratios as illustrated in Section 5.9.2 (e.g., Figure 5.37).

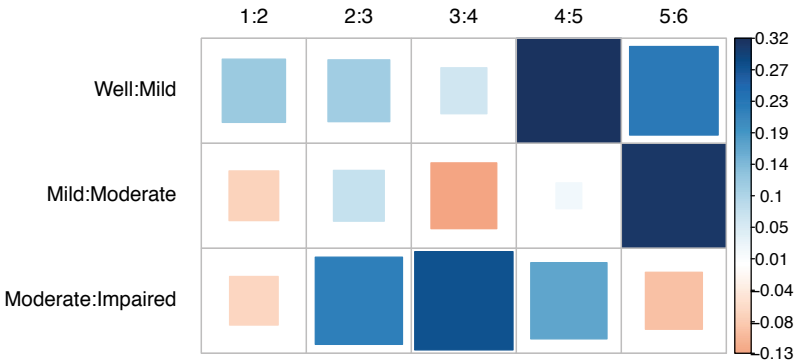


Figure 10.2: Shaded-square plot of the local log odds ratios in the *Mental* data.

```
> mean(LMT$coefficients)

[1] 0.10323
```

As a baseline, we first fit the independence model (testing $H_0 : \log(\theta_{ij}) = 0$) with `glm()`. As expected, this model fits quite badly, with $G^2\ (15) = 47.418$.

```
> indep <- glm(Freq ~ mental + ses, data = Mental, family = poisson)
> LRstats(indep)

Likelihood summary table:
      AIC BIC LR Chisq Df Pr(>Chisq)
indep 210 220     47.4 15    3.2e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The mosaic display of standardized residuals from this model is shown in Figure 10.3. The argument `labeling=labeling_residuals` is used to show the numerical values in the cells with absolute values greater than `suppress=1`.

```
> long.labels <- list(set_varnames = c(mental="Mental Health Status",
+                                      ses="Parent SES"))
> mosaic(indep,
+        gp=shading_Friendly,
+        residuals_type="rstandard",
+        labeling_args = long.labels,
+        labeling=labeling_residuals, suppress=1,
+        main="Mental health data: Independence")
```

This figure shows the classic opposite-corner pattern of the signs and magnitudes of the residuals that would arise if the association between mental health and SES could be explained by the ordinal relation of these factors using one of the $L \times L$, R, or C models.

To fit such ordinal models, you can use `as.numeric()` on a factor variable to assign integer scores, or assign other values if integer spacing is not appropriate.

```
> Cscore <- as.numeric(Mental$ses)
> Rscore <- as.numeric(Mental$mental)
```

Then, the $L \times L$, R, C, and $R + C$ models can be fit as follows, using `update()`, where beyond the main effects of `mental` and `ses`, their association is represented as the interaction of the numeric score(s) or factor(s), as appropriate in each case.

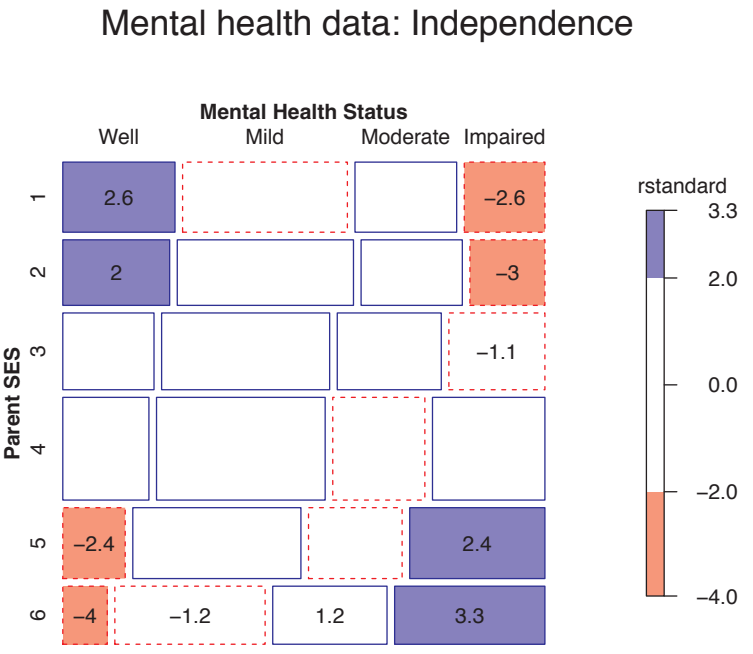


Figure 10.3: Mosaic display of the independence model for the mental health data.

```
> linlin <- update(indep, . ~ . + Rscore:Cscore)
> roweff <- update(indep, . ~ . + mental:Cscore)
> coleff <- update(indep, . ~ . + Rscore:ses)
> rowcol <- update(indep, . ~ . + Rscore:ses + mental:Cscore)
```

Goodness-of-fit tests for these models are shown below. They show that all of the $L \times L$, R, and C models are acceptable in terms of the likelihood-ratio G^2. The $L \times L$ model, with only one more parameter than the independence model, is judged the best by both AIC and BIC.

```
> LRstats(indep, linlin, roweff, coleff, rowcol)

Likelihood summary table:
         AIC   BIC LR Chisq Df Pr(>Chisq)
indep  209.6 220.2    47.42 15   3.16e-05 ***
linlin 174.1 185.8     9.90 14      0.770
roweff 174.4 188.6     6.28 12      0.901
coleff 179.0 195.5     6.83 10      0.741
rowcol 179.2 198.1     3.05  8      0.931
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

In cases where such overall tests are unclear, you can carry out tests of nested sets of models using `anova()`, giving tests of ΔG^2.

```
> anova(indep, linlin, roweff, test = "Chisq")

Analysis of Deviance Table

Model 1: Freq ~ mental + ses
Model 2: Freq ~ mental + ses + Rscore:Cscore
Model 3: Freq ~ mental + ses + mental:Cscore
  Resid. Df Resid. Dev Df Deviance Pr(>Chi)
```

```
1         15       47.4
2         14        9.9  1     37.5    9e-10 ***
3         12        6.3  2      3.6     0.16
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

> anova(indep, linlin, coleff, test = "Chisq")

Analysis of Deviance Table

Model 1: Freq ~ mental + ses
Model 2: Freq ~ mental + ses + Rscore:Cscore
Model 3: Freq ~ mental + ses + ses:Rscore
  Resid. Df Resid. Dev Df Deviance Pr(>Chi)
1         15       47.4
2         14        9.9  1     37.5    9e-10 ***
3         10        6.8  4      3.1     0.55
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Under the $L \times L$ model, the estimate of the coefficient of `Rscore:Cscore` is $\widehat{\gamma} = 0.0907$ (s.e.=0.015) with unit-spaced scores, as shown below.

```
> # interpret linlin association parameter
> coef(linlin)[["Rscore:Cscore"]]

[1] 0.090687

> exp(coef(linlin)[["Rscore:Cscore"]])

[1] 1.0949
```

This corresponds to a local odds ratio, $\hat{\theta}_{ij} = \exp(0.0907) = 1.095$. This single number describes the association succinctly: each step down the socioeconomic scale increases the odds of being classified one step poorer in mental health by 9.5%.

△

10.1.2 Visualizing model structure

In Section 5.8 we illustrated how to use mosaic displays to visualize the *structure* of loglinear models. The basic idea was just to use mosaic plots or mosaic matrices to show the *fitted* values implied by a given model. As just described, loglinear models for ordinal variables have very simple structures in terms of log odds ratios, and you can similarly understand their structure by calculating or plotting the local odds ratios from the fitted frequencies for a given model.

EXAMPLE 10.2: Mental impairment and parents' SES

We illustrate this idea numerically here, for the row effects (R) model, `roweff`, fit to the *Mental* data. `fitted()` gets the fitted frequencies for this model, and using `loddsratio()` on the result show that these are constant in each row.

```
> roweff.fit <- matrix(fitted(roweff), 4, 6,
+                      dimnames=dimnames(mental.tab))
> round(as.matrix(loddsratio(roweff.fit)), 3)

                    ses
mental                 1:2   2:3   3:4   4:5   5:6
  Well:Mild          0.145 0.145 0.145 0.145 0.145
  Mild:Moderate      0.018 0.018 0.018 0.018 0.018
  Moderate:Impaired 0.143 0.143 0.143 0.143 0.143
```

Similarly, the column effects (C) model, `coleff`, shows these values to be constant in each column.

```
> coleff.fit <- matrix(fitted(coleff), 4, 6,
+                      dimnames = dimnames(mental.tab))
> round(as.matrix(loddsratio(coleff.fit)), 3)

                    ses
mental                 1:2   2:3   3:4   4:5   5:6
  Well:Mild         -0.013 0.125 0.053 0.142 0.139
  Mild:Moderate     -0.013 0.125 0.053 0.142 0.139
  Moderate:Impaired -0.013 0.125 0.053 0.142 0.139
```

Using `plot(loddsratio(model))` for these cases and the R+C model gives the plots in Figure 10.4.

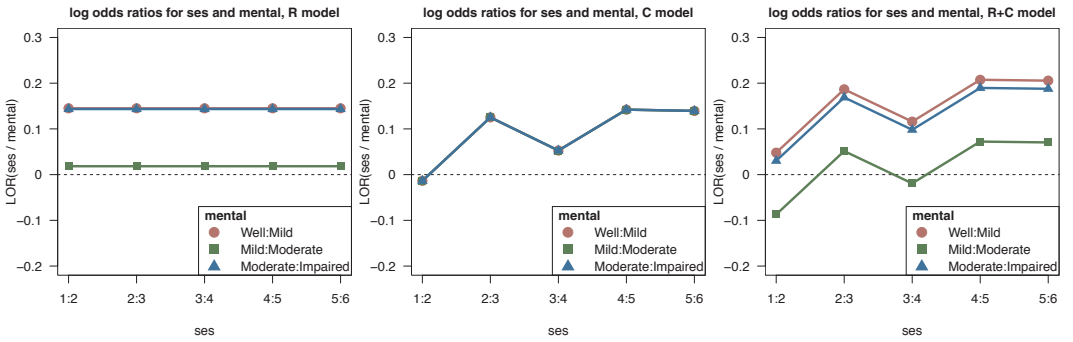


Figure 10.4: Log odds ratio plots for the R (left), C (middle), and R+C (right) models fit to the mental health data.

In contrast, the (more parsimonious) $L \times L$ model has a constant log odds ratio, $\widehat{\gamma} = 0.0907$. $\triangle$

10.1.3 Log-multiplicative (RC) models

The association models described above are all more parsimonious and easier to interpret than the saturated model. However, they depend on assigning fixed and possibly arbitrary scores to the variable categories. A generalization of the $L \times L$ model that treats *both* row and column scores as parameters is the ***row-and-column effects model*** (RC(1)) suggested by Goodman (1979),

$$\log(m_{ij}) = \mu + \lambda_i^A + \lambda_j^B + \gamma\, \alpha_i \beta_j \,, \tag{10.4}$$

where γ, $\boldsymbol{\alpha}$, and $\boldsymbol{\beta}$ comprise additional parameters to be estimated beyond the independence model.[2] This model has a close connection with correspondence analysis (Goodman, 1985), where the estimated scores $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are analogous to correspondence analysis scores on a first dimension.[3] γ, called the *intrinsic association coefficient*, is analogous to the same parameter in the $L \times L$ model.

For identifiability and interpretation it is necessary to impose some normalization constraints on the $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$. An *unweighted, unit standardized* solution forces $\sum_i \alpha_i = \sum_j \beta_j = 0$ and

[2] In contrast to the R, C, and R+C models, RC models do not assume that the categories are appropriately ordered because the category scores are estimated from the data.

[3] However, when estimated by maximum likelihood, the RC(1) model allows likelihood-ratio tests of parameters and model fit, AIC and BIC statistics, and methods for estimating standard errors of the parameters. Such model-based methods are not available for correspondence analysis.

$\sum_i \alpha_i^2 = \sum_j \beta_j^2 = 1$. Alternatively, and more akin to correspondence analysis solutions, the *marginally weighted* solution uses the marginal probabilities π_{i+} of the row variable and π_{+j} of the columns as weights:

$$\begin{aligned} \sum_i \alpha_i \pi_{i+} &= \sum_j \beta_j \pi_{+j} = 0 \,, \\ \sum_i \alpha_i^2 \pi_{i+} &= \sum_j \beta_j^2 \pi_{+j} = 1 \,. \end{aligned} \tag{10.5}$$

Goodman (1986) generalized this to multiple bilinear terms of the form $\gamma_k \; \alpha_{ik}\beta_{jk}$, with M terms (the RC(M) model) and showed that *all* associations in the saturated model could be expressed exactly as

$$\lambda_{ij}^{AB} = \sum_{k=1}^{M} \gamma_k \; \alpha_{ik}\beta_{jk} \qquad M = \min\left(I-1,\, J-1\right) . \tag{10.6}$$

In practice, models with fewer terms usually suffice. For example, an RC(2) model with two multiplicative terms is analogous to a two-dimensional correspondence analysis solution. In addition to the normalization constraints for the RC(1) model, parameters in an RC(M) model must satisfy the additional constraints that the (possibly weighted) scores for distinct dimensions are orthogonal (uncorrelated), similar to correspondence analysis solutions.

The RC model is *not* a loglinear model because it contains a multiplicative term in the parameters. This model and a wide variety of other nonlinear models for categorical data can be fit using `gnm()` in the gnm package. This provides the basic machinery for extending `glm()` models to nonlinear terms, quite generally. The function `rc()` in the logmult package uses `gnm()` for fitting, and offers greater convenience in normalizing the category scores, calculating standard errors, and plotting.

EXAMPLE 10.3: Mental impairment and parents' SES

The gnm package provides a number of functions that can be used in model formulas for nonlinear association terms. Among these, `Mult()` expresses a multiplicative association in terms of two (or more) factors. The RC(1) model for factors `A, B` uses `Mult(A,B)` for the association term in Eqn. (10.4). Multiple multiplicative RC terms, as in Eqn. (10.6), can be expressed using `instances(Mult(A,B), m)`.

To illustrate, we fit the RC(1) and RC(2) models to the *Mental* data using `gnm()`. In this table, both factors are ordered, but we don't want to use the default polynomial contrasts, so we set their contrast attributes to `treatment`.

```
> library(gnm)
> contrasts(Mental$mental) <- contr.treatment
> contrasts(Mental$ses) <- contr.treatment
> indep <- gnm(Freq ~ mental + ses, data = Mental, family = poisson)
> RC1 <- update(indep, . ~ . + Mult(mental, ses), verbose = FALSE)
> RC2 <- update(indep, . ~ . + instances(Mult(mental, ses), 2),
+               verbose = FALSE)
```

For comparison with the loglinear association models fit in Example 10.1 we show the G^2 goodness-of-fit tests for all these models. The ordinal loglinear models and the RC models all fit well, with the $L \times L$ model preferred on the basis of parsimony by AIC and BIC.

```
> LRstats(indep, linlin, roweff, coleff, RC1, RC2)

Likelihood summary table:
         AIC   BIC LR Chisq Df Pr(>Chisq)
indep  209.6 220.2    47.42 15   3.16e-05 ***
```

```
linlin 174.1 185.8     9.90 14      0.770
roweff 174.4 188.6     6.28 12      0.901
coleff 179.0 195.5     6.83 10      0.741
RC1    179.7 198.6     3.57  8      0.894
RC2    186.7 211.4     0.52  3      0.914
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The substantive difference between the $L \times L$ model and the RC(1) model is whether the categories of mental health status and SES can be interpreted as equally spaced along some latent continua, versus the alternative that category spacing is unequal. We can test this directly using the likelihood-ratio test, $G^2(L \times L \,|\, RC(1))$. Similarly, model `RC1` is nested within model `RC2`, so $G^2(RC(1) \,|\, RC(2))$ gives a direct test of the need for a second dimension.

```
> anova(linlin, RC1, RC2, test = "Chisq")

Analysis of Deviance Table

Model 1: Freq ~ mental + ses + Rscore:Cscore
Model 2: Freq ~ mental + ses + Mult(mental, ses)
Model 3: Freq ~ mental + ses + Mult(mental, ses, inst = 1) + Mult(mental,
    ses, inst = 2)
  Resid. Df Resid. Dev Df Deviance Pr(>Chi)
1        14       9.90
2         8       3.57  6     6.32     0.39
3         3       0.52  5     3.05     0.69
```

We see that estimated scores for the categories in the model `RC1` do not provide a significantly better fit, and there is even less evidence for a second dimension of category parameters in the `RC2` model.

Nevertheless, for cases where RC models *do* provide some advantage, it is useful to know how to visualize the estimated category parameters. The key to this is the function `getContrasts()`, which computes contrasts or scaled contrasts for a set of (non-eliminated) parameters from a "gnm" model, together with standard errors for the estimated contrasts following the methods of Firth (2003) and also Firth and Menezes (2004). The details are explained in `help(getContrasts)` and in `vignette("gnmOverview")`, which comes with the gnm package.

The coefficients in the marginally weighted solution Eqn. (10.5) can be obtained as follows.

```
> rowProbs <- with(Mental, tapply(Freq, mental, sum) / sum(Freq))
> colProbs <- with(Mental, tapply(Freq, ses, sum) / sum(Freq))
> mu <- getContrasts(RC1, pickCoef(RC1, "[.]mental"),
+                    ref = rowProbs, scaleWeights = rowProbs)
> nu <- getContrasts(RC1, pickCoef(RC1, "[.]ses"),
+                    ref = colProbs, scaleWeights = colProbs)
```

In our notation, the coefficients α and β can be extracted as the `qvframe` component of the "qv" object returned by `getContrasts()`.

```
> (alpha <- mu$qvframe)

                             Estimate Std. Error
Mult(., ses).mentalWell       1.67378    0.19043
Mult(., ses).mentalMild       0.14009    0.20018
Mult(., ses).mentalModerate -0.13669    0.27948
Mult(., ses).mentalImpaired -1.41055    0.17418

> (beta  <- nu$qvframe)
```

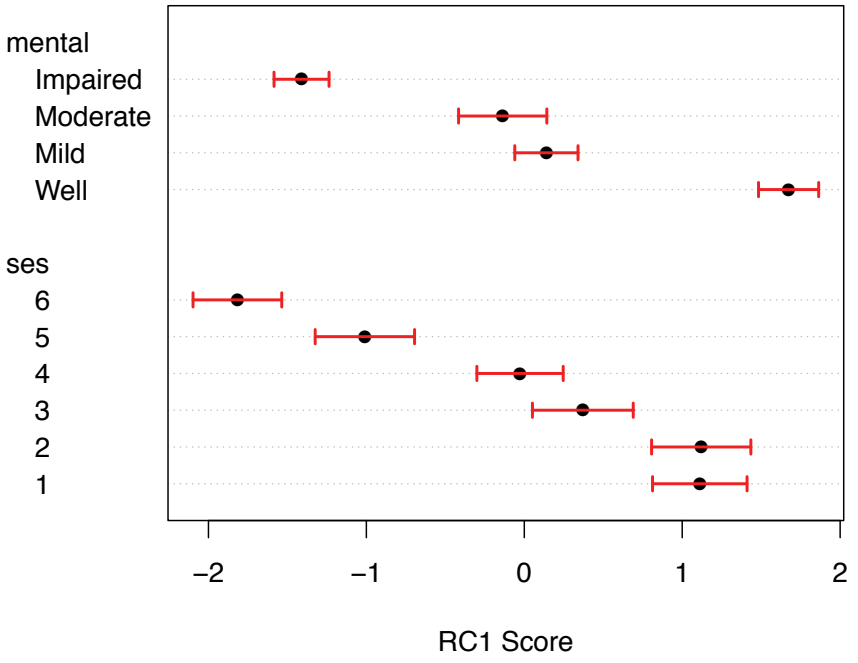


Figure 10.5: Dotchart of the scaled category scores for the RC(1) model fit to the mental health data. Error bars show ±1 standard error.

```
                     Estimate Std. Error
Mult(mental, .).ses1  1.111360    0.29921
Mult(mental, .).ses2  1.120459    0.31422
Mult(mental, .).ses3  0.370752    0.31915
Mult(mental, .).ses4 -0.027006    0.27328
Mult(mental, .).ses5 -1.009480    0.31470
Mult(mental, .).ses6 -1.816647    0.28095
```

For plotting this RC(1) solution for the scaled category scores together with their estimated standard errors, a `dotchart()`, shown in Figure 10.5, provides a reasonable visualization.

To create this plot, first combine the row and column scores in a data frame, and add columns `lower, upper` corresponding to ±1 standard error (or some other multiple).

```
> scores <- rbind(alpha, beta)
> scores <- cbind(scores,
+                 factor = c(rep("mental", 4), rep("ses", 6)) )
> rownames(scores) <- c(levels(Mental$mental), levels(Mental$ses))
> scores$lower <- scores[,1] - scores[,2]
> scores$upper <- scores[,1] + scores[,2]
> scores

          Estimate Std. Error factor   lower  upper
Well         1.674      0.190 mental  1.4834  1.864
Mild         0.140      0.200 mental -0.0601  0.340
Moderate    -0.137      0.279 mental -0.4162  0.143
Impaired    -1.411      0.174 mental -1.5847 -1.236
1            1.111      0.299    ses  0.8121  1.411
2            1.120      0.314    ses  0.8062  1.435
3            0.371      0.319    ses  0.0516  0.690
4           -0.027      0.273    ses -0.3003  0.246
5           -1.009      0.315    ses -1.3242 -0.695
6           -1.817      0.281    ses -2.0976 -1.536
```

The dotchart shown in Figure 10.5 is then a plot of `Estimate`, grouped by `factor`, with arrows showing the range of `lower` to `upper` for each parameter.

```
> with(scores, {
+   dotchart(Estimate, groups = factor, labels = rownames(scores),
+            cex = 1.2, pch = 16, xlab = "RC1 Score",
+            xlim = c(min(lower), max(upper)))
+   arrows(lower, c(8 + (1 : 4), 1 : 6), upper, c(8 + (1 : 4), 1 : 6),
+          col = "red", angle = 90, length = .05, code = 3, lwd = 2)
+   })
```

In this plot, the main substantive difference from the $L \times L$ model is in the spacing of the lowest two categories of `ses` and the middle two categories of `mental`, which are not seen to differ in the RC1 model.

The coefficients in the `RC2` model can also be plotted (in a 2D plot) by extracting the coefficients from the "gnm" object and reshaping them to 2-column matrices. The function `pickCoef()` is handy here to get the indices of a subset of parameters by matching a pattern in their names.

```
> alpha <- coef(RC2)[pickCoef(RC2, "[.]mental")]
> alpha <- matrix(alpha, ncol=2)
> rownames(alpha) <- levels(Mental$mental)
> colnames(alpha) <- c("Dim1", "Dim2")
> alpha

               Dim1      Dim2
Well       0.497610 -0.192275
Mild       0.042652 -0.039651
Moderate   0.127071  0.208333
Impaired -0.505584 -0.012349

> beta <- coef(RC2)[pickCoef(RC2, "[.]ses")]
> beta <- matrix(beta, ncol=2)
> rownames(beta) <- levels(Mental$ses)
> colnames(beta) <- c("Dim1", "Dim2")
> beta

        Dim1      Dim2
1  0.547607 -0.167775
2  0.574184 -0.082320
3  0.205737  0.078252
4 -0.087662 -0.349440
5 -0.502335  0.059357
6 -0.758785  1.233043
```

For plotting and interpretation, these dimension scores need to be standardized as described at the start of this section (e.g., Eqn. (10.5)). We don't show the steps for doing this or producing a plot, because it is much simpler to use the logmult package, as described next. A basic plot using the marginal-weighted scaling is shown in Figure 10.6.

The patterns of the row and column category scores in Figure 10.6 are quite similar to the 2D correspondence analysis solution shown in Figure 6.2. The main difference is in the relative scaling of the axes. In Figure 10.6, the variances of the two dimensions are equated; in the correspondence analysis plot, the axes are scaled in relation to their contributions to Pearson χ^2, allowing an interpretation of distance between points in terms of χ^2-distance.

△

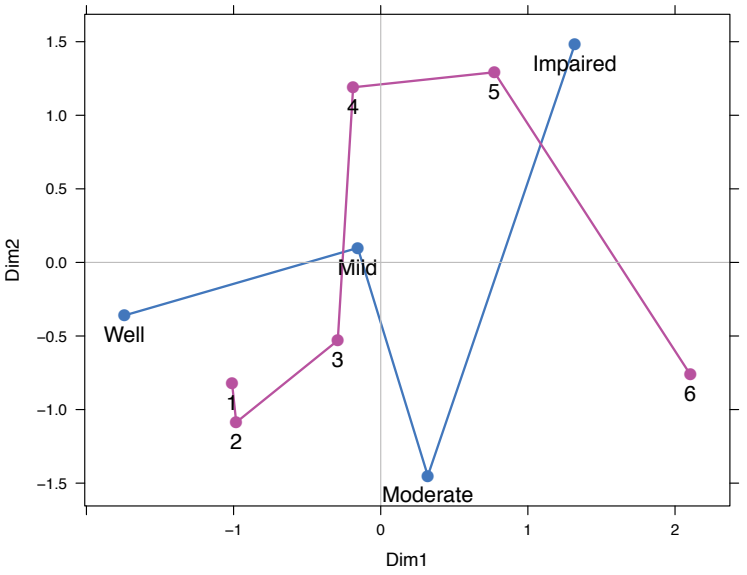


Figure 10.6: Scaled category scores for the RC(2) model fit the mental health data.

10.1.3.1 Using logmult

It takes a fair bit of work to extract the coefficients from "gnm" objects and carry out the scaling necessary for informative plots. Much of this effort is now performed by the logmult package with several convenience functions that do the heavy lifting.

`rc()` fits the class of RC(M) models, allowing an argument `nd` to specify the number of dimensions, and also providing for standard errors estimated using jackknife and bootstrap methods (Milan and Whittaker, 1995), which are computationally intensive. For square tables, a `symmetric` argument constrains the row and column scores to be equal, and a `diagonal` option fits parameters for each diagonal cell, providing for models of quasi-independence and quasi-symmetry (see Section 10.2).

It returns an object of class "rc" with the components of the "gnm" object. An `assoc` component is also returned, containing the normalized association parameters for the categories.

`rcL()` fits extensions of RC models to tables with multiple layers, called RC(M)-L models by Wong (2010).

`plot.rc()` is a plot method for visualizing scores for RC(M) models in two selected dimensions. Among other options, it can plot confidence ellipses for the category scores, using the estimated covariance matrix (assuming a normal distribution of the category scores). The plot method returns (invisibly) the coordinates of the scores as plotted, facilitating additional plot annotation.

EXAMPLE 10.4: Mental impairment and parents' SES

Here we use `rc()` to estimate the RC(1) and RC(2) models for the *`Mental`* data. In contrast to `gnm()`, which has a formula interface for a `data` argument, `rc()` requires the input in the form of a two-way table, given here as `mental.tab`.

```
> library(logmult)
> rc1 <- rc(mental.tab, verbose = FALSE, weighting = "marginal",
+           se = "jackknife")
> rc2 <- rc(mental.tab, verbose = FALSE, weighting = "marginal", nd = 2,
+           se = "jackknife")
```

The option `weighting="marginal"` gives the marginally weighted solution and `se = "jackknife"` estimates the covariance matrix using the leave-one-out jackknife.[4]

A plot of the scaled category scores similar to Figure 10.6, with 1 standard error confidence ellipses (making them comparable to the 1D solution shown in Figure 10.5) but no connecting lines can then be easily produced with the `plot()` method for "rc" objects.

```
> coords  <- plot(rc2, conf.ellipses = 0.68, cex = 1.5,
+                 rev.axes = c(TRUE, FALSE))
```

The orientation of the axes is arbitrary in RC(M) models, so the horizontal axis is reversed here to conform with Figure 10.6.

This produces (in Figure 10.7) a symmetric biplot in which the scaled coordinates of points for rows (α_{ik}) and columns (β_{jk}) on both axes are the product of normalized scores and the square root of the intrinsic association coefficient (γ_k) corresponding to each dimension.

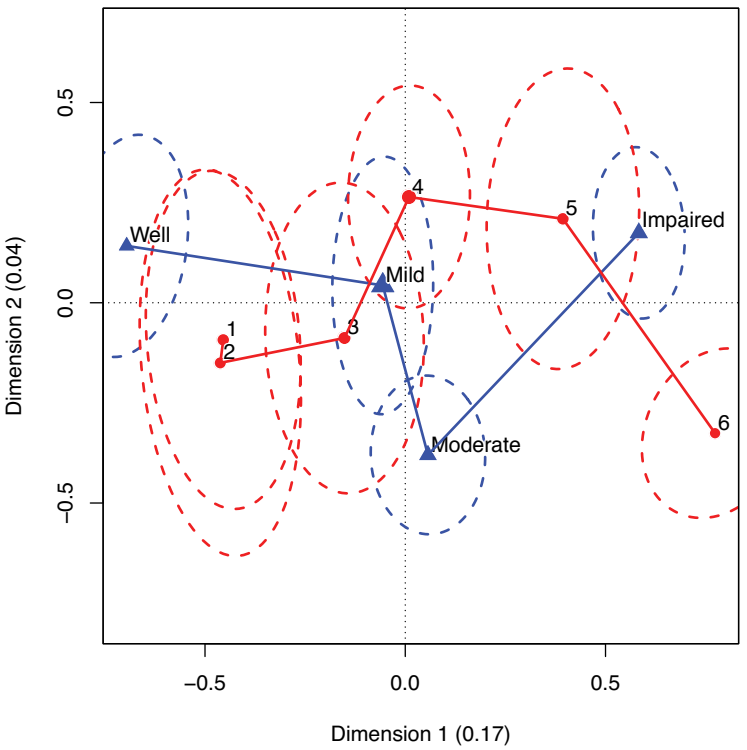


Figure 10.7: Scaled category scores for the RC(2) model fit and plotted using the logmult package. The 68% confidence ellipses correspond to bivariate ± 1 confidence intervals for the category parameters.

[4] Becker and Clogg (1989) recommend using unweighted solutions, `weighting="none"` (they call them "uniformly weighted") to preserve independence of inferences about association and marginal effects and estimates of the intrinsic association parameters, γ_k. That choice makes very little difference in the plots for this example, but the γ_k parameters are affected considerably.

Such plots can be customized using the category coordinates (`coords`) returned by the `plot()` method. As in other biplots, joining the row and column points by lines (sorted by the first dimension) makes it easier to see their relationships across the two dimensions. The following code draws the lines shown in Figure 10.7.

```
> scores <- rbind(coords$row, coords$col)
> lines(scores[1 : 4,], col = "blue", lwd = 2)
> lines(scores[-(1 : 4),], col = "red", lwd = 2)
```

We saw earlier that there was not strong evidence supporting the need for a second RC dimension to describe the relationship between mental health and SES. This is apparent in the sizes of the confidence ellipses, which overlap much more along Dimension 2 than Dimension 1. △

10.2 Square tables

Square tables, where the row and column variables have the same categories, comprise an important special case for loglinear models that can account for associations more parsimoniously than the saturated model. Some examples are the data on visual acuity in Example 4.14, categorical ratings of therapy clients by two observers, and mobility tables, tracking the occupational categories between generations in the same families, or migration tables, giving movement of people between regions. The latter topic has been important in sociological and geographic research and has spurred the development of a wide range of specialized loglinear models for this purpose.

10.2.1 Quasi-independence, symmetry, quasi-symmetry, and topological models

In many square tables, such as the `Vision` data, independence is not a credible hypothesis because the diagonal cells, representing equal values of the row and column variables, tend to be very large and often contribute most of the lack of fit. A substantively more interesting hypothesis is whether the table exhibits independence, ignoring the diagonal cells. This leads to what is called the ***quasi-independence model***, that specifies independence only in the off-diagonal cells.

For a two-way table, quasi-independence can be expressed as

$$\pi_{ij} = \pi_{i+}\pi_{+j} \qquad \text{for } i \neq j$$

or in loglinear form as

$$\log m_{ij} = \mu + \lambda_i^A + \lambda_j^B + \delta_i I(i = j) \; .$$

This model effectively adds one parameter, δ_i, for each main diagonal cell that fits those frequencies perfectly.

Another hypothesis of substantive interest for square tables, particularly those concerning occupational and geographical mobility, is that the joint distribution of row and column variables is symmetric, that is, $\pi_{ij} = \pi_{ji}$ for all $i \neq j$. For example, this ***symmetry model*** (S) asserts that sons are as likely to move from their father's occupation i to another, j, as the reverse. This form of symmetry is quite strong, because it also implies ***marginal homogeneity*** (MH), that the marginal probabilities of the row and column variables are equal, $\pi_{i+} = \sum_j \pi_{ij} = \sum_j \pi_{ji} = \pi_{+i}$ for all i.

To separate marginal homogeneity from symmetry of the association terms per se, the model of ***quasi-symmetry*** (QS) uses the standard main-effect terms in the loglinear model,

$$\log m_{ij} = \mu + \lambda_i^A + \lambda_j^B + \lambda_{ij} \; , \tag{10.7}$$

where $\lambda_{ij} = \lambda_{ji}$. It can be shown (Caussinus, 1966) that

$$\begin{aligned} \text{symmetry} &= \text{quasi-symmetry} + \text{marginal homogeneity} \\ G^2(S) &= G^2(QS) + G^2(MH) \end{aligned}$$

where $G^2(MH)$ is defined by the likelihood-ratio test of the difference between the S and QS models,

$$G^2(MH) \equiv G^2(S \,|\, QS) = G^2(S) - G^2(QS)\,. \tag{10.8}$$

The gnm package provides several model building convenience functions that facilitate fitting these and related models:

- `Diag(row, col, ...)` constructs a diagonals association factor for two (or more) factors with integer levels where the original factors are equal, and `"."` otherwise.
- `Symm(row, col, ...)` constructs an association factor giving equal levels to sets of symmetric cells. The QS model is specified using `Diag() + Symm()`.
- `Topo(row, col, ..., spec)` creates an association factor for two or more factors, as specified by an array of levels, which may be arbitrarily structured. Both `Diag()` and `Symm()` factors are special cases of `Topo()`.

The factor levels representing these association effects for a 4×4 table are shown below by their unique values in each array.

$$\text{Diag}_{4\times4} = \begin{bmatrix} 1 & . & . & . \\ . & 2 & . & . \\ . & . & 3 & . \\ . & . & . & 4 \end{bmatrix} \quad \text{Symm}_{4\times4} = \begin{bmatrix} 11 & 12 & 13 & 14 \\ 12 & 22 & 23 & 24 \\ 13 & 23 & 33 & 34 \\ 14 & 24 & 34 & 44 \end{bmatrix} \quad \text{Topo}_{4\times4} = \begin{bmatrix} 2 & 3 & 4 & 4 \\ 3 & 3 & 4 & 4 \\ 4 & 4 & 5 & 5 \\ 4 & 4 & 5 & 1 \end{bmatrix}.$$

EXAMPLE 10.5: Visual acuity

Example 4.14 presented the data on tests of visual acuity in the left and right eyes of a large sample of women working in the Royal Ordnance factories in World War II. A sieve diagram (Figure 4.10) showed that, as expected, most women had the same acuity in both eyes, but the off-diagonal cells had a pattern suggesting some form of symmetry.

The data set *VisualAcuity* contains data for both men and women in frequency form and for this example we subset this to include only the 4×4 table for women.

```
> data("VisualAcuity", package="vcd")
> women <- subset(VisualAcuity, gender=="female", select=-gender)
```

The four basic models of independence, quasi-independence, symmetry, and quasi-symmetry for square tables are fit as shown below. We use `update()` to highlight the relations among these models in two pairs.

```
> #library(vcdExtra)
> indep <- glm(Freq ~ right + left,  data = women, family = poisson)
> quasi <- update(indep, . ~ . + Diag(right, left))
>
> symm <- glm(Freq ~ Symm(right, left), data = women, family = poisson)
> qsymm <- update(symm, . ~ right + left + .)
```

The brief summary of goodness of fit of these models below shows that the QS model fits reasonably well, but none of the others do by likelihood-ratio tests or AIC or BIC.

```
> LRstats(indep, quasi, symm, qsymm)

Likelihood summary table:
       AIC  BIC LR Chisq Df Pr(>Chisq)
indep 6803 6808     6672  9     <2e-16 ***
quasi  338  347      199  5     <2e-16 ***
symm   157  164       19  6     0.0038 **
qsymm  151  161        7  3     0.0638 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Beyond just saying that the QS model fits best, the reasons *why* it does can be seen in mosaic displays. Figure 10.8 compares the mosaics for the models of quasi-independence (accounting only for the diagonal cells) and quasi-symmetry (also accounting for symmetry). It can be seen in the left panel that the non-diagonal associations are largely symmetric, and also that when they differ, visual acuity in the two eyes is most likely to differ by only one eye grade.

```
> labs <- c("High", "2", "3", "Low")
> largs <- list(set_varnames = c(right = "Right eye grade",
+                                left = "Left eye grade"),
+               set_labels=list(right = labs, left = labs))
> mosaic(quasi, ~ right + left, residuals_type = "rstandard",
+        gp = shading_Friendly,
+        labeling_args = largs,
+        main = "Quasi-Independence (women)")
> mosaic(qsymm, ~ right + left, residuals_type = "rstandard",
+        gp = shading_Friendly,
+        labeling_args = largs,
+        main = "Quasi-Symmetry (women)")
```

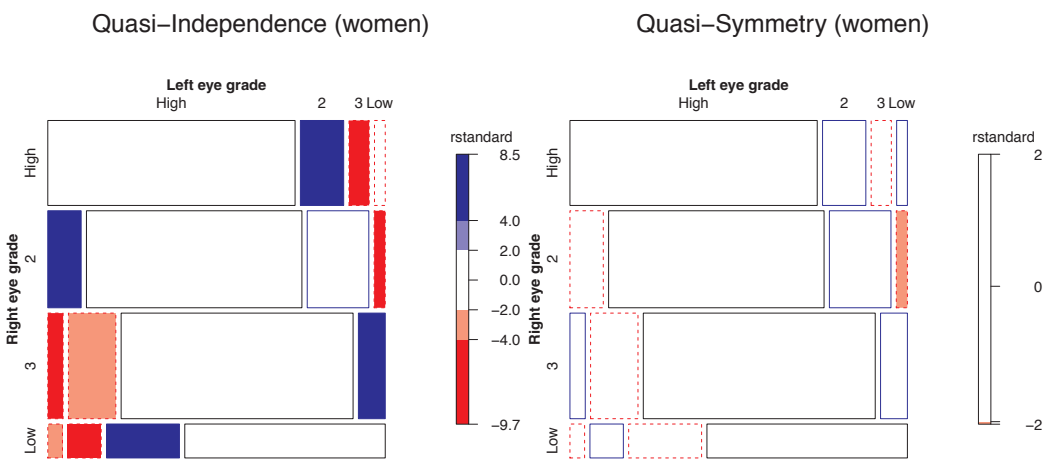


Figure 10.8: Mosaic displays comparing the models of quasi-independence and quasi-symmetry for visual acuity in women.

Finally, as usual, `anova()` can be used to carry out specific tests of nested models. For example, the test of marginal homogeneity, Eqn. (10.8), compares models S and QS and shows here that the marginal probabilities for the left and right eyes differ.

```
> anova(symm, qsymm, test = "Chisq")

Analysis of Deviance Table

Model 1: Freq ~ Symm(right, left)
Model 2: Freq ~ right + left + Symm(right, left)
  Resid. Df Resid. Dev Df Deviance Pr(>Chi)
1         6      19.25
2         3       7.27  3       12   0.0075 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

△

EXAMPLE 10.6: Hauser's occupational mobility table

The data *Hauser79* in vcdExtra, from Hauser (1979), gives a 5×5 table in frequency form cross-classifying 19,912 individuals in the United States by father's occupation and son's first occupation. The occupational categories are represented by abbreviations, of Upper Non-Manual (UpNM), Lower Non-Manual (LoNM), Upper Manual (UpM), Lower Manual (LoM), and Farm. These data were also analyzed by Powers and Xie (2008, Chapter 4).

```
> data("Hauser79", package = "vcdExtra")
> structable(~ Father + Son, data = Hauser79)

        Son UpNM LoNM  UpM  LoM Farm
Father
UpNM       1414  521  302  643   40
LoNM        724  524  254  703   48
UpM         798  648  856 1676  108
LoM         756  914  771 3325  237
Farm        409  357  441 1611 1832
```

Before fitting any models, it is useful to calculate and plot the observed local log odds ratios, as we did in Example 10.1, to see the patterns in the data that need to be accounted for. These are calculated using `loddsratio()`.

```
> hauser.tab <- xtabs(Freq ~ Father + Son, data = Hauser79)
> (lor.hauser <- loddsratio(hauser.tab))

log odds ratios for Father and Son

           Son
Father      UpNM:LoNM LoNM:UpM UpM:LoM LoM:Farm
  UpNM:LoNM     0.675   -0.179   0.262   0.0931
  LoNM:UpM      0.115    1.003  -0.346  -0.0579
  UpM:LoM       0.398   -0.449   0.790   0.1009
  LoM:Farm     -0.326    0.381  -0.166   2.7697
```

This 4×4 table is graphed using `plot(lor.hauser)`, giving Figure 10.9.

```
> plot(lor.hauser, confidence = FALSE, legend_pos = "topleft",
+      xlab = "Father's status comparisons")
> m <- mean(lor.hauser$coefficients)        # mean LOR
> grid.lines(x = unit(c(0, 1), "npc"),
+            y = unit(c(m, m), "native"))
```

Among the features here, you can see that there is a tendency for the odds ratio contrasting sons in the non-manual categories (UpNM:LoNM) to decline with the adjacent comparisons of their

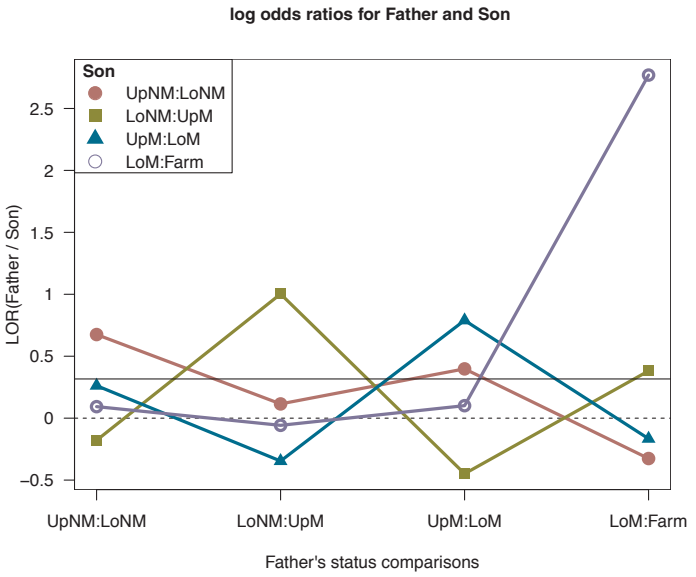


Figure 10.9: Plot of observed local log odds ratios in the Hauser79 data. The dotted horizontal line at zero shows local independence; the solid black horizontal line shows the mean.

fathers' occupations. As well, the 2×2 table for fathers and sons in the `LoM:Farm` stands out as deserving some attention. These observed features will be smoothed by fitting models, as described below. For additional interpretation, you can always construct similar plots of the log odds ratios using the `fitted()` values (see: Section 10.1.2) from any of the models described below.

We begin by fitting the independence model and the quasi-independence model, where the diagonal parameters in the latter are specified as `Diag(Father,Son)`. As expected, given the large frequencies in the diagonal cells, the quasi-independence model is a considerable improvement, but the fit is still very poor.

```
> hauser.indep <- gnm(Freq ~ Father + Son, data = Hauser79,
+                     family = poisson)
> hauser.quasi <-  update(hauser.indep, ~ . + Diag(Father, Son))
> LRstats(hauser.indep, hauser.quasi)

Likelihood summary table:
              AIC  BIC LR Chisq Df Pr(>Chisq)
hauser.indep 6391 6402     6170 16     <2e-16 ***
hauser.quasi  914  931      683 11     <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The pattern of associations can be seen in the mosaic displays for both models, shown in Figure 10.10.

```
> mosaic(hauser.indep, ~ Father + Son, main = "Independence model",
+        gp = shading_Friendly)
> mosaic(hauser.quasi, ~ Father + Son, main = "Quasi-independence model",
+        gp = shading_Friendly)
```

The mosaic for quasi-independence shows an approximately symmetric pattern of residuals, so we proceed to add `Symm(Father,Son)` to the model to specify quasi-symmetry.

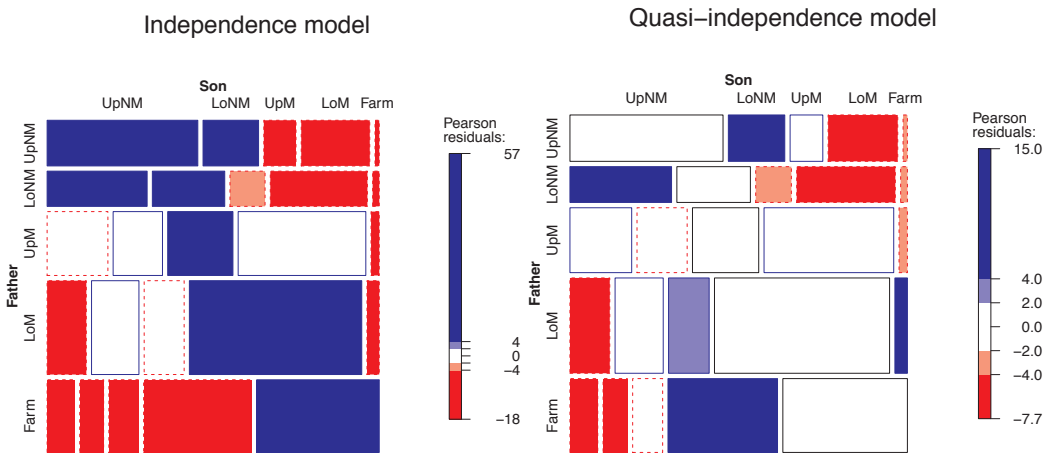


Figure 10.10: Mosaic displays for the Hauser79 data. Left: independence model; right:quasi-independence model.

```
> hauser.qsymm <-  update(hauser.indep,
+                         ~ . + Diag(Father, Son) + Symm(Father, Son))
> LRstats(hauser.qsymm)

Likelihood summary table:
             AIC BIC LR Chisq Df Pr(>Chisq)
hauser.qsymm 268 291     27.4  6    0.00012 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

This model represents a huge improvement in goodness of fit. With such a large sample size, it might be considered an acceptable fit. The remaining lack of fit is shown in the mosaic for this model, Figure 10.11.

```
> mosaic(hauser.qsymm, ~ Father + Son, main = "Quasi-symmetry model",
+        gp = shading_Friendly, residuals_type = "rstandard")
```

The cells with the largest lack of symmetry (using standardized residuals) are those for the upper and lower non-manual occupations, where the son of an upper manual worker is less likely to move to lower non-manual work than the reverse.

For cases like this involving structured associations in square tables, Hauser (1979) developed the more general idea of grouping the row and column categories into levels of an association factor based on similar values of residuals or local odds ratios observed from the independence model. Such models are called ***topological model***s or ***levels model***s, which are implemented in the `Topo()` function.

To illustrate, Hauser suggested the following matrix of levels to account for the pattern of associations seen in Figure 10.10. The coding here takes the diagonal cell for the Farm category as the reference cell. Four other parameters are assigned by the numbers 2–5 to account for lack of independence.

```
> levels <- matrix(c(
+   2,  4,  5,  5,  5,
+   3,  4,  5,  5,  5,
```

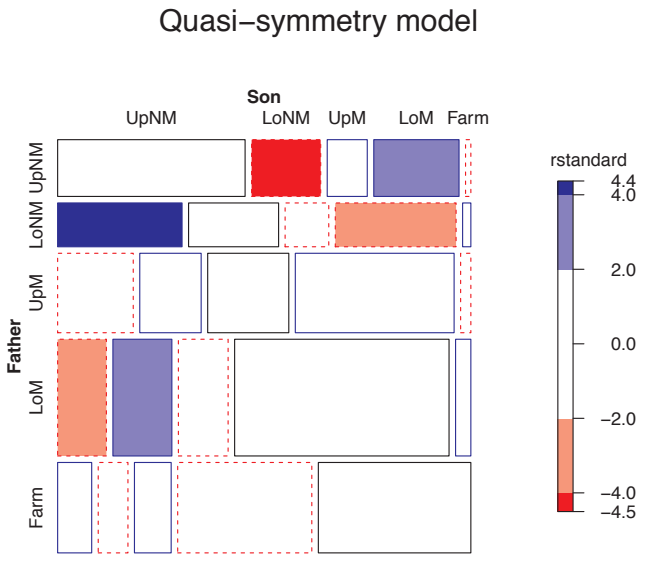


Figure 10.11: Mosaic display for the model of quasi-symmetry fit to the Hauser79 data.

```
+    5,  5,  5,  5,  5,
+    5,  5,  5,  4,  4,
+    5,  5,  5,  4,  1
+    ),
+    5, 5, byrow = TRUE)
```

This model is fit using `Topo()` as shown below. It also provides a huge improvement over the independence model, with 4 additional parameters.

```
> hauser.topo <- update(hauser.indep,
+                       ~ . + Topo(Father, Son, spec = levels))
> LRstats(hauser.topo)

Likelihood summary table:
            AIC BIC LR Chisq Df Pr(>Chisq)
hauser.topo 295 311     66.6 12    1.4e-09 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

As with other models fit using `gnm()`, you can extract the coefficients for particular terms using `pickCoef()`.

```
> as.vector((coef(hauser.topo)[pickCoef(hauser.topo, "Topo")]))

[1] -1.8128 -2.4973 -2.8035 -3.4026
```

The models fit in this example are summarized below. Both AIC and BIC prefer the quasi-symmetry model, `hauser.quasi`.

```
> LRstats(hauser.indep, hauser.quasi, hauser.qsymm, hauser.topo)

Likelihood summary table:
              AIC  BIC LR Chisq Df Pr(>Chisq)
```

```
hauser.indep 6391 6402     6170 16    < 2e-16 ***
hauser.quasi  914  931      683 11    < 2e-16 ***
hauser.qsymm  268  291       27  6    0.00012 ***
hauser.topo   295  311       67 12    1.4e-09 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

△

10.2.2 Ordinal square tables

The theory presented in Section 10.2.1 treats the row and column variables as nominal. In many instances, such as Example 10.6, the variable categories are also ordered, yet these models do not exploit their ordinal nature. In such cases, the models such as uniform association ($L \times L$), row effects, RC, and others discussed in Section 10.1 can be combined with terms for quasi-independence and symmetry of the remaining associations.

For example, the $L \times L$ model Eqn. (10.2) of uniform association applies directly to square tables, and, for square tables, can also be amended to include a diagonals term, `Diag()`, giving a model of *quasi-uniform association*. In this model, all adjacent 2×2 sub-tables not involving diagonal cells have a common local odds ratio.

A related model is the ***crossings model*** (Goodman, 1972). This hypothesizes that there are different difficulty parameters for crossing from one category to the next, and that the associations between categories decreases with their separation. In the crossings model for an $I \times I$ table, there are $I - 1$ crossings parameters, $\nu_1, \nu_2, \ldots, \nu_{I-1}$. The association parameters, λ_{ij}^{AB} have the form of the product of the intervening ν parameters,

$$\lambda_{ij}^{AB} = \begin{cases} \prod_{k=j}^{k=i-1} \nu_k & : \quad i > j \\ \prod_{k=i}^{k=j-1} \nu_k & : \quad i < j \end{cases} .$$

This model can also be cast in *quasi* form, by addition of a `Diag` term to fit the main diagonal cells. See Powers and Xie (2008, Section 4.4.7) for further details of this model. The `Crossings()` function in vcdExtra implements such crossings terms.

EXAMPLE 10.7: Hauser's occupational mobility table

Without much comment or detail, for reference we first fit some of the ordinal models to the *Hauser79* data: Uniform association ($L \times L$), row effects, and the RC(1) model.

```
> Fscore <- as.numeric(Hauser79$Father)    # numeric scores
> Sscore <- as.numeric(Hauser79$Son)       # numeric scores
>
> # uniform association
> hauser.UA <- update(hauser.indep, ~ . + Fscore * Sscore)
> # row effects model
> hauser.roweff <- update(hauser.indep, ~ . + Father * Sscore)
> # RC model
> hauser.RC <- update(hauser.indep,
+                     ~ . + Mult(Father, Son), verbose = FALSE)
```

All of these fit very poorly, yet they are all substantial improvements over the independence model.

```
> LRstats(hauser.indep, hauser.UA, hauser.roweff, hauser.RC)

Likelihood summary table:
               AIC  BIC LR Chisq Df Pr(>Chisq)
hauser.indep  6391 6402     6170 16     <2e-16 ***
hauser.UA     2503 2516     2281 15     <2e-16 ***
hauser.roweff 2309 2325     2080 12     <2e-16 ***
hauser.RC      920  940      685  9     <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The $L \times L$ model, `hauser.UA` might be improved by ignoring the diagonals, and, indeed it is.

```
> hauser.UAdiag <- update(hauser.UA, ~ . + Diag(Father, Son))
> anova(hauser.UA, hauser.UAdiag, test = "Chisq")

Analysis of Deviance Table

Model 1: Freq ~ Father + Son + Fscore + Sscore + Fscore:Sscore
Model 2: Freq ~ Father + Son + Fscore + Sscore + Fscore:Sscore + Diag(Father,
    Son)
  Resid. Df Resid. Dev Df Deviance Pr(>Chi)
1        15       2281
2        10         73  5     2208   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

In this model, the estimated common local log odds ratio—the coefficient for the linear-by-linear term `Fscore:Sscore`, is

```
> coef(hauser.UAdiag)[["Fscore:Sscore"]]

[1] 0.1584
```

For comparisons not involving the diagonal cells, each step down the scale of occupational categories for the father multiplies the odds that the son will also be in one lower category by $\exp(0.158) = 1.172$, an increase of 17%.

The crossings model, with and without the diagonal cells, can be fit as follows:

```
> hauser.CR <- update(hauser.indep, ~ . + Crossings(Father, Son))
> hauser.CRdiag <- update(hauser.CR, ~ . + Diag(Father, Son))
> LRstats(hauser.CR, hauser.CRdiag)

Likelihood summary table:
              AIC BIC LR Chisq Df Pr(>Chisq)
hauser.CR     319 334     89.9 12    5.1e-14 ***
hauser.CRdiag 299 318     64.2  9    2.0e-10 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The quasi-crossings model `hauser.CRdiag` has a reasonable G^2 fit statistic, and its interpretation and lack of fit is worth exploring further. The crossings coefficients ν can be extracted as follows.

```
> nu <- coef(hauser.CRdiag)[pickCoef(hauser.CRdiag, "Crossings")]
> names(nu) <- gsub("Crossings(Father, Son)C", "nu", names(nu),
+                   fixed = TRUE)
> nu

     nu1      nu2      nu3      nu4
-0.42275 -0.38768 -0.27500 -1.40244
```

They indicate the steps between adjacent categories in terms of the barriers for a son moving to a lower occupational category. The numerically largest gap separates the lower non-manual category from farming.

In contrast to the `UAdiag` model, the quasi-crossing model with diagonal terms implies that all 2×2 off-diagonal sub-tables are independent, i.e., the local odds ratios are all equal to 1.0. The reasons for lack of fit of this model can be seen in the corresponding mosaic display, shown in Figure 10.12.

```
> mosaic(hauser.CRdiag, ~ Father + Son,
+        gp = shading_Friendly, residuals_type = "rstandard",
+        main = "Crossings() + Diag()")
```

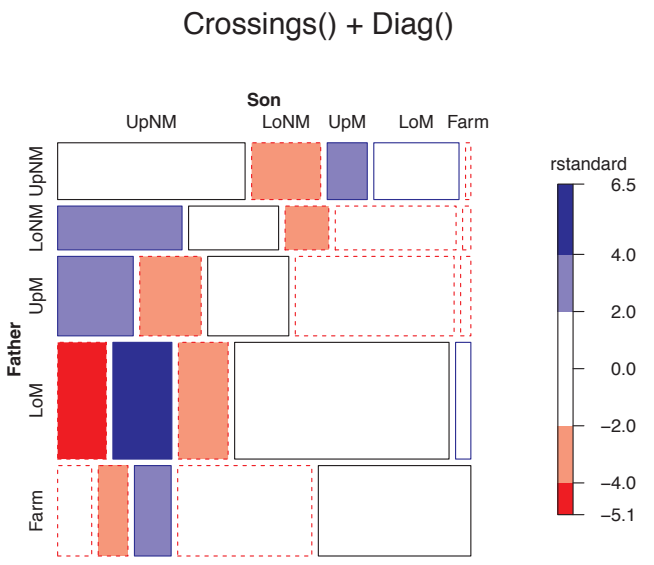


Figure 10.12: Mosaic display for the quasi-crossings model fit to the Hauser79 data.

It can be seen that lack of fit for this model is largely concentrated in the lower triangle, where the father's occupation is lower than that of his son.

In this example and the last, we have fit quite a few different models to the Hauser (1979) data. In presentations, articles and books, it is common to summarize such a collection in a table, sorted by G^2, degrees of freedom, AIC or BIC, to show their ordering along some metric. For instance, here we collect all the models fit in Example 10.6 and this example in a `glmlist()` and sort in decreasing order of BIC to show model fit by this measure.

```
> modlist <- glmlist(hauser.indep, hauser.roweff, hauser.UA,
+                    hauser.UAdiag, hauser.quasi, hauser.qsymm,
+                    hauser.topo, hauser.RC, hauser.CR, hauser.CRdiag)
> LRstats(modlist, sortby = "BIC")

Likelihood summary table:
               AIC  BIC LR Chisq Df Pr(>Chisq)
hauser.indep  6391 6402     6170 16    < 2e-16 ***
hauser.UA     2503 2516     2281 15    < 2e-16 ***
hauser.roweff 2309 2325     2080 12    < 2e-16 ***
```

```
hauser.RC       920  940       685  9    < 2e-16 ***
hauser.quasi    914  931       683 11    < 2e-16 ***
hauser.CR       319  334        90 12    5.1e-14 ***
hauser.UAdiag   306  324        73 10    1.2e-11 ***
hauser.CRdiag   299  318        64  9    2.0e-10 ***
hauser.topo     295  311        67 12    1.4e-09 ***
hauser.qsymm    268  291        27  6    0.00012 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

When there are more than just a few models, a more useful display is a ***model comparison plot*** of measures like G^2/df, AIC, or BIC against degrees of freedom. For example, Figure 10.13 plots `BIC` against `Df` from the result of `LRstats()`. Because interest is focused on the smallest values of BIC and these values span a large range, BIC is shown on the log scale using `log="y"`.

```
> sumry <- LRstats(modlist)
> mods <- substring(rownames(sumry), 8)
> with(sumry, {
+   plot(Df, BIC, cex = 1.3, pch = 19,
+        xlab = "Degrees of freedom", ylab = "BIC (log scale)",
+        log = "y", cex.lab = 1.2)
+   pos <- ifelse(mods == "UAdiag", 1, 3)
+   text(Df, BIC + 55, mods, pos = pos, col = "red", xpd = TRUE, cex = 1.2)
+   })
```

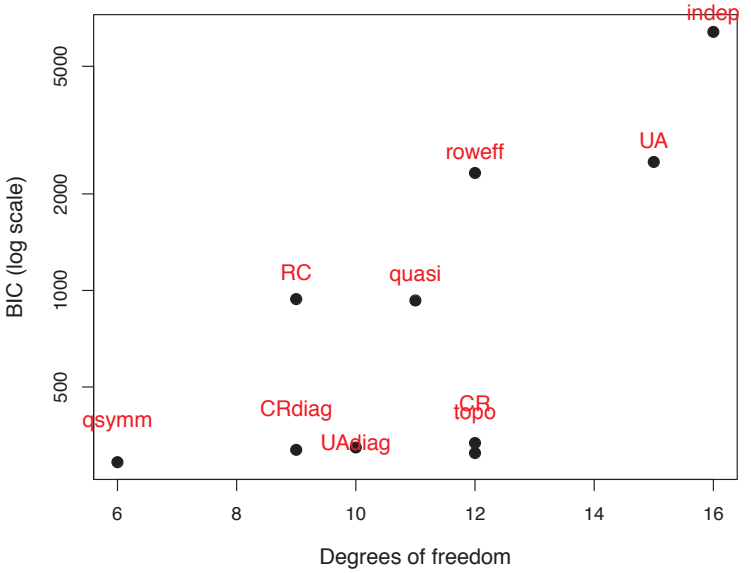


Figure 10.13: Model comparison plot for the models fit to the Hauser79 data.

Compared with the sorted tabular display shown above, such a plot sorts the models *both* by a measure of fit and by model complexity (degrees of freedom). Figure 10.13 shows that the quasi-symmetry model is best by BIC, but also shows that the next four best models by this measure are quite similar in terms of BIC. Similar plots for AIC and G^2/df show that the model of quasi-symmetry is favored by these measures.

△

10.3 Three-way and higher-dimensional tables

The models and methods for ordinal factors and square tables described in Section 10.1 and Section 10.2 extend readily to multidimensional tables with these properties for some of the factors. In three-way tables, these models provide a more parsimonious account than the saturated model, $[ABC]$, and also allow simpler models than the general model of homogeneous association, $[AB][AC][BC]$, using scores for ordinal factors or terms for symmetry and diagonal factors in square layers.

For example, consider the case where all three factors are ordinal and the model of homogeneous association $[AB][AC][BC]$ fits poorly. In this case we can generalize the model of uniform association by assigning scores $\boldsymbol{a}$, $\boldsymbol{b}$, and $\boldsymbol{c}$ and model the three-way association, λ_{ijk}^{ABC} as

$$\lambda_{ijk}^{ABC} = \gamma a_i b_j c_k$$

with only one more parameter. This gives the model of ***uniform interaction*** (or *homogeneous uniform association*)

$$\log(m_{ijk}) = \mu + \lambda_i^A + \lambda_j^B + \lambda_k^C + \lambda_{ij}^{AB} + \lambda_{ik}^{AC} + \lambda_{jk}^{BC} + \gamma a_i b_j c_k \; . \tag{10.9}$$

This model posits that (with equally spaced scores) all local odds ratios θ_{ijk} in adjacent rows, columns, and layers are constant,

$$\log(\theta_{ijk}) = \gamma \qquad \forall \quad i, j, k \; .$$

The homogeneous association model is the special case of $\log \theta_{ijk} = \gamma = 0$.

A less restricted model of ***heterogeneous uniform association*** retains the linear-by-linear form of association for factors A and B, but allows the strength of this association to vary over layers, C, representing λ_{ijk}^{ABC} as

$$\lambda_{ijk}^{ABC} = (\gamma + \gamma_k) a_i b_j$$

with the constraint $\sum_k \gamma_k = 0$. This model is equivalent to fitting separate models of uniform association at each level k of factor C and gives estimates of the conditional local log odds ratios, $\log \theta_{ij(k)} = \gamma + \gamma_k$.

Following the development in Section 10.1 there is a large class of other models for ordinal factors (see Figure 10.1), where not all factors are assigned scores. For three-way tables, these can be represented in homogeneous form when the two-way association of A and B is the same for all levels of C, or in a heterogeneous form, when it varies over C.

Similarly, the models for square tables described in Section 10.2 extend to three-way tables with several layers (strata), allowing both homogeneous and heterogeneous terms for diagonals and symmetry describing the AB association over levels of C.

EXAMPLE 10.8: Visual acuity

We continue the analysis of the `VisualAcuity` data, but now consider the three-way, $4 \times 4 \times 2$ table comprising both men and women. The main questions here are whether the pattern of quasi-symmetry observed in the analysis for women also pertains to men and whether there is heterogeneity of the association between right and left acuity across gender.

A useful first step for n-dimensional tables is to consider the models composed of all 1-way, 2-way, ... n-way terms as a quick overview. The function `Kway()` in the vcdExtra package does this automatically, returning a "glmlist" object containing the fitted models. That is, for this problem, `Kway()` generates (and fits) the following model formulae, also including the 0-way model, corresponding to $\log m_{ijk\ldots} = \mu$.

```
> Freq ~ 1
> Freq ~ right + left + gender
> Freq ~ (right + left + gender)^2
> Freq ~ (right + left + gender)^3
```

We use `Kway()` as follows:

```
> vis.kway <- Kway(Freq ~ right + left + gender, data = VisualAcuity)
> LRstats(vis.kway)

Likelihood summary table:
         AIC   BIC LR Chisq Df Pr(>Chisq)
kway.0 13857 13858    13631 31    < 2e-16 ***
kway.1  9925  9937     9686 24    < 2e-16 ***
kway.2   298   332       28  9    0.00079 ***
kway.3   287   334        0  0    < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

This shows that the model of homogeneous association `kway.2` ($[RL][RG][LG]$) does not fit well, but it doesn't account for diagonal agreement or symmetry to simplify the associations.

As a basis for comparison, we first fit the simple models of quasi-independence and quasi-symmetry that do not involve `gender`, asserting the same pattern of diagonal and off-diagonal cells for males and females.

```
> vis.indep <- glm(Freq ~ right + left + gender,  data = VisualAcuity,
+                  family = poisson)
> vis.quasi <- update(vis.indep, . ~ . + Diag(right, left))
> vis.qsymm <- update(vis.indep, . ~ . + Diag(right, left)
+                                      + Symm(right, left))
>
> LRstats(vis.indep, vis.quasi, vis.qsymm)

Likelihood summary table:
           AIC  BIC LR Chisq Df Pr(>Chisq)
vis.indep 9925 9937     9686 24     <2e-16 ***
vis.quasi  696  714      449 20     <2e-16 ***
vis.qsymm  435  456      184 18     <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The model of homogeneous quasi-symmetry fits quite badly, even worse than the all two-way association model. We can see why in the mosaic for this model, shown in Figure 10.14.

```
> mosaic(vis.qsymm, ~ gender + right + left, condvars = "gender",
+        residuals_type = "rstandard", gp = shading_Friendly,
+        labeling_args = largs, rep = FALSE,
+        main = "Homogeneous quasi-symmetry")
```

It can be seen in Figure 10.14 that the pattern of residuals for men and women are nearly completely opposite in the upper and lower portions of the plot: men have positive residuals in the same `right`, `left` cells where women have negative residuals, and vice-versa. In particular, the diagonal cells of both tables have large absolute residuals, because the term `Diag(right, left)` fits a common set of diagonals for both men and women.

We can correct for this by allowing separate diagonal and symmetry terms, given as interactions of `gender` with `Diag()` and `Symm()`.

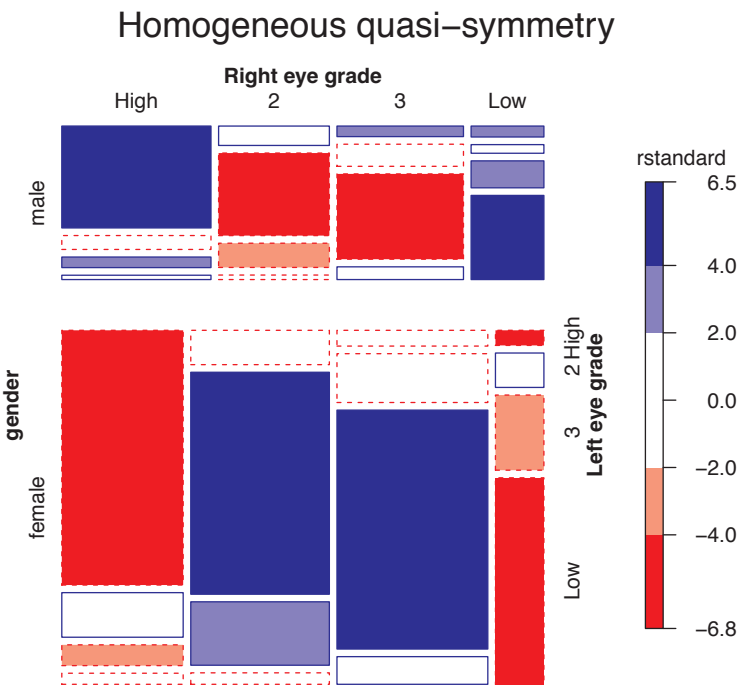


Figure 10.14: Mosaic display for the model of homogeneous quasi-symmetry fit to the VisualAcuity data.

```
> vis.hetdiag <- update(vis.indep, . ~ . + gender * Diag(right, left) +
+                       Symm(right, left))
> vis.hetqsymm <- update(vis.indep, . ~ . + gender * Diag(right, left) +
+                        gender * Symm(right, left))
> LRstats(vis.qsymm, vis.hetdiag, vis.hetqsymm)

Likelihood summary table:
             AIC BIC LR Chisq Df Pr(>Chisq)
vis.qsymm    435 456    183.7 18    < 2e-16 ***
vis.hetdiag  312 338     52.3 14    2.5e-06 ***
vis.hetqsymm 287 321     17.7  9      0.038 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Note that the model `vis.hetqsymm` fits better than the model `vis.hetdiag` in absolute terms, but the latter, with fewer parameters, fits better by AIC and BIC. The mosaic for the model `vis.hetqsymm` is shown in Figure 10.15.

```
> mosaic(vis.hetqsymm, ~ gender + right + left, condvars="gender",
+        residuals_type = "rstandard", gp = shading_Friendly,
+        labeling_args = largs, rep = FALSE,
+        main="Heterogeneous quasi-symmetry")
```

As in the two-way case, this model now fits the diagonal cells in each table exactly, effectively ignoring this part of the association between right and left eye acuity. All remaining residuals are relatively small in magnitude, except for the two opposite off-diagonal cells `(Low, High)` and `(High, Low)` in the table for women.

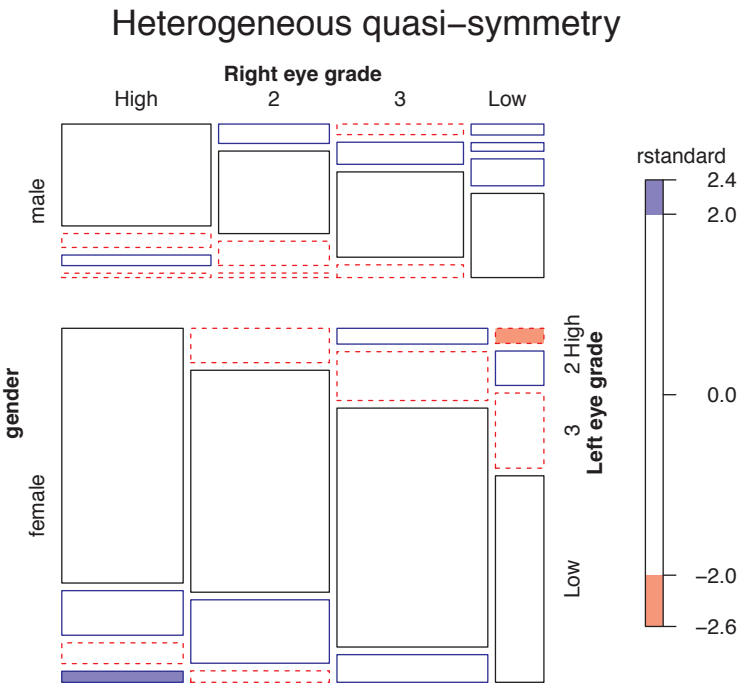


Figure 10.15: Mosaic display for the model of heterogeneous quasi-symmetry fit to the VisualAcuity data.

The substantive interpretation of this example is that visual acuity is largely the same (diagonal cells) in the right and left eyes of both men and women. Ignoring the diagonal cells, when visual acuity differs, both men and women exhibit approximately symmetric associations. However, deviations from symmetry (Figure 10.14) are such that men are slightly more likely to have a lower grade in the right eye, while women are slightly more likely to have a higher grade in the right eye. △

10.4 Multivariate responses*

In many studies, there may be *several* categorical responses observed along with one or more explanatory variables. In a clinical trial, for example, the efficacy of a drug might be the primary response, but the occurrence of side-effects might give rise to additional response variables of substantive interest. Or, in a study of occupational health, the occurrence of two or more distinct symptoms might be treated as response variables.

If there are *no* explanatory variables, then the problem is simply to understand the joint distribution of the response categories, and the loglinear models and graphical displays described earlier are sufficient. Otherwise, in these cases we usually wish to understand how the various responses are affected by the explanatory variables. Moreover, it may also be important to understand how the association between the categorical responses depends on the explanatory variables. That is, we would like to study how *both* the marginal distributions of the responses, and their joint distribution depends on the predictors. In the occupational health example, the goal might be to understand both

how the prevalence of several symptoms varies with one or more predictors, and how the association (loosely, "correlation") among those symptoms varies with those predictors.

Although the general loglinear model is often used in these situations, there are special reparameterizations that may be used to separate the *marginal* dependence of each response on the explanatory variables from the relationship of the *association* among the responses on the explanatory variables.

Let us say that categorical responses, Y_1, $Y_2, \ldots$ have been observed, together with possible explanatory variables, X_1, $X_2, \ldots$, and let $\pi_{ij\ldots}$ be the joint probability of all the responses and explanatory variables; we also use $\boldsymbol{x}$ to refer to the values of X_1, $X_2, \ldots$.

Note that the minimal model of independence of all responses from each other and from the explanatory variables is the loglinear model $[Y_1][Y_2]\cdots[X_1X_2\cdots]$ (i.e., all associations among the X_i must be included). A no-effect model, in which the responses do not depend on the explanatory variables, but may be associated among themselves, is $[Y_1Y_2\cdots][X_1X_2\cdots]$. However, these models do not separate the individual (marginal) effects of $X_1, X_2\ldots$ on each Y_i from their associative effects on the joint relationships among the Y_i.

There are three useful general approaches that *do* separate these effects:

1. Model the marginal dependence of each response, Y_i, separately on X_1, $X_2, \ldots$, and, in addition, model the interdependence among the responses, $Y_1, Y_2, \ldots$.[5]

2. Model the joint dependence of all responses on X_1, $X_2, \ldots$, but parameterized so that marginal and associative effects are delineated.

3. Construct simultaneous models, estimated together, for the marginal and joint dependence of the responses on the explanatory variables.

The first approach is the simplest, an informative starting place, and is satisfactory in the (often unlikely) case that the responses are not associated, or if the associations among responses do not vary much over the explanatory variables (i.e., no terms like $[Y_1Y_2X_j]$ are required). In the clinical trial example, we would construct separate loglinear or logit models for efficacy of the drug, and for occurrence of side-effects, and supplement these analyses with mosaic or other displays showing the relations between efficacy and side-effects and a model for their joint association. If those who improve with the drug also show more serious side effects, the worth of the treatment would be questioned. A limitation of this method is that it does not provide an overall model comprising these effects.

In the second approach, the joint probabilities, $\pi_{ij\ldots}$, are recast to give separate information regarding the dependence of the univariate marginal probabilities $\pi_{i\bullet}, \pi_{\bullet j}, \ldots$, on the explanatory variables and the dependence of the intra-response associations on the explanatory variables. The VGAM package provides several versions of this approach with the function `vglm()` (for *vector generalized linear model*).

The third approach, developed, for example, by Lang and Agresti (1994), is the most general, and provides a scheme to represent a model $\mathcal{J}(\bullet)$ for the joint distributions of the X, Y variables together with a model $\mathcal{M}(\bullet)$ for their first-order marginal distributions. The joint models are typically loglinear models, ranging from the mutual independence model, $\mathcal{J}(I) = [Y_1][Y_2][\cdots][X_1][X_2][\cdots]$, to the saturated model, $\mathcal{J}(S) = [Y_1Y_2\cdots X_1X_2\cdots]$, while the marginal models are logit models for the response variables. The combined model, denoted $\mathcal{J}(\bullet) \cap \mathcal{M}(\bullet)$, is estimated simultaneously by maximum likelihood. This approach is implemented in R in the hmmm (Roberto et al., 2014) package (hierarchical multinomial marginal models). However, model specification in this implementation is complicated, and it will not be considered further here.

[5] For quantitative responses, this is roughly analogous to fitting univariate response models for each Y_i, followed by something like a principal component analysis of the relationships among the Y_i. But in this case, the multivariate linear model, $\boldsymbol{Y} = \boldsymbol{XB} + \boldsymbol{E}$ provides a general solution.

10.4.1 Bivariate, binary response models

We focus here on two related models reflecting the second approach, as discussed by McCullagh and Nelder (1989, Section 6.5). We consider here only the case of two binary responses, though the general approach can be applied to $R > 2$ responses $Y_1, Y_2, \ldots, Y_R$, and these may be polytomous or ordinal.

Let $\boldsymbol{x}$ refer to the values of all the explanatory variables and let $\pi_{ij}(\boldsymbol{x})$ be the joint probabilities in cell $Y_1 = i$, $Y_2 = j$. The essential idea of the ***bivariate logistic model*** arises from a linear transformation of the cell probabilities $\boldsymbol{\pi}$ to interpretable functions of the marginal probabilities (logits) and their association (odds ratio), a mapping of $\boldsymbol{\pi} \rightarrow \boldsymbol{\eta}$,

$$\begin{aligned} \eta_1 &= \text{logit}(\pi_{1\bullet}) \\ \eta_2 &= \text{logit}(\pi_{\bullet 1}) \\ \eta_{12} &= \log\left(\frac{\pi_{11}\,\pi_{22}}{\pi_{12}\,\pi_{21}}\right) . \end{aligned} \tag{10.10}$$

The predictors in $\boldsymbol{x}$ are then taken into account by considering models that relate $\boldsymbol{\pi}$ to $\boldsymbol{x}$ through $\boldsymbol{\eta}$,

$$\begin{aligned} \eta_1 &= \boldsymbol{x}_1^{\mathsf{T}}\boldsymbol{\beta}_1 \\ \eta_2 &= \boldsymbol{x}_2^{\mathsf{T}}\boldsymbol{\beta}_2 \\ \eta_{12} &= \boldsymbol{x}_{12}^{\mathsf{T}}\boldsymbol{\beta}_{12} , \end{aligned} \tag{10.11}$$

where $\boldsymbol{x}_1$, $\boldsymbol{x}_2$, and $\boldsymbol{x}_{12}$ are subsets of the predictors in $\boldsymbol{x}$ for each sub-model, and $\boldsymbol{\beta}_1$, $\boldsymbol{\beta}_2$, and $\boldsymbol{\beta}_{12}$ are the corresponding parameters to be estimated.

McCullagh and Nelder (1989) arrive at this joint bivariate model in two steps. First, transform the cell probabilities $\boldsymbol{\pi}$ to a vector of probabilities $\boldsymbol{\gamma}$, which also includes the univariate margins, given by

$$\boldsymbol{\gamma} = \boldsymbol{L}\boldsymbol{\pi} , \tag{10.12}$$

where $\boldsymbol{L}$ is a matrix of 0s and 1s of the form of a factorial design matrix. In the 2×2 case,

$$\boldsymbol{\gamma} = \begin{pmatrix} \pi_{1\bullet} \\ \pi_{2\bullet} \\ \pi_{\bullet 1} \\ \pi_{\bullet 2} \\ \pi_{11} \\ \pi_{12} \\ \pi_{21} \\ \pi_{22} \end{pmatrix} = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} \pi_{11} \\ \pi_{12} \\ \pi_{21} \\ \pi_{22} \end{pmatrix} . \tag{10.13}$$

There are of course only three linearly independent probabilities, because $\sum\sum \pi_{ij} = 1$. In the second step, the bivariate logistic model is formulated in terms of factorial contrasts on the elements of $\boldsymbol{\gamma}$, which express separate models for the two logits and the log odds. The model is expressed as

$$\boldsymbol{\eta} = \boldsymbol{C}\log\boldsymbol{\gamma} = \boldsymbol{C}\log(\boldsymbol{L}\boldsymbol{\pi}) , \tag{10.14}$$

where $\boldsymbol{C}$ is a matrix of contrasts. In the 2×2 case, the usual contrasts may be defined by

$$\boldsymbol{\eta} = \begin{pmatrix} \eta_1 \\ \eta_2 \\ \eta_{12} \end{pmatrix} = \begin{pmatrix} \text{logit } \pi_{1\bullet} \\ \text{logit } \pi_{\bullet 1} \\ \log(\theta_{12}) \end{pmatrix} = \begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & -1 & 1 \end{bmatrix} \begin{pmatrix} \log \pi_{1\bullet} \\ \log \pi_{2\bullet} \\ \log \pi_{\bullet 1} \\ \log \pi_{\bullet 2} \\ \log \pi_{11} \\ \log \pi_{12} \\ \log \pi_{21} \\ \log \pi_{22} \end{pmatrix}. \tag{10.15}$$

Thus, we are modeling the marginal log odds of each response, together with the log odds ratio $\log(\theta_{12})$ simultaneously.

Specific models are then formulated for the dependence of $\eta_1(\boldsymbol{x}), \eta_2(\boldsymbol{x})$, and $\eta_{12}(\boldsymbol{x})$ on some or all of the explanatory variables. For example, with one quantitative explanatory variable, x, the model

$$\begin{pmatrix} \eta_1 \\ \eta_2 \\ \eta_{12} \end{pmatrix} = \begin{pmatrix} \alpha_1 + \beta_1 x \\ \alpha_2 + \beta_2 x \\ \log(\theta) \end{pmatrix} \tag{10.16}$$

asserts that the log odds of each response changes linearly with x, while the odds ratio between the responses remains constant. In the general form given by McCullagh and Nelder (1989) the submodels in Eqn. (10.16) may each depend on the explanatory variables in different ways. For example, the logits could both depend quadratically on x, while an intercept-only model could be posited for the log odds ratio.

The second model is the ***bivariate loglinear model***, the special case obtained by taking $\boldsymbol{L} = \boldsymbol{I}$ in Eqn. (10.12) and Eqn. (10.14) so that $\boldsymbol{\gamma} = \boldsymbol{\pi}$. Then a loglinear model of the form

$$\boldsymbol{\eta}(\boldsymbol{x}) = \boldsymbol{C} \log \boldsymbol{\pi}$$

expresses contrasts among log probabilities as linear functions of the explanatory variables. For the 2×2 case, we take the contrasts $\boldsymbol{C}$ as shown below

$$\boldsymbol{\eta} = \begin{pmatrix} l_1 \\ l_2 \\ \eta_{12} \end{pmatrix} = \begin{bmatrix} 1 & 1 & -1 & -1 \\ 1 & -1 & 1 & -1 \\ 1 & -1 & 1 & -1 \end{bmatrix} \begin{pmatrix} \log \pi_{11} \\ \log \pi_{12} \\ \log \pi_{21} \\ \log \pi_{22} \end{pmatrix} \tag{10.17}$$

and models for the dependence of $l_1(\boldsymbol{x})$, $l_2(\boldsymbol{x})$, and $\eta_{12}(\boldsymbol{x})$ are expressed in the same way as in Eqn. (10.16). The estimates of the odds ratio, η_{12}, are the same under both models. The marginal functions are parameterized differently, however, but lead to similar predicted probabilities.

In R, bivariate logistic models of the form Eqn. (10.10) and Eqn. (10.11) can be fit using `vglm()` with the `binom2.or()` family in the VGAM package.[6] The fitting and graphing of these models is illustrated in the next example.

EXAMPLE 10.9: Breathlessness and wheeze in coal miners

In Example 4.12 we examined the association between the occurrence of two pulmonary conditions, breathlessness and wheeze, among coal miners classified by age (Ashford and Sowden, 1970). Figure 4.7 showed fourfold displays focused on the odds ratio for the co-occurrence of these symptoms, and Figure 4.8 plotted these odds ratios against age directly. Here, we consider models that examine the changes in prevalence of the two symptoms over age, together with the changes in their association.

[6] This package also provides for bivariate and trivariate loglinear models with `loglinb2()` and `loglinb2`.

10.4.1.1 Plotting bivariate response data

As a starting point and overview of what is necessary for bivariate response models, we calculate the empirical log odds for breathlessness and for wheeze, and the log odds ratio for their association in each 2×2 table. The log odds ratios are the same values plotted in Figure 4.8 (but the youngest age group was not included in the earlier analysis).

The `CoalMiners` data is a $2 \times 2 \times 9$ table. For convenience in this analysis (and for use with VGAM) we convert it to a 4×9 data frame, and relabel the columns to use the combinations of `("B", "b")` and `("W", "w")` to represent the conditions of breathlessness and wheeze, where the upper case letter indicates presence of the condition. A variable `age` is also created, using the midpoints of the age categories.

```
> data("CoalMiners", package = "vcd")
> coalminers <- data.frame(t(matrix(aperm(CoalMiners, c(2, 1, 3)),
+                                   4, 9)))
> colnames(coalminers) <- c("BW", "Bw", "bW", "bw")
> coalminers$age <- c(22, 27, 32, 37, 42, 47, 52, 57, 62)
> coalminers

   BW  Bw  bW   bw age
1   9   7  95 1841  22
2  23   9 105 1654  27
3  54  19 177 1863  32
4 121  48 257 2357  37
5 169  54 273 1778  42
6 269  88 324 1712  47
7 404 117 245 1324  52
8 406 152 225  967  57
9 372 106 132  526  62
```

With the data in this form, a simple function `blogits()` in vcdExtra calculates the logits and log odds ratios corresponding to Eqn. (10.10). The `add` argument accommodates cases where there are very small, or 0 frequencies in some cells, and it is common to add a small constant, such as 0.5, to each cell in calculating ***empirical logits***. This function is used to calculate the empirical logits and log odds as follows:

```
> logitsCM <- blogits(coalminers[, 1 : 4], add = 0.5)
> colnames(logitsCM)[1:2] <- c("logitB", "logitW")
> logitsCM

        logitB   logitW  logOR
 [1,] -4.73568 -2.86844 3.1956
 [2,] -3.97656 -2.55717 3.6583
 [3,] -3.31713 -2.09388 3.3790
 [4,] -2.73322 -1.84818 3.1327
 [5,] -2.21492 -1.42014 3.0069
 [6,] -1.73870 -1.10922 2.7770
 [7,] -1.10116 -0.79681 2.9217
 [8,] -0.75808 -0.57219 2.4368
 [9,] -0.31902 -0.22591 2.6318
```

We plot these as shown below, using `matplot()`, which is convenient for plotting multiple columns against a given horizontal variable, `age` here.[7] For ease of interpretation of the log odds, we also use right vertical axis showing the equivalent probabilities for breathlessness and wheeze.

[7] It is actually a small graphical misdemeanor to plot logits and odds ratios on the same vertical axis because they are not strictly commensurable. We plead guilty with the explanation that this graph shows what we want to see here and does not distort the data.

```
> col <- c("blue", "red", "black")
> pch <- c(15, 17, 16)
> age <- coalminers$age
>
> op <- par(mar = c(4, 4, 1, 4)+.2)
> matplot(age, logitsCM, type = "p",
+   col = col, pch = pch, cex = 1.2, cex.lab = 1.25,
+   xlab = "Age", ylab = "Log Odds or Odds Ratio")
> abline(lm(logitsCM[,1] ~ age), col = col[1], lwd = 2)
> abline(lm(logitsCM[,2] ~ age), col = col[2], lwd = 2)
> abline(lm(logitsCM[,3] ~ age), col = col[3], lwd = 2)
>
> # right probability axis
> probs <- c(.01, .05, .10, .25, .5)
> axis(4, at = qlogis(probs), labels = probs)
> mtext("Probability", side = 4, cex = 1.2, at = -2, line = 2.5)
> # curve labels
> text(age[2], logitsCM[2, 1] + .5, "Breathlessness",
+      col = col[1], pos = NULL, cex = 1.2)
> text(age[2], logitsCM[2, 2] + .5, "Wheeze",
+      col = col[2], pos = NULL, cex = 1.2)
> text(age[2], logitsCM[2, 3] - .5, "log OR\n(B|W)/(B|w)",
+      col = col[3], pos = 1, cex = 1.2)
> par(op)
```

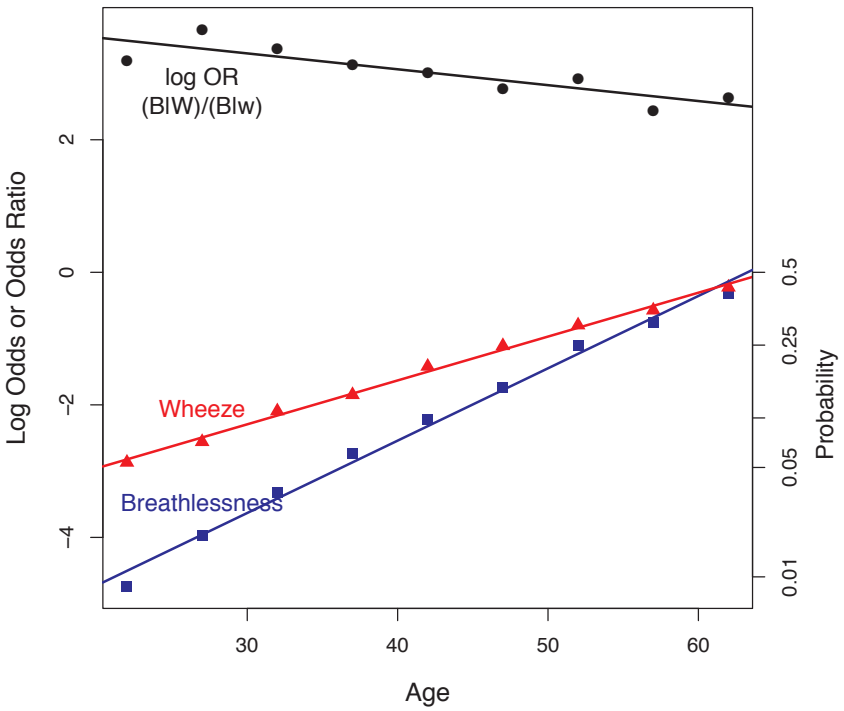


Figure 10.16: Empirical logits and log odds ratio for breathlessness and wheeze in the CoalMiners data. The lines show separate linear regressions for each function. The right vertical axis shows equivalent probabilities for the logits.

In Figure 10.16 we see that both symptoms, while quite rare among young miners, increase steadily with age (or years working in the mine). By age 60, the probability is nearly 0.5 of having either condition. There is a hint of curvilinearity, particularly in the logit for breathlessness. The

decline in the odds ratio with age may reflect selection, as miners who had retired for health or other reasons were excluded from the study.

10.4.1.2 Fitting `glm` models

Next, we illustrate what can easily be achieved using the standard `glm()` approach for loglinear models and why the bivariate models we described are more useful in this situation. `glm()` requires a data frame as input, so first reshape *`CoalMiners`* to a frequency data frame. For convenience, we simplify the variable names to `B` and `W`.

```
> CM <- as.data.frame(CoalMiners)
> colnames(CM)[1:2] <- c("B", "W")
> head(CM)

    B   W   Age Freq
1   B   W 20-24    9
2 NoB   W 20-24   95
3   B NoW 20-24    7
4 NoB NoW 20-24 1841
5   B   W 25-29   23
6 NoB   W 25-29  105
```

As a point of comparison, we fit the mutual independence model, [B][W][Age], and the baseline model for associated responses, [BW][Age], which asserts that the association between `B` and W is independent of `Age`.

```
> cm.glm0 <- glm(Freq ~ B + W + Age, data = CM, family = poisson)
> cm.glm1 <- glm(Freq ~ B * W + Age, data = CM, family = poisson)
> LRstats(cm.glm0, cm.glm1)

Likelihood summary table:
         AIC  BIC LR Chisq Df Pr(>Chisq)
cm.glm0 7217 7234     6939 25     <2e-16 ***
cm.glm1 2981 3000     2702 24     <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The baseline model `cm.glm1` fits very badly. We can see the pattern of the residual association in a mosaic display for this model shown in Figure 10.17. The formula argument here specifies the order of the variables in the mosaic.

```
> vnames <- list(set_varnames = c(B = "Breathlessness", W = "Wheeze"))
> lnames <- list(B=c("B", "b"), W = c("W", "w"))
> mosaic(cm.glm1, ~ Age + B + W,
+        labeling_args = vnames, set_labels = lnames)
```

As structured here, it is easy to see the increase in the prevalence of breathlessness and wheeze with age and the changing pattern of their association with age.

From Figure 10.16 and Figure 10.17, it is apparent that both breathlessness and wheeze increase with age, so we can model this by adding terms [B Age][W Age] to the baseline model. This is the no-three-way interaction model, which could also be specified as `Freq ~ (B + W + Age)^2`.

```
> cm.glm2 <- glm(Freq ~ B * W + (B + W) * Age, data = CM, family = poisson)
> LRstats(cm.glm1, cm.glm2)

Likelihood summary table:
         AIC  BIC LR Chisq Df Pr(>Chisq)
```

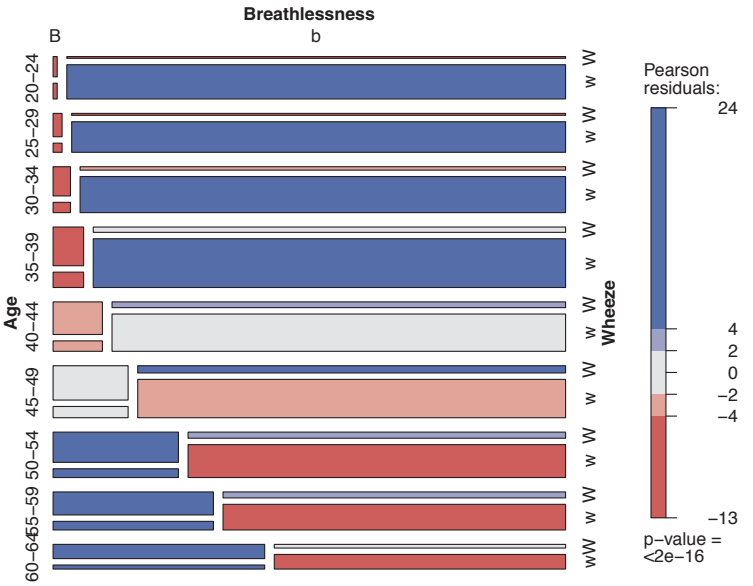


Figure 10.17: Mosaic display for the baseline model, [BW][Age], fit to the CoalMiners data.

```
cm.glm1 2981 3000     2702 24     <2e-16 ***
cm.glm2  338  383       27  8      8e-04 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The improvement in fit is substantial, and all terms are highly significant, yet, the residual $G^2(8)$ indicates there is still lack of fit.

```
> library(car)
> Anova(cm.glm2)

Analysis of Deviance Table (Type II tests)

Response: Freq
      LR Chisq Df Pr(>Chisq)
B         11026  1     <2e-16 ***
W          7038  1     <2e-16 ***
Age         887  8     <2e-16 ***
B:W        3025  1     <2e-16 ***
B:Age      1130  8     <2e-16 ***
W:Age       333  8     <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

One way to improve the model using the `glm()` framework is to make use of `Age` as a quantitative variable and add a term to allow the odds ratio for the [BW] association to vary linearly with age. Here, we construct the variable `age` using the midpoints of the Age intervals.

```
> CM$age <- rep(seq(22, 62, 5), each = 4)
```

In the `glm()` approach, the odds ratio cannot be modeled directly, but we can use the following trick: For each 2×2 subtable, the odds ratio can be parameterized in terms of the frequency in

any one cell, say, n_{11k}, given that the marginal total n_{++k} is included in the model. We do this by adding a new interaction variable, `ageOR`, having the value of `age` for the $(1, 1, k)$ cells and 0 otherwise.

```
> CM$ageOR <- (CM$B == "B") * (CM$W == "W") * CM$age
> cm.glm3 <- update(cm.glm2, . ~ . + ageOR)
> LRstats(cm.glm0, cm.glm1, cm.glm2, cm.glm3)

Likelihood summary table:
         AIC  BIC LR Chisq Df Pr(>Chisq)
cm.glm0 7217 7234     6939 25     <2e-16 ***
cm.glm1 2981 3000     2702 24     <2e-16 ***
cm.glm2  338  383       27  8     0.0008 ***
cm.glm3  320  366        7  7     0.4498
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The model `cm.glm3`, with one more parameter, now fits reasonably well, having residual $G^2(7) = 6.80$. The likelihood ratio test of model `cm.glm3` against `cm.glm2`, which assumes equal odds ratios over age, can be regarded as a test of the hypothesis of homogeneity of odds ratios, against the alternative that the [BW] association changes linearly with age. The `glm()` models fit in this example are summarized above. As usual, `anova()` can be used to compare competing nested models.

```
> anova(cm.glm2, cm.glm3, test = "Chisq")

Analysis of Deviance Table

Model 1: Freq ~ B * W + (B + W) * Age
Model 2: Freq ~ B + W + Age + ageOR + B:W + B:Age + W:Age
  Resid. Df Resid. Dev Df Deviance Pr(>Chi)
1         8       26.7
2         7        6.8  1     19.9  8.2e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

This analysis, while useful, also shows the limitations of the `glm()` approach: (a) It doesn't easily allow us to represent and test the substantively interesting hypotheses regarding *how* the prevalence of the binary responses, `B` and `W`, vary with `Age`, such as seen in Figure 10.16. (b) It doesn't represent the odds ratio for the [BW] association directly, but only through the coding trick we used here. Thus, it is difficult to interpret the coefficient for `ageOR` = -0.02613 in a substantively meaningful way, except that is shows that the odds ratio is decreasing.[8]

10.4.1.3 Fitting `vglm` models

The `vglm()` function in the VGAM package provides a very general implementation of these and other models for discrete multivariate responses. The family function, `binom2.or()` for binary logistic models, allows some or all of the logits or odds ratio submodels to be constrained to be intercept-only (e.g., as in Eqn. (10.16)) and the two marginal distributions can be constrained to be equal.

Quantitative predictors (such as age, here), can be modeled linearly or nonlinearly, using `poly()` for a parametric fit, or smooth regression splines, as provided by the functions `ns()`, `bs()`, and others in model formulas. In this illustration, we fit bivariate linear and quadratic models in age.

`vglm()` takes its input data in the wide form we called `coalminers` at the beginning of this

[8] Actually, the interpretability of the coefficient for the log odds ratio can be enhanced here by centering age, and representing its units in steps of 5 years, as we do below.

example. We could use the 9-level factor, `Age`, as we did with `glm()`, but we plan to use `age` as a numeric variable in all three submodels. The coefficients in these models will be more easily interpreted if we center age and express it as `agec` in units of five years, as shown below.

```
> coalminers <- transform(coalminers, agec = (age - 42) / 5)
> coalminers$Age <- dimnames(CoalMiners)[[3]]
> coalminers

   BW  Bw  bW   bw age agec   Age
1   9   7  95 1841  22   -4 20-24
2  23   9 105 1654  27   -3 25-29
3  54  19 177 1863  32   -2 30-34
4 121  48 257 2357  37   -1 35-39
5 169  54 273 1778  42    0 40-44
6 269  88 324 1712  47    1 45-49
7 404 117 245 1324  52    2 50-54
8 406 152 225  967  57    3 55-59
9 372 106 132  526  62    4 60-64
```

`vglm()` takes the 2×2 response frequencies as a 4-column matrix on the left-hand side of the model formula. However, denoting the responses of failure and success by 0 and 1, respectively, it takes these in the order $y_{00}, y_{01}, y_{10}, y_{11}$. We specify the order below so that the logits are calculated for the occurrence of breathlessness or wheeze, rather than their absence.

```
> library(VGAM)
> #                          00  01  10  11
> cm.vglm1 <- vglm(cbind(bw, bW, Bw, BW) ~ agec,
+                  binom2.or(zero = NULL), data = coalminers)
> cm.vglm1

Call:
vglm(formula = cbind(bw, bW, Bw, BW) ~ agec, family = binom2.or(zero = NULL),
    data = coalminers)

Coefficients:
(Intercept):1 (Intercept):2 (Intercept):3        agec:1
     -2.26247      -1.48776       3.02191       0.51451
       agec:2        agec:3
      0.32545      -0.13136

Degrees of Freedom: 27 Total; 21 Residual
Residual deviance: 30.394
Log-likelihood: -100.53
```

In this call, the argument `zero = NULL` indicates that none of the linear predictors, $\eta_1, \eta_2, \eta_{12}$, are modeled as constants.[9]

At this writing, there is no `anova()` method for the "vgam" objects produced by `vglm()`, but we can test the residual deviance of the model (against the saturated model) as follows, showing that this model has an acceptable fit.

```
> (G2 <- deviance(cm.vglm1))

[1] 30.394

> # test residual deviance
> 1-pchisq(deviance(cm.vglm1), cm.vglm1@df.residual)

[1] 0.084355
```

[9] The default, `zero=3` gives the model shown in Eqn. (10.16), with the odds ratio constant.

The estimated coefficients in this model are usefully shown as below, using the argument `matrix=TRUE` in `coef()`.

Using `exp()` on the result gives values of odds that can be easily interpreted:

```
> coef(cm.vglm1, matrix = TRUE)

            logit(mu1) logit(mu2) loge(oratio)
(Intercept)   -2.26247   -1.48776      3.02191
agec           0.51451    0.32545     -0.13136

> exp(coef(cm.vglm1, matrix = TRUE))

            logit(mu1) logit(mu2) loge(oratio)
(Intercept)    0.10409    0.22588      20.5304
agec           1.67282    1.38465       0.8769
```

Thus, the odds of a miner showing breathlessness are multiplied by 1.67, a 67% increase, for each 5 years' increase in age; similarly, the odds of wheeze are multiplied by 1.38, a 38% increase. The odds ratio for the association between the two symptoms are multiplied by 0.88, a 12% decrease over each 5-year interval.

The VGAM package has no special plot methods for "vglm" objects, but it is not hard to construct these using the methods we showed earlier in this example. First, we can obtain the fitted probabilities for the 4 response combinations using `fitted()`, and the corresponding observed probabilities using `depvar()`.

```
> age <- coalminers$age
> P <- fitted(cm.vglm1)
> colnames(P) <- c("bw", "bW", "Bw", "BW")
> head(P)

        bw        bW         Bw         BW
1 0.93747 0.049409 0.0046356 0.0084831
2 0.91461 0.063636 0.0069757 0.0147776
3 0.88411 0.080029 0.0104965 0.0253679
4 0.84394 0.097484 0.0158138 0.0427671
5 0.79188 0.113839 0.0238598 0.0704196
6 0.72578 0.125910 0.0359684 0.1123366

> Y <- depvar(cm.vglm1)
```

In the left panel of Figure 10.18, we plot the fitted probabilities in the matrix `P` using `matplot()` and the observed probabilities in `Y` using `matpoints()`.

```
> col <- c("red", "blue", "red", "blue")
> pch <- c(1, 2, 16, 17)
>
> op <- par(mar = c(5, 4, 1, 1) + .1)
> matplot(age, P, type = "l",
+   col = col,
+   lwd = 2, cex = 1.2, cex.lab = 1.2,
+   xlab = "Age", ylab = "Probability",
+   xlim = c(20,65))
> matpoints(age, Y, pch = pch, cex = 1.2, col = col)
> # legend
> text(64, P[9,]+ c(0,.01, -.01, 0), labels = colnames(P), col = col, cex = 1.2)
> text(20, P[1,]+ c(0,.01, -.01, .01), labels = colnames(P), col = col, cex = 1.2)
> par(op)
```

The right panel of Figure 10.18 shows these on the log odds scale, produced using the same code as above, applied to the probabilities transformed using `qlogis()`, the quantile function for the logistic distribution.

```
> lP <- qlogis(P)
> lY <- qlogis(Y)
```

In Figure 10.16 we plotted the empirical logits and log odds using the function `blogits()` to transform frequencies to these values. An essentially identical plot can be produced by transforming the fitted and observed probabilities, as calculated below.

```
> # blogits, but for B and W
> logitsP <- blogits(P[, 4 : 1])
> logitsY <- blogits(Y[, 4 : 1])
```

To test for nonlinearity in the prevalence of the symptoms or their odds ratio with age, we can fit a similar model using `poly()` or a smoothing spline, such as `ns()`. We illustrate this here using a bivariate model allowing quadratic effects of age on all three components.

```
> cm.vglm2 <- vglm(cbind(bw, bW, Bw, BW) ~ poly(agec, 2),
+                  binom2.or(zero = NULL), data = coalminers)
```

This model has a residual $G^2 = 16.963$ with 18 df. Compared to the linear model `cm.vglm1`, this represents a significant improvement in goodness of fit.

```
> (LR <- deviance(cm.vglm1) - deviance(cm.vglm2))

[1] 13.43

> 1 - pchisq(LR, cm.vglm1@df.residual - cm.vglm2@df.residual)

[1] 0.0037925
```

A plot of the fitted logits and log odds ratios under this model is shown in Figure 10.19. You can interpret this plot as showing that the statistical evidence for the quadratic model indicates some slight tendency for the prevalence of breathlessness and wheeze to level off slightly with age, particularly the former.

△

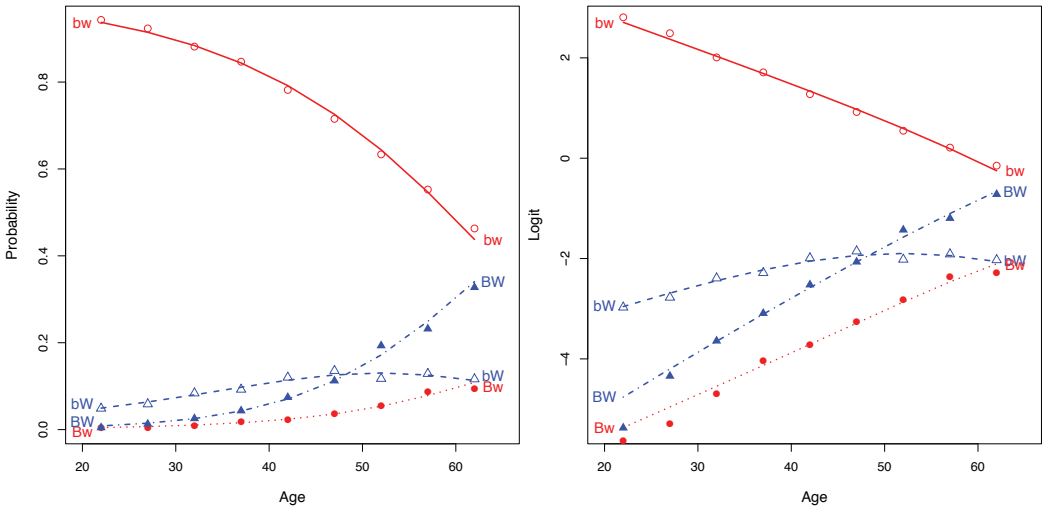


Figure 10.18: Observed and fitted values for the combinations of breathlessness and wheeze in the binary logistic regression model `cm.vglm1`. Left: probabilities; right: on the log odds scale.

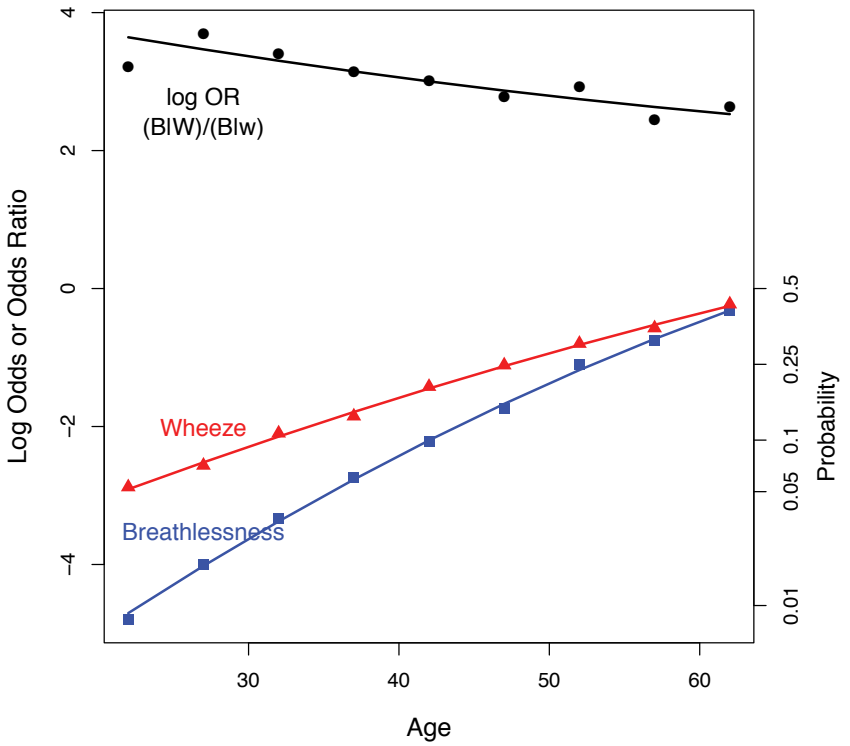


Figure 10.19: Observed (points) and fitted (lines) logits and log odds ratios for the quadratic binary logistic regression model `cm.vglm2`.

10.4.2 More complex models

When there is more than one explanatory variable and several responses, the methods described above using `glm()` and `vglm()` still apply. However, it is useful to begin with a more thorough visual examination of the relations within and between these sets. Some useful graphical displays include:

- mosaic displays showing the marginal relations among the response variables and of the explanatory variables, each collapsed over the other set;
- conditional mosaics or fourfold displays of the associations among the responses, stratified by one or more of the explanatory variables;
- plots of empirical logits and log odds ratios, as in Figure 10.16 or model-based plots, such as Figure 10.19, showing a model-smoothed summary.

These displays can, and should, inform our search for an adequate descriptive or explanatory model. Some of these ideas are illustrated in the following example.

EXAMPLE 10.10: Toxaemic symptoms in pregnancy

Brown et al. (1983) gave the data used here on two signs of *toxaemia*, an abnormal condition during pregnancy characterized by high blood pressure (hypertension) and high levels of protein in the urine. If untreated, both the mother and baby are at risk of complications or death. The data frame `Toxaemia` in vcdExtra represents 13,384 expectant mothers in Bradford, England in

their first pregnancy, who were also classified according to social class and the number of cigarettes smoked per day.

There are thus two response variables, and two explanatory variables in this data set in frequency form. For convenience, we also convert it to a $2 \times 2 \times 5 \times 3$ table.

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

'data.frame': 60 obs. of  5 variables:
 $ class: Factor w/ 5 levels "1","2","3","4",..: 1 1 1 1 1 1 1 1 1 1 ...
 $ smoke: Factor w/ 3 levels "0","1-19","20+": 1 1 1 1 2 2 2 2 3 3 ...
 $ hyper: Factor w/ 2 levels "Low","High": 2 2 1 1 2 2 1 1 2 2 ...
 $ urea : Factor w/ 2 levels "Low","High": 2 1 2 1 2 1 2 1 2 1 ...
 $ Freq : int  28 82 21 286 5 24 5 71 1 3 ...

> tox.tab <- xtabs(Freq ~ class + smoke + hyper + urea, Toxaemia)
> ftable(tox.tab, row.vars = 1)

      smoke    0                1-19                20+
      hyper  Low       High      Low       High      Low       High
      urea   Low High  Low High  Low High  Low High  Low High  Low High
class
1            286   21   82   28   71    5   24    5   13    0    3    1
2            785   34  266   50  284   17   92   13   34    3   15    0
3           3160  164 1101  278 2300  142  492  120  383   32   92   16
4            656   52  213   63  649   46  129   35  163   12   40    7
5            245   23   78   20  321   34   74   22   65    4   14    7
```

The questions of main interest are how the occurrence of each symptom varies with social class and smoking, and how the association between these symptoms varies. It is useful, however, to examine first the marginal relationship between the two responses, and between the two predictors. The calls to `mosaic()` below produce the two panels in Figure 10.20.

```
> mosaic(~ smoke + class, data = tox.tab, shade = TRUE,
+         main = "Predictors", legend = FALSE)
> mosaic(~ hyper + urea, data = tox.tab, shade = TRUE,
+         main = "Responses", legend = FALSE)
```

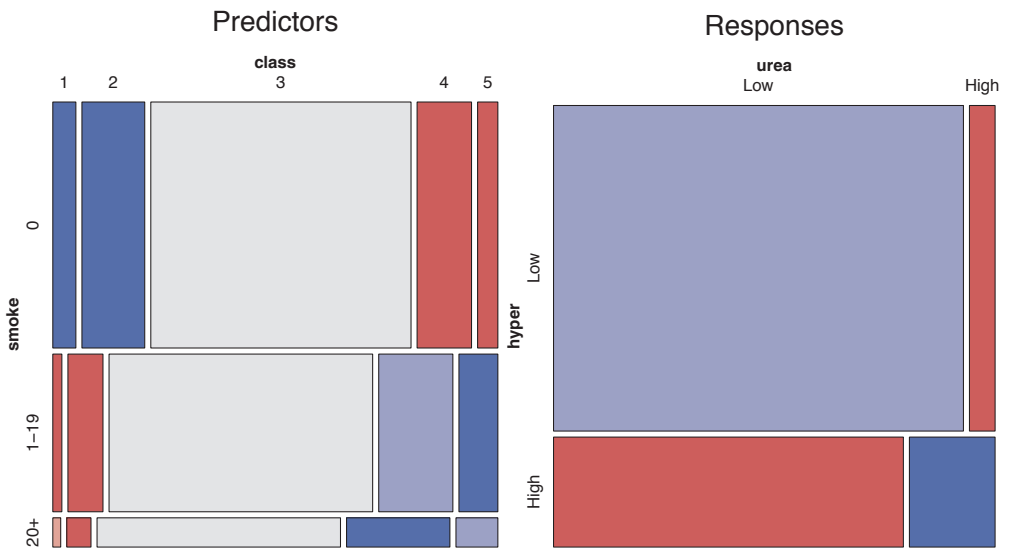


Figure 10.20: Mosaic displays for Toxaemia data: Predictor and response associations.

We see in Figure 10.20 that the majority of the mothers are in the third social class, and that

smoking is negatively related to social class, with the highest levels of smoking in classes 4 and 5. (Social class 1 is the highest in status here.) More than 50% are non-smokers. Within the responses, the great majority of women exhibit neither symptom, but showing one symptom makes it much more likely to show the other. Marginally, hypertension is somewhat more prevalent than protein urea.

We next examine how the association between responses varies with social class and with smoking. Figure 10.21 shows a collection of conditional mosaic plots using `cotabplot()` of the association between hypertension and urea, for each level of smoking, collapsed over social class.

```
> cotabplot(~ hyper + urea | smoke, tox.tab, shade = TRUE,
+             legend = FALSE, layout = c(1, 3))
```

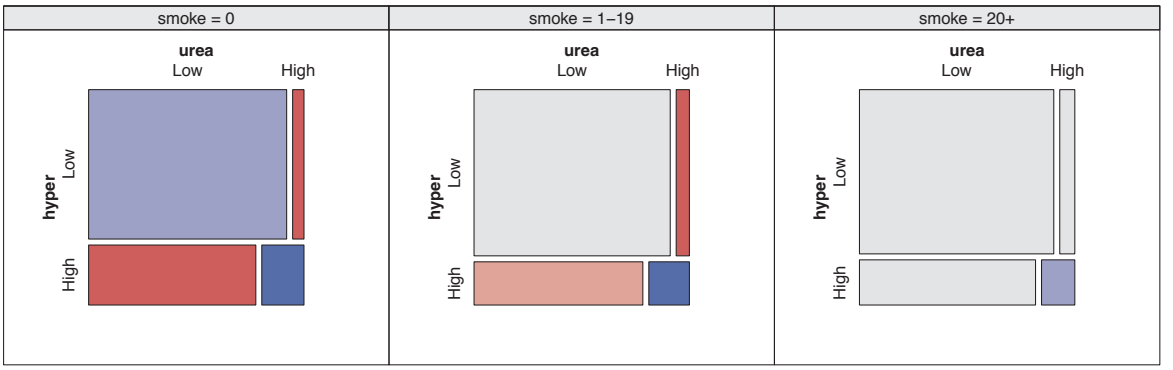


Figure 10.21: Toxaemia data: Response association conditioned on smoking level.

Figure 10.22 is similar, but stratified by social class. The marginal frequencies of the conditioning variable is not represented in these plots. (For example, as can be seen in Figure 10.20, the greatest number of women are in class 3.)

```
> cotabplot(~ hyper + urea | class, tox.tab, shade = TRUE,
+             legend = FALSE, layout = c(1, 5))
```

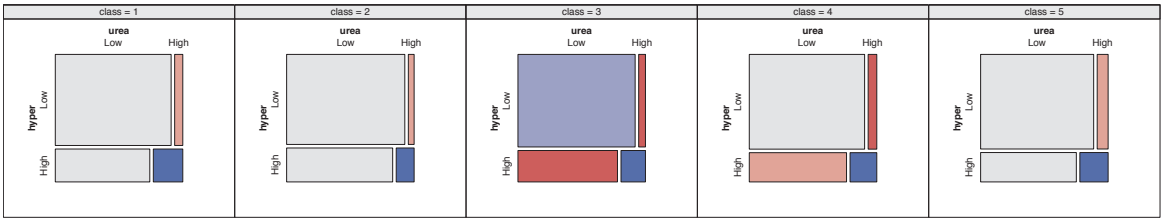


Figure 10.22: Toxaemia data: Response association conditioned on social class.

Ignoring social class, the association between hypertension and protein urea decreases with smoking. Ignoring smoking, the association is greatest in social class 3. However, these displays don't show directly how the two symptoms are associated in the combinations of social class and smoking. The fourfold display in Figure 10.23 does that.

```
> fourfold(aperm(tox.tab), fontsize = 16)
```

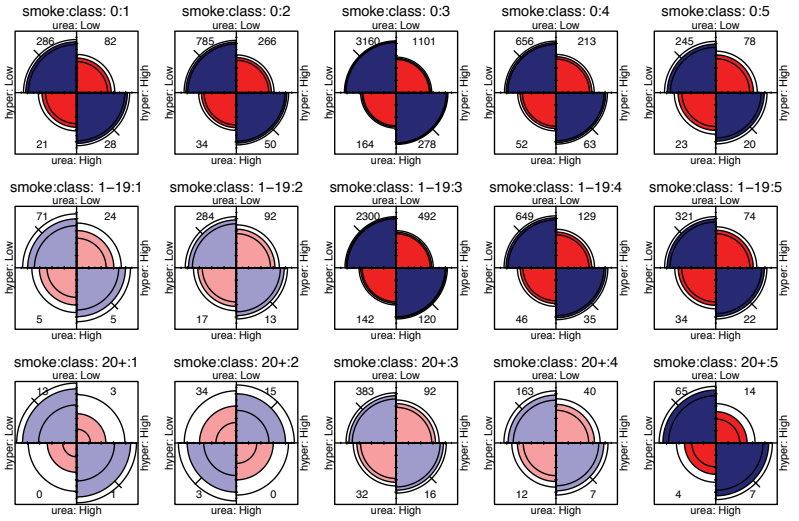


Figure 10.23: Fourfold display for Toxaemia data. Smoking levels vary in the rows and social class in the columns.

It can be seen in Figure 10.23 that the odds ratio appears to increase with both smoking and social class number and these two symptoms are positively associated in nearly all cases. In only two cases the odds ratio is not significantly different from 1: mothers in classes 1 and 2, who smoke more than 20 cigarettes a day, but the frequency in this cell is quite small.

```
> margin.table(tox.tab, 2 : 1)

      class
smoke     1    2    3    4    5
  0     417 1135 4703  984  366
  1-19  105  406 3054  859  451
  20+    17   52  523  222   90
```

From these plots, it is useful to examine the association between hypertension and urea more directly, by calculating and plotting the odds ratios. For a $2 \times 2 \times K \times L \times \cdots$ table, the function `loddsratio()` in vcd calculates these for each 2×2 subtable, and returns an array of dimension $K \times L \times \cdots$, together with similar array of standard errors.

```
> (LOR <- loddsratio(urea ~ hyper | smoke + class, data = tox.tab))

log odds ratios for urea and hyper by smoke, class

      class
smoke       1        2       3       4       5
  0    1.5268  1.46196 1.58056 1.31351 1.0036
  1-19 1.0710  0.86401 1.37370 1.34260 1.0348
  20+  2.4485 -1.14579 0.74425 0.88469 2.0187
```

The `plot()` method for the resulting "logoddsratio" object treats the conditioning variables in the formula argument as strata, and plots the log odds ratios for the first such variable on the horizontal axis with curves for the subsequent strata variables. The lines below produce Figure 10.24.

```
> plot(t(LOR), confidence = FALSE, legend_pos = "bottomright",
+      xlab = "Social class of mother")
```

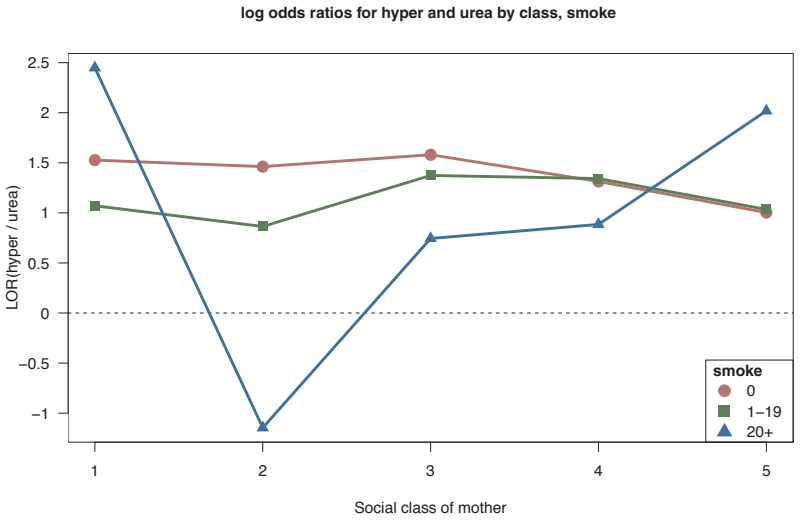


Figure 10.24: Log odds ratios for protein urea given hypertension, by social class and level of maternal smoking.

The association between the response symptoms, shown in Figure 10.24, is clearer, once we take the variation in sample sizes into account. Except for the heavy smokers, particularly in social classes 1 and 2, the log odds ratio appears to range only between 1–1.5, meaning that, given one symptom, the odds of also having the other range between $\exp(1) = 2.72$ and $\exp(1.5) = 4.48$.

This initial overview of the data is completed by calculating and plotting the log odds for each symptom within each class–smoke population. This could be done in the same way as in Example 10.9 (except that there are now two explanatory factors). The steps used there were: (a) Reshape the $2 \times 2 \times K \cdots$ table to a matrix with four columns corresponding to the binary response combinations. (b) Calculate the logits (and log odds ratio) using `blogits()`.

Here, it is more useful to make separate plots for each of the logits, and we illustrate a more general approach that applies to two or more binary responses, with two or more predictor variables. The essential idea is to fit a separate logit model for each response separately, using the *highest-order interaction* of all predictors (the saturated model). The fitted logits in these models then match those in the data.

```
> tox.hyper <- glm(hyper == "High" ~ class * smoke, weights = Freq,
+                  data = Toxaemia, family = binomial)
> tox.urea <- glm(urea == "High" ~ class * smoke, weights = Freq,
+                 data = Toxaemia, family = binomial)
```

It is then simple to plot these results using the effects package as shown in Figure 10.25. Each plot shows the logit for the response measure against class, with separate curves for the levels of smoking.[10]

[10] As is usual for effect plots of binary response `glm()` models, the vertical axis is plotted on the scale of log odds, but labeled in terms of probabilities.

```
> library(effects)
>
> plot(allEffects(tox.hyper),
+    ylab = "Probability (hypertension)",
+    xlab = "Social class of mother",
+    main = "Hypertension: class*smoke effect plot",
+    colors = c("blue", "black", "red"),
+    lwd=3,  multiline = TRUE,
+    key.args = list(x = 0.05, y = 0.2, cex = 1.2, columns = 1)
+    )
>
> plot(allEffects(tox.urea),
+    ylab = "Probability (Urea)",
+    xlab = "Social class of mother",
+    main = "Urea: class*smoke effect plot",
+    colors = c("blue", "black", "red"),
+    lwd=3,  multiline = TRUE,
+    key.args = list(x = 0.65, y = 0.2, cex = 1.2, columns = 1)
+    )
```

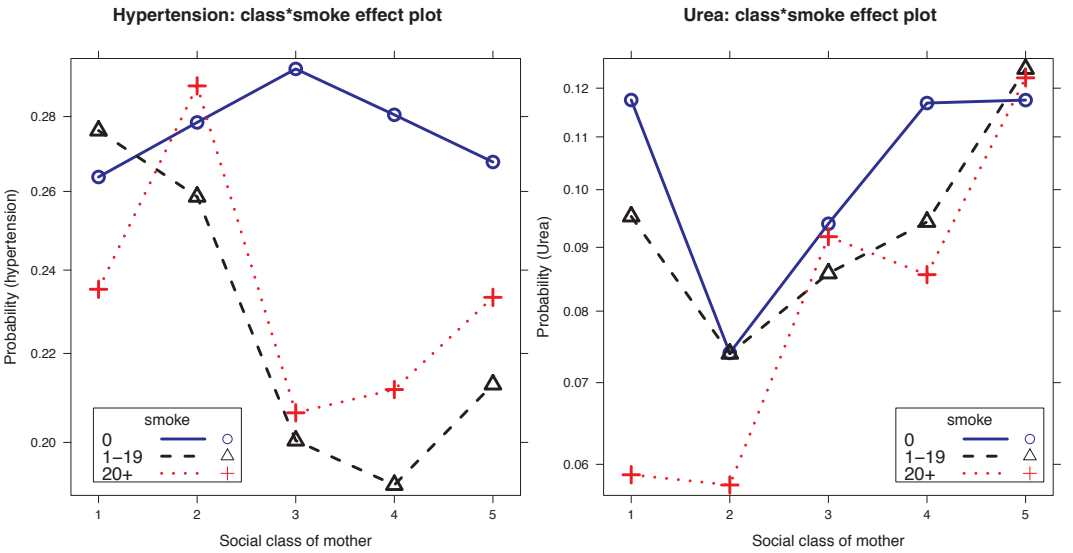


Figure 10.25: Effect plots for hypertension and urea, by social class of mother and smoking.

From Figure 10.25, it can be seen that the prevalence of these symptoms has a possibly complex relation to social class and smoking. However, the mosaic for these predictors in Figure 10.20 has shown us that several of the class–smoking categories are quite small (particularly heavy smokers in Classes 1 and 2), so the response effects for these classes will be poorly estimated. Taking this into account, we suspect that protein urea varies with social class, but not with smoking, while the prevalence of hypertension may truly vary with neither, just one, or both of these predictors.

10.4.2.1 Fitting models

The plots shown so far in this example are all essentially *data-based*, in that they use the observed frequencies or transformations of them and don't allow for a simpler view, based on a reasonable model. That is, abbreviating the table variables by their initial letters, the plots in Figure 10.24 and Figure 10.25 are plots of the saturated model, [CSHU], which fits perfectly, but with the data transformed for each 2×2 subtable to the log odds ratio and the two log odds for `hyper` and `urea`.

The bivariate logistic model fit by `vglm()` still applies when there are two or more predictors; however, like other multivariate response models, it doesn't easily allow the logits to depend on *different* predictor terms. To illustrate this, we first transform the `Toxaemia` to a 15×4 data frame in the form required by `vglm()`.

```
> tox.tab <- xtabs(Freq~class + smoke + hyper + urea, Toxaemia)
> toxaemia <- t(matrix(aperm(tox.tab), 4, 15))
> colnames(toxaemia) <- c("hu", "hU", "Hu", "HU")
> rowlabs <- expand.grid(smoke = c("0", "1-19", "20+"),
+                        class = factor(1:5))
> toxaemia <- cbind(toxaemia, rowlabs)
> head(toxaemia)

   hu hU  Hu HU smoke class
1 286 21  82 28     0     1
2  71  5  24  5  1-19     1
3  13  0   3  1   20+     1
4 785 34 266 50     0     2
5 284 17  92 13  1-19     2
6  34  3  15  0   20+     2
```

In the model specification for `vglm()`, the `zero` argument in `binom.or()` allows any one or more of the two log odds and log odds ratio to be fit as a constant (intercept-only) in Eqn. (10.11). However, in that equation, the predictors $\boldsymbol{x}_1$, $\boldsymbol{x}_2$, $\boldsymbol{x}_{12}$, must be the *same* in all three submodels. For example, the model `tox.vglm1` below uses main effects of `class` and `smoke` in both models for the logits, and `zero=3` for a constant log odds ratio.

```
> tox.vglm1 <- vglm(cbind(hu, hU, Hu, Hu) ~ class + smoke,
+                   binom2.or(zero = 3), data = toxaemia)
> coef(tox.vglm1, matrix=TRUE)

             logit(mu1) logit(mu2) loge(oratio)
(Intercept) -0.50853648 -1.2214518       2.7808
class2       0.18156457  0.0382046       0.0000
class3       0.06332765 -0.0087552       0.0000
class4      -0.02227055 -0.0031541       0.0000
class5      -0.00077172  0.0821863       0.0000
smoke1-19   -0.41298650 -0.2198673       0.0000
smoke20+    -0.30562472 -0.1245019       0.0000
```

Instead, when there are no quantitative predictors, and when the odds ratio is relatively constant (as here) it is easier to fit ordinary loglinear models than to use the bivariate logit formulation of the previous example. These allow the responses H and U to depend on the class-smoking combinations separately, by including the terms $[CSH]$ or $[CSU]$, respectively.

The minimal, null model, $[CS][H][U]$, fits the marginal association of the numbers in each class–smoking category, but asserts that the responses, H and U, are independent, which we have already seen is contradicted by the data. We take $[CS][HU]$ as the baseline model (Model 1), asserting no relation between response and predictor variables, but associations within each set are allowed. These models are fit as shown below.

```
> # null model
> tox.glm0 <- glm(Freq ~ class*smoke + hyper + urea,
+                 data = Toxaemia, family = poisson)
> # baseline model: no association between predictors and responses
> tox.glm1 <- glm(Freq ~ class*smoke + hyper*urea,
+                 data = Toxaemia, family = poisson)
```

We proceed to fit a collection of other models, adding terms to allow more associations between the responses and predictors. Summary measures of goodness of fit and parsimony are shown in Table 10.1.

Table 10.1: Loglinear models, `tox.glm*`, fit to the Toxaemia data

Model	Terms	df	G^2	p-value	G^2/df	AIC	BIC	R^2
0	CS H U	43	672.85	0.0000	15.65	586.85	264.27	.
1	CS HU	42	179.03	0.0000	4.26	95.03	-220.04	0.000
2	CS HU SH CU	36	46.12	0.1203	1.28	-25.88	-295.94	0.742
3	CS CH CU HU SH CU	30	40.47	0.0960	1.35	-19.53	-244.58	0.774
4	CSH CU HU	24	26.00	0.3529	1.08	-22.00	-202.04	0.855
5	CSH CU SU HU	22	25.84	0.2588	1.17	-18.16	-183.20	0.856
6	CSH CSU HU	14	22.29	0.0729	1.59	-5.71	-110.74	0.875
7	CSH CSU SHU	12	15.65	0.2079	1.30	-8.35	-98.37	0.913
8	CSH CSU CHU SHU	8	12.68	0.1233	1.59	-3.32	-63.33	0.929
9	CSHU	0	0.00	0	0	0.00	0.00	1.000

```
> tox.glm2 <- update(tox.glm1, . ~ . + smoke*hyper + class*urea)
>
> tox.glm3 <- glm(Freq ~ (class + smoke + hyper + urea)^2,
+                 data=Toxaemia, family=poisson)
>
> tox.glm4 <- glm(Freq ~ class*smoke*hyper + hyper*urea + class*urea,
+                 data=Toxaemia, family=poisson)
>
> tox.glm5 <- update(tox.glm4, . ~ . + smoke*urea)
>
> tox.glm6 <- update(tox.glm4, . ~ . + class*smoke*urea)
>
> tox.glm7 <- update(tox.glm6, . ~ . + smoke*hyper*urea)
>
> tox.glm8 <- glm(Freq ~ (class + smoke + hyper + urea)^3,
+                 data = Toxaemia, family = poisson)
>
> tox.glm9 <- glm(Freq ~ (class + smoke + hyper + urea)^4,
+                 data = Toxaemia, family = poisson)
```

Model 2 adds the simple dependence of hypertension on smoking ($[SH]$) and that of urea on class ($[CU]$). Model 3 includes all two-way terms. In Model 4, hypertension is allowed to depend on both class and smoking jointly ($[CSH]$). In Model 5 an additional dependence of urea on smoking ($[SU]$) is included, while in Model 6 urea depends on class and smoking jointly ($[CSU]$).

None of these models contain three-way terms involving both H and U, so these models assume that the log odds ratio for hypertension given urea is constant over the explanatory variables. Recalling the conditional mosaics (Figure 10.21 and Figure 10.22), Models 7 and 8 add terms that allow the odds ratio to vary, first with smoking ($[SHU]$), then with class ($[CHU]$) as well. Finally, Model 9 is the saturated model, that fits perfectly.

How do we choose among these models? Model 2 is the smallest model whose deviance is non-significant. Models 4 and 5 both have a smaller ratio of G^2/df. For comparing nested models, we can also examine the change in deviance as terms are added (or dropped). Thus, going from Model 2 to Model 3 decreases the deviance by 5.65 on 6 df, while the step from Model 3 to Model 4 gives a decrease of 14.47, also on 6 df. These tests can be performed using `lrtest()` in the lmtest package, shown below for models `tox.glm1`–`tox.glm5`.

```
> library(lmtest)
> lmtest::lrtest(tox.glm1, tox.glm2, tox.glm3, tox.glm4, tox.glm5)

Likelihood ratio test

Model 1: Freq ~ class * smoke + hyper * urea
Model 2: Freq ~ class + smoke + hyper + urea + class:smoke + hyper:urea +
    smoke:hyper + class:urea
```

```
Model 3: Freq ~ (class + smoke + hyper + urea)^2
Model 4: Freq ~ class * smoke * hyper + hyper * urea + class * urea
Model 5: Freq ~ class + smoke + hyper + urea + class:smoke + class:hyper +
    smoke:hyper + hyper:urea + class:urea + smoke:urea + class:smoke:hyper
  #Df LogLik Df  Chisq Pr(>Chisq)
1  18   -260
2  24   -194  6 132.91     <2e-16 ***
3  30   -191  6   5.65      0.464
4  36   -184  6  14.47      0.025 *
5  38   -184  2   0.17      0.920
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The AIC and BIC statistics, balancing parsimony and goodness-of-fit, have their minimum value for Model 2, which we adopt here for this example.

10.4.2.2 Plotting model results

Whatever model is chosen, as a final step, it is important to determine what that model implies about the original research questions. Because our focus here is on the prevalence of each symptom, and their association, it is helpful to graph the fitted logits and log odds ratios implied by the model, as was done in Figure 10.18 and Figure 10.19.

The presentation goal here is to produce plots showing the observed logits and log odds ratios as in Figure 10.25 and Figure 10.24, supplemented by lines showing these values according to the fitted model. In Example 10.9 we fit the bivariate logit model, for which the response functions were the desired logits and log odds. Here, where we have fit ordinary loglinear models, the observed and fitted logits can be calculated from the observed and fitted frequencies. The calculations require a bit of R calisthenics to arrange these into forms suitable for plotting.

As we did earlier, we first reshape the *Toxaemia* to wide format, as a 15×4 table of observed frequencies. Because there are now two predictor variables, we take care to include the levels of `smoke` and `class` as additional columns.

```
> # reshape to 15 x 4 table of frequencies
> tox.tab <- xtabs(Freq ~ class + smoke + hyper + urea, Toxaemia)
> toxaemia <- t(matrix(aperm(tox.tab), 4, 15))
> colnames(toxaemia) <- c("hu", "hU", "Hu", "HU")
> rowlabs <- expand.grid(smoke = c("0", "1-19", "20+"),
+                        class = factor(1:5))
> toxaemia <- cbind(toxaemia, rowlabs)
```

Applying `blogits()`, we get the observed logits and log odds ratios in `logitsTox`.

```
> # observed logits and log odds ratios
> logitsTox <- blogits(toxaemia[,4:1], add=0.5)
> colnames(logitsTox)[1:2] <- c("logitH", "logitU")
> logitsTox <- cbind(logitsTox, rowlabs)
> head(logitsTox)

    logitH  logitU    logOR smoke class
1 -1.02057 -1.9988  1.52679     0     1
2 -0.94261 -2.1665  1.07102  1-19     1
3 -1.02962 -2.1401  2.44854   20+     1
4 -0.95040 -2.5158  1.46196     0     2
5 -1.04699 -2.4983  0.86401  1-19     2
6 -0.86500 -2.5257 -1.14579   20+     2
```

The fitted frequencies are extracted using `predict(tox.glm2, type="response")`, and then manipulated in a similar way to give `logitsFit`.

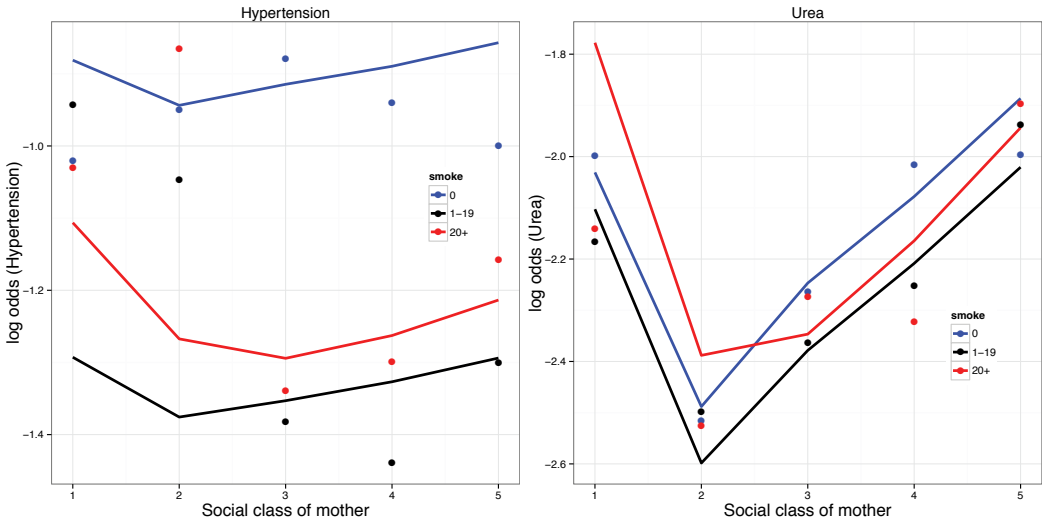


Figure 10.26: Observed (points) and fitted (lines) logits for the *Toxaemia* data under Model 2.

```
> # fitted frequencies, as a 15 x 4 table
> Fit <- t(matrix(predict(tox.glm2, type = "response"), 4, 15))
> colnames(Fit) <- c("HU", "Hu", "hU", "hu")
> Fit <- cbind(Fit, rowlabs)
> logitsFit <- blogits(Fit[, 1 : 4], add=0.5)
> colnames(logitsFit)[1 : 2] <- c("logitH", "logitU")
> logitsFit <- cbind(logitsFit, rowlabs)
```

In tabular form, you can examine any of these components; for example, the log odds ratios from the fitted values shown below.

```
> matrix(logitsFit$logOR, 3, 5,
+         dimnames = list(smoke = c("0", "1-19", "20+"), class = 1 : 5))

      class
smoke       1      2      3      4      5
  0    1.3588 1.3638 1.3675 1.3643 1.3582
  1-19 1.3582 1.3678 1.3683 1.3674 1.3658
  20+  1.2799 1.3471 1.3662 1.3622 1.3511
```

Finally, we can plot the observed values in `logitsTox` (as points) and the fitted values under Model 2 in `logitsFit` (as lines), separately for the `logitH`, `logitU`, and `logOR` components. The code below uses ggplot2 for the log odds of hypertension, and is repeated for urea and the log odds ratio. These graphs are shown in Figure 10.26 and Figure 10.27.

```
> ggplot(logitsFit, aes(x = as.numeric(class), y = logitH,
+                       color = smoke)) +
+   theme_bw() +
+   geom_line(size = 1.2) +
+   scale_color_manual(values = c("blue", "black", "red")) +
+   ylab("log odds (Hypertension)") +
+   xlab("Social class of mother") +
+   ggtitle("Hypertension") +
+   theme(axis.title = element_text(size = 16)) +
+   geom_point(data = logitsTox,
+              aes(x = as.numeric(class), y = logitH, color = smoke),
+              size = 3) +
+   theme(legend.position = c(0.85, .6))
```

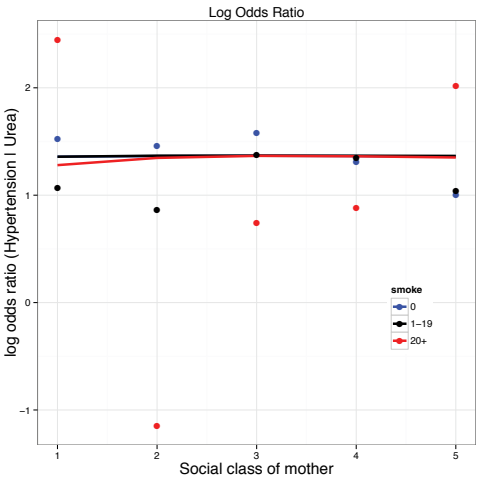


Figure 10.27: Observed (points) and fitted (lines) log odds ratios for the `Toxaemia` data under Model 2.

According to this model, Figure 10.27 shows that the fitted log odds ratio is in fact nearly constant, while Figure 10.26 shows that the log odds for hypertension depend mainly on smoking (with a large difference of the non-smoking mothers from the rest), and that for protein urea depends mainly on social class.[11]

Yet the great variability of the observed points around the fitted curves indicates that these relationships are not well-determined. Adding error bars showing the standard error around each fitted point would indicate that the data conforms as closely to the model as can be expected, given the widely different sample sizes. However, this would make the plots more complex, and so was omitted here. In addition to showing the pattern of the results according to the fitted model, such graphs also help us to appreciate the model's limitations.

△

10.5 Chapter summary

- Standard loglinear models treat all variables as unordered factors. When one or more factors are ordinal, however, loglinear and logit models may be simplified by assigning quantitative scores to the levels of an ordered factor. Such models are often more sensitive and have greater power because they are more focused.
- Models for square tables, with the same row and column categories, are an important special case. For these and other structured tables, a variety of techniques provide the opportunity to fit models more descriptive than the independence model and more parsimonious than the saturated model.
- When there are several categorical responses, along with one or more explanatory variables, some special forms of loglinear and logit models may be used to separate the marginal dependence of each response on the explanatory variables from the interdependence among the responses.

[11] Some possible enhancements to these graphs include (a) plotting on the scale of probabilities or including a right vertical axis showing corresponding probabilities; (b) using the same vertical axis limits for the two graphs for direct comparison.

- In all these cases, the interplay between graphing and fitting is important in arriving at an understanding of the relationships among variables and an adequate descriptive model that is faithful to the details of the data.
- In particular, mosaic-like displays show all the data by areas, and indicate goodness of fit of a model by shading. In contrast, for more complex models, plots of derived quantities like log odds and log odds ratios can be more effective.

10.6 Lab exercises

Exercise 10.1 Example 10.5 presented an analysis of the data on visual acuity for the subset of women in the *VisualAcuity* data. Carry out a parallel analysis of the models fit there for the `men` in this data set, given by:

```
> data("VisualAcuity", package="vcd")
> men <- subset(VisualAcuity, gender=="male", select=-gender)
```

Exercise 10.2 Table 10.2 gives a 4×4 table of opinions about premarital sex and whether methods of birth control should be made available to teenagers aged 14–16, from the 1991 General Social Survey (Agresti, 2013, Table 10.3). Both variables are ordinal, and their grades are represented by the case of the row and column labels.

Table 10.2: Opinions about premarital sex and availability of teenage birth control. *Source*: Agresti (2013, Table 10.3).

Premarital sex	Birth control			
	DISAGREE	disagree	agree	AGREE
WRONG	81	68	60	38
Wrong	24	26	29	14
wrong	18	41	74	42
OK	36	57	161	157

(a) Fit the independence model to these data using `loglm()` or `glm()`.
(b) Make a mosaic display showing departure from independence and describe verbally the pattern of association.
(c) Treating the categories as equally spaced, fit the $L \times L$ model of uniform association, as in Section 10.1. Test the difference against the independence model with a likelihood-ratio test.
(d) Fit the RC(1) model with `gnm()`, and test the difference of this against the model of uniform association.
(e) Write a brief summary of these results, including plots useful for explaining the relationships in this data set.

Exercise 10.3 For the data on attitudes toward birth control in Table 10.2,

(a) Calculate and plot the observed local log odds ratios.
(b) Also fit the R, C, and R+C models.
(c) Use the method described in Section 10.1.2 to visualize the structure of fitted local log odds ratios implied by each of these models, together with the RC(1) model.

Exercise 10.4 The data set *gss8590* in logmult gives a $4 \times 5 \times 4$ table of education levels and

occupational categories for the four combinations of gender and race from the General Social Surveys, 1985–1990, as reported by Wong (2001, Table 2). Wong (2010, Table 2.3B) later used the subset pertaining to women to illustrate RC(2) models. This data is created below as `Women.tab`, correcting an inconsistency to conform with the 2010 table.

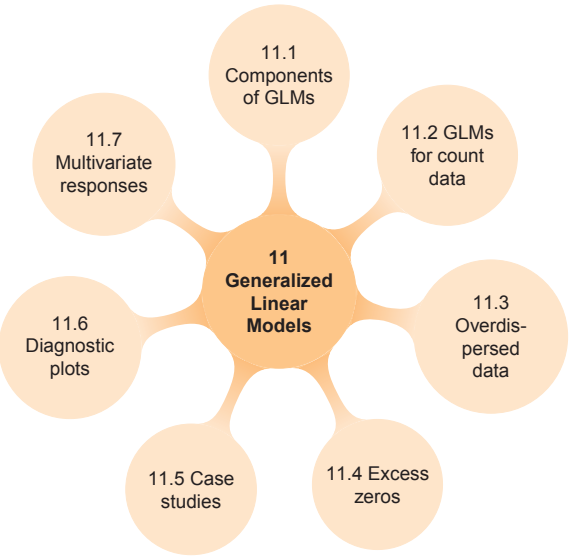
```
> data("gss8590", package="logmult")
> Women.tab <- margin.table(gss8590[,,c("White Women", "Black Women")], 1:2)
> Women.tab[2,4] <- 49
> colnames(Women.tab)[5] <- "Farm"
```

(a) Fit the independence model, and also the RC(1) and RC(2) models using `rc()` with marginal weights, as illustrated in Example 10.4. Summarize these statistical tests in a table.

(b) Plot the solution for the RC(2) model with 68% confidence ellipses. What verbal labels would you use for the two dimensions?

(c) Is there any indication that a simpler model, using integer scores for the row (Education) or column (Occupation) categories, or both, might suffice? If so, fit the analogous column effects, row effects, or $L \times L$ model, and compare with the models fit in part (a).

This page intentionally left blank



11

Generalized Linear Models for Count Data

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

> In one word, to draw the rule from experience, one must generalize; this is a necessity that imposes itself on the most circumspect observer.
>
> Henri Poincaré, *The Value of Science: Essential Writings of Henri Poincaré*

In the modern history of statistics, most developments occur incrementally, with small additions to existing models and theory that extend their range and applicability to new problems and data. Occasionally, there is a major synthesis that unites a wide class of existing methods in a general framework and provides opportunities for far greater growth.

A prime example is the theory of generalized linear models, introduced originally by Nelder and Wedderburn (1972), that extended the familiar (classical) linear models for regression and ANOVA to include related models, such as logistic regression and logit models (described in Chapter 7) and loglinear models (described in Chapter 9), and other variations, as "families" within a single general system.

This approach has proved attractive because it: (a) integrates many familiar statistical models in a general theory where they are just special cases; (b) provides the basis for extending these and

developing new models within the same or similar framework; (c) simplifies the implementation of these models in software, since the same algorithm can be used for estimation, inference, and assessing model adequacy for all generalized linear models.

Section 11.1 gives a brief sketch of the GLM framework. The focus of this book is on visualization methods for categorical data, and the two important topics concern models and methods for binomial response data and for count data. The first of these was described extensively in Chapter 7, with extensions to multinomial data (Chapter 8), and there is little to add here, except for changes in notation.

GLM models for count data, however, provide the opportunity to extend the scope of these methods beyond what was covered in Chapter 9, and this topic is introduced in Section 11.2. Extensions to the GLM framework also provide the opportunity to deal with common problems of overdispersion (Section 11.3) and an overabundance of zero counts (Section 11.4), giving some new models and visualization methods that help to understand such data in greater detail. These are illustrated with two case studies in Section 11.5. Section 11.6 illustrates other graphical methods for diagnostic model checking, some of which were introduced in earlier chapters. Finally, Section 11.7 outlines some simple extensions of these models to handle multivariate responses.

11.1 Components of generalized linear models

The motivation for the **generalized linear model** (GLM) and its structure are most easily seen by considering the classical linear model,

$$y_i = \boldsymbol{x}_i^\mathsf{T} \boldsymbol{\beta} + \epsilon_i \; ,$$

where y_i is the response variable for case $i, i = 1, \ldots n$, $\boldsymbol{x}_i$ is the vector of explanatory variables or regressors, $\boldsymbol{\beta}$ is the vector of model parameters, and the ϵ_i are random errors. In the classical linear model, the ϵ_i are assumed to (a) have constant variance, σ^2_ϵ, (b) follow a normal (Gaussian) distribution (conditional on $\boldsymbol{x}_i$), and (c) be independent across observations.

Thus, Nelder and Wedderburn (1972) generalized this Gaussian linear model to consist of the following three components, by relaxing assumptions (a) and (b) above:[1]

random component: The conditional distribution of the $y_i \,|\, \boldsymbol{x}_i$, with mean $\mathcal{E}(y_i) = \mu_i$. Under classical assumptions, this is independent, normal with constant variance σ^2, i.e., $y_i \overset{\text{iid}}{\sim} N(\mu_i, \sigma^2)$. In the GLM, the probability distribution of the y_i can be any member of the ***exponential family***, including the normal, Poisson, binomial, gamma, and others. Subsequent work has extended this framework to include multinomial distributions and some non-exponential families such as the negative binomial distribution.

systematic component: The idea that the predicted value of y_i itself is a linear combination of the regressors is replaced by that of a ***linear predictor***, η, that captures this aspect of linear models,

$$\eta_i = \boldsymbol{x}_i^\mathsf{T} \boldsymbol{\beta} \; .$$

link function: The connection between the mean of the response, μ_i, and the linear predictor, η_i, is specified by the ***link function***, $g(\bullet)$, giving

$$g(\mu_i) = \eta_i = \boldsymbol{x}_i^\mathsf{T} \boldsymbol{\beta} \; .$$

[1] The remaining assumption of independent observations is relaxed in ***generalized linear mixed models*** (GLMMs), in which random effects to account for non-independence are added to the linear predictor. This allows the modeling of correlated (responses of family members), clustered (residents in different communities), or hierarchical data (patients within hospitals within regions). See: McCulloch and Neuhaus (2005) and Hedeker (2005)

Table 11.1: Common link functions and their inverses used in generalized linear models

Link name	Function: $\eta_i = g(\mu_i)$	Inverse: $\mu_i = g^{-1}(\eta_i)$
identity	μ_i	η_i
square-root	$\sqrt{\mu_i}$	η_i^2
log	$\log_e(\mu_i)$	$\exp(\eta_i)$
inverse	μ_i^{-1}	η_i^{-1}
inverse-square	μ_i^{-2}	$\eta_i^{-1/2}$
logit	$\log_e \frac{\mu_i}{1-\mu_i}$	$\frac{1}{1+\exp(-\eta_i)}$
probit	$\Phi^{-1}(\mu_i)$	$\Phi(\eta_i)$
log-log	$-\log_e[-\log_e(\mu_i)]$	$\exp[-\exp(-\eta_i)]$
comp. log-log	$\log_e[-\log_e(1-\mu_i)]$	$1-\exp[-\exp(\eta_i)]$

The link function $g(\bullet)$ must be both *smooth* and *monotonic*, meaning that it is one-to-one, so an inverse transformation, $g^{-1}(\bullet)$ exists,

$$\mu_i = g^{-1}(\eta_i) = g^{-1}(\boldsymbol{x}_i^\mathsf{T}\boldsymbol{\beta}) \;,$$

which allows us to obtain and plot the predicted values on their original scale. The link function captures the familiar idea that linear models are often estimated with a transformation of the response, such as $\log(y_i)$ for a frequency variable or $\text{logit}(y_i)$ for a binomial variable. The inverse function $g^{-1}(\bullet)$ is also called the ***mean function***.

Some commonly used link functions are shown in Table 11.1. Some of these link functions have restrictions on the range of y_i to which they can be applied. For example, the square-root and log links apply only to non-negative and positive values, respectively. The last four link functions in this table are for binomial data, where y_i represents the observed proportion of successes in n_i independent trials, and thus the mean μ_i represents the probability of success (symbolized by π_i in Chapter 7). Binary data are the special case where $n_i = 1$.

11.1.1 Variance functions

The GLM has the additional property that, for distributions in the exponential family, the conditional variance of $y_i \,|\, \eta_i$ is a known function, $\mathcal{V}(\mu_i)$, of the mean and possibly one other parameter called the ***scale parameter*** or ***dispersion parameter***, ϕ. Some commonly used distributions in the exponential family and their variance functions are shown in Table 11.2.

- In the classical Gaussian linear model, the conditional variance is constant, $\phi = \sigma_\epsilon^2$.
- In the Poisson family, $\mathcal{V}(\mu_i) = \mu_i$ and the dispersion parameter is fixed at $\phi = 1$. In practice, it is common for count data to exhibit ***overdispersion***, meaning that $\mathcal{V}(\mu_i) > \mu_i$. One way to correct for this is to extend the GLM to allow the dispersion parameter to be estimated from the data, giving what is called the ***quasi-Poisson*** family, with $\mathcal{V}(\mu_i) = \widehat{\phi}\mu_i$.
- Similarly, for binomial data, the variance function is $\mathcal{V}(\mu_i) = \mu_i(1-\mu_i)/n_i$, with ϕ fixed at 1. Overdispersion often results from failures of the assumptions of the binomial model: supposedly independent observations may be correlated or clustered and the probability of success may not be constant, or vary with unmeasured or unmodeled variables.

Table 11.2: Common distributions in the exponential family used with generalized linear models and their canonical link and variance functions

Family	Notation	Canonical link	Range of y	Variance function, $\mathcal{V}(\mu \mid \eta)$
Gaussian	$N(\mu, \sigma^2)$	identity: μ	$(-\infty, +\infty)$	ϕ
Poisson	$\text{Pois}(\mu)$	$\log_e(\mu)$	$0, 1, \ldots, \infty$	μ
Negative-Binomial	$\text{NBin}(\mu, \theta)$	$\log_e(\mu)$	$0, 1, \ldots, \infty$	$\mu + \mu^2/\theta$
Binomial	$\text{Bin}(n, \mu)/n$	$\text{logit}(\mu)$	$\{0, 1, \ldots, n\}/n$	$\mu(1-\mu)/n$
Gamma	$G(\mu, \nu)$	μ^{-1}	$(0, +\infty)$	$\phi\mu^2$
Inverse-Gaussian	$IG(\mu, \nu)$	μ^2	$(0, +\infty)$	$\phi\mu^3$

- The gamma and inverse-Gaussian families are distributions useful for modeling a continuous and positive response variable with no upper bound (e.g., reaction time). They both have the property that conditional variance increases with the mean, and for the inverse-Gaussian, variance increases at a faster rate. Their dispersion parameters ϕ are simple functions of their intrinsic "shape" parameters, indicated as ν in the table.

The important points from this discussion are that the GLM together with the exponential family of distributions:

- provide for simple linear relations between the response and the predictors via the link function and the linear predictor.
- allow a very flexible relationship between the mean and conditional variance to be specified in terms of a set of known families.
- incorporate a dispersion parameter ϕ that in some cases can be estimated or tested for departure from that entailed in a given family.
- have allowed further extensions of this framework outside the exponential family, ranging from simple adjustments for statistical inference ("quasi" families, adjusted "sandwich" covariances) to separate modeling of the variance relation to the predictors.

Further details of generalized linear models are beyond the scope of this book, but the interested reader should consult Fox (2008, Section 15.3) and Agresti (2013, Ch. 4) for a comprehensive treatment.

11.1.2 Hypothesis tests for coefficients

GLMs are fit using maximum likelihood estimation, and implemented in software using an iterative algorithm known as *iteratively weighted least squares* that generalizes the least squares method for classical linear models. This provides estimates $\widehat{\boldsymbol{\beta}}$ of the model coefficients for the predictors in $\boldsymbol{x}$, as well as an estimated asymptotic (large sample) variance matrix of $\widehat{\boldsymbol{\beta}}$, given by

$$\mathcal{V}(\widehat{\boldsymbol{\beta}}) = \phi(\boldsymbol{X}^\mathsf{T}\boldsymbol{W}\boldsymbol{X}) \,, \tag{11.1}$$

where $\boldsymbol{W}$ is a diagonal matrix of weights computed in the final iteration. In the standard Poisson GLM, the weight matrix is $\boldsymbol{W} = \text{diag}\,(\widehat{\boldsymbol{\mu}})$ and $\phi = 1$ is assumed.

Asymptotic standard errors, $\text{se}(\widehat{\beta}_j)$, for the coefficients are then the square roots of the diagonal elements of $\mathcal{V}(\widehat{\boldsymbol{\beta}})$, and tests of hypotheses regarding an individual coefficient, e.g., $H_0 : \beta_j = 0$,

can be carried out using the Wald test statistic, $z_j = \widehat{\beta}_j / \text{se}(\widehat{\beta}_j)$. When the null hypothesis is true, z_j has a standard normal $\mathcal{N}(0,1)$ distribution, providing p-values for significance tests.[2]

More generally, we can test any ***linear hypothesis***, of the form $H_0 : \boldsymbol{L}\boldsymbol{\beta} = \boldsymbol{c}$, where $\boldsymbol{L}$ is a constant hypothesis matrix of size $h \times p$ giving h linear combinations of the coefficients, to be tested for equality with the constants in $\boldsymbol{c}$, typically taken as $\boldsymbol{c} = \boldsymbol{0}$. The test statistic is the Wald chi-square,

$$Z^2 = (\boldsymbol{L}\widehat{\boldsymbol{\beta}} - \boldsymbol{c})^\mathsf{T} \, [\boldsymbol{L}\mathcal{V}(\widehat{\boldsymbol{\beta}})\boldsymbol{L}^\mathsf{T}]^{-1} \, (\boldsymbol{L}\widehat{\boldsymbol{\beta}} - \boldsymbol{c}) \, , \tag{11.2}$$

which has a χ^2 distribution on h degrees of freedom.[3]

For example, to test the hypothesis that all of $\beta_1 = \beta_2 = \beta_3 = 0$ in a model with three predictors, you can use

$$\boldsymbol{L} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{0} & \boldsymbol{I} \end{bmatrix} , \qquad \boldsymbol{c} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} .$$

Similarly, to test the hypothesis that $\beta_1 = \beta_2$ in the same model, you can use $\boldsymbol{L} = [0, 1, -1, 0]$ and $\boldsymbol{c} = [0]$.[4]

In R, such tests are most conveniently carried out using `linearHypothesis()` in the car package, supporting Fox and Weisberg (2011b). The hypothesis matrix $\boldsymbol{L}$ can be supplied as a numeric matrix, or more conveniently, the hypothesis can be specified symbolically as a character vector of the names of the coefficients involved in each row of $\boldsymbol{L}$. For example, the first hypothesis test above could be specified using the vector `c("x1=0", "x2=0", "x3=0")`, and the test of equality as `"x1-x2=0"`.

11.1.3 Goodness-of-fit tests

The basic ideas for testing goodness-of-fit were discussed in Section 9.3.2 in connection with log-linear models for contingency tables. As before, these assess the overall performance of a model in reproducing the data. The commonly used measures include the Pearson chi-square and likelihood-ratio deviance statistics, which can be seen as weighted sums of residuals. We re-state these test statistics here in the wider context of the GLM.

Let $y_i, i = 1, 2, \ldots, n$ be the response and $\widehat{\mu}_i = g^{-1}(\boldsymbol{x}_i^\mathsf{T}\widehat{\boldsymbol{\beta}})$ the fitted mean using the estimated coefficients, having estimated variance $\widehat{\omega}_i = \mathcal{V}(\widehat{\mu}_i \,|\, \eta_i)$ as in Table 11.2. Then the normalized squared residual for observation i is $(y_i - \widehat{\mu}_i)^2 / \widehat{\omega}_i$, and the Pearson statistic is

$$X^2_P = \sum_{i=1}^{n} \frac{(y_i - \widehat{\mu}_i)^2}{\widehat{\omega}_i} \, . \tag{11.3}$$

In the GLM for count data, the main focus of this chapter, the Poisson family sets $\omega = \mu$ with the dispersion parameter fixed at $\phi = 1$.

The ***residual deviance*** statistic, as in logistic regression and loglinear models, is defined as twice the difference between the maximum possible log-likelihood for the *saturated model* that fits perfectly and maximized log-likelihood for the fitted model. The deviance can be defined as

$$D(\boldsymbol{y}, \widehat{\boldsymbol{\mu}}) \equiv 2[\log_e \mathcal{L}(\boldsymbol{y}; \boldsymbol{y}) - \log_e \mathcal{L}(\boldsymbol{y}; \widehat{\boldsymbol{\mu}})] \, .$$

For classical linear models under normality, the deviance is simply the residual sum of squares,

[2] Wald tests are sometimes carried out using z^2, which has an equivalent χ^2_1 distribution with 1 degree of freedom.

[3] When a dispersion parameter ϕ has been estimated from the data, it is common to use an F-test, using the statistic $F = Z^2/h$, with h and $n - p$ degrees of freedom.

[4] Such a test is only sensible if the predictors $\boldsymbol{x_1}$ and $\boldsymbol{x_2}$ are on the same scale, so their coefficients are commensurable.

$\sum_i^n (y_i - \widehat{\mu}_i)$. This has led to the deviance being taken in the GLM framework as a generalization of the sum of squares used in ANOVA, and hence, an analogous ***analysis of deviance*** to carry out tests for individual terms in GLMs, or to compare nested models.

In R, `anova(mod)` for the "glm" object `mod` gives *sequential* ("Type I") tests of successive terms in a model, while `Anova()` in the car package gives the more generally useful "Type II" (and "Type III") *partial* tests, that assess the additional contribution of each term above all others, taking marginality into account.

For Poisson models with a log link giving $\boldsymbol{\mu} = \exp(\boldsymbol{x}^\mathsf{T}\boldsymbol{\beta})$, the deviance takes the form[5]

$$D(\boldsymbol{y}, \widehat{\boldsymbol{\mu}}) = 2 \sum_{i=1}^{n} \left[y_i \log_e \left(\frac{y_i}{\widehat{\mu}_i} \right) - (y_i - \widehat{\mu}_i) \right] . \tag{11.4}$$

For a GLM with p parameters, both the Pearson and residual deviance statistics follow approximate χ^2_{n-p} distributions with $n - p$ degrees of freedom.

11.1.4 Comparing non-nested models

The flexibility of the GLM and its extensions allows us to fit models to the same data using different families and different link functions, and to fit models that allow for overdispersion (Section 11.3) or that make special provisions for zero counts (Section 11.4). One price paid for this additional versatility is that standard LR tests and F tests (such as provided by `anova()` and `linearHypothesis()` in the car package) do not apply to models that are not nested; that is, where one model cannot be represented as a restricted, special case of another.

For models estimated by maximum likelihood, one general route to comparing non-nested models is through the AIC information criterion proposed initially by Akaike (1973) and the related BIC criterion (Schwartz, 1978), based on the fitted log-likelihood function:

$$\text{AIC} = -2 \log_e \mathcal{L} + 2k . \tag{11.5}$$

$$\text{BIC} = -2 \log_e \mathcal{L} + \log_e(n) k . \tag{11.6}$$

As noted in Section 9.3.2, these both penalize models with larger k, the number of parameters in the model, with BIC adding a greater penalty with larger sample size. However, because they are based only on the maximized log-likelihood, they are agnostic as to whether models are nested or not, and give comparable results (lower is better) provided the same observations have been used in all models.

In R, these results are given for a collection of models by the generic functions `AIC()` and `BIC()`; these can be calculated for any model for which `logLik()` and (for BIC) `nobs()` methods exist. The vcdExtra function `LRstats()` is a convenient wrapper for these methods.

AIC and BIC do not give significance tests for assessing whether one model is significantly "better" than another. A series of tests that *do* this was proposed by Vuong (1989): they are based on comparing the predicted probabilities or the pointwise log-likelihoods of the two models, and test the null hypothesis that each is equally close to the saturated model, against the alternative that one model is closer. The different tests handle nested, partially nested and non-nested cases. However, whenever ***Vuong's test*** is mentioned in literature, this typically refers to the test assuming that both models are *strictly* non-nested, which may not be obvious to see in all cases.[6] For example, some models may neither be nested or non-nested, but *overlapping*, that is, yield the same moments and fit statistics only for some, not all data. However, our use of Vuong's test will be confined to count data models, precluding most of these issues.

[5] In the context of the loglinear models discussed in Section 9.3.2, this is also referred to as the likelihood-ratio G^2 statistic.

[6] The test versions for (partially) nested models are difficult to compute in practice, and at the time of writing, no implementation for them is available for R.

For two such models, let $f_1(y_i \mid \boldsymbol{x}_i, \boldsymbol{\theta}_1)$ be the density function under model 1, with parameters $\boldsymbol{\theta}_1$ and similarly $f_2(y_i \mid \boldsymbol{x}_i, \boldsymbol{\theta}_2)$ under model 2 with parameters $\boldsymbol{\theta}_2$, where $f_1(\bullet)$ and $f_2(\bullet)$ need not be the same. Vuong's test compares these based on the observation-wise log-likelihood ratios,

$$\ell_i = \log_e \left(\frac{f_1(y_i \mid \boldsymbol{x}_i, \widehat{\boldsymbol{\theta}}_1)}{f_2(y_i \mid \boldsymbol{x}_i, \widehat{\boldsymbol{\theta}}_2)} \right) .$$

The test statistic is

$$V = \frac{\bar{\ell} - \text{penalty}}{\sqrt{n} s_\ell} ,$$

where $\bar{\ell}$ is the mean of the ℓ_i, s_ℓ is their variance, and penalty is an adjustment for model parsimony, typically taken as $\log(n)(k_1 - k_2)/2$ when model 1 has k_1 parameters in $\boldsymbol{\theta}_1$ and model 2 has k_2 parameters in $\boldsymbol{\theta}_2$.

The test statistic V has an asymptotic normal $N(0, 1)$ distribution, and is directional, with large positive values favoring model 1, and large negative values favoring model 2. This test is implemented as the `vuong()` function in the pscl (Jackman et al., 2015) package, and a more flexible version is provided by `vuongtest()` in nonnest2 (Merkle and You, 2014) package[7].

11.2 GLMs for count data

The prototypical GLM for count data, where the response y_i takes on non-negative values $0, 1, 2, \ldots$, uses the Poisson family with the log link. We used this model extensively throughout all of Chapter 9. There the focus was on the special case of the loglinear model applied largely to contingency tables, where the loglinear model could be seen as a fairly direct extension of ANOVA models for a quantitative response applied to the log of cell frequency.

The advantage there was that models for two-way, three-way, and by implication, *n*-way tables could be discussed and illustrated using notation and graphs that separated the parameters and effects for one-way terms ("main effects"), two-way terms ("simple associations"), and higher-way terms ("conditional associations").

The disadvantage is that these models as formulated there do not easily accommodate general quantitative predictors and were limited to the log link and the Poisson family. For example, the models discussed in Section 10.1 for ordinal variables allow one or more table factors to be assigned quantitative scores or have such scores estimated from the data, as in RC() models (Section 10.1.3). Yet the contingency table approach for loglinear models breaks down if there are continuous predictors, and count data often exhibits features that make the equivalent Poisson regression model unsuitable or incomplete. We consider some extended models here.

EXAMPLE 11.1: Publications of PhD candidates

In Example 3.24 we considered the distribution of the number of publications by PhD candidates in their last three years of study, but without taking any available predictors into account. For these data, a simple calculation shows why the Poisson distribution is unsuitable (for the marginal distribution), because the variance is 2.19 times the mean.

```
> data("PhdPubs", package = "vcdExtra")
> with(PhdPubs, c(mean = mean(articles), var = var(articles),
+                 ratio = var(articles) / mean(articles)))

  mean    var  ratio
1.6929 3.7097 2.1914
```

[7]This also allows for testing *nested* models where the full model is not assumed to be correct as required by classical likelihood ratio tests, and also for other models than count regression models, such as Structural Equation Models (SEMs).

The earlier example showed rootograms (in Figure 3.25) of the number of articles, but here it is useful to consider some more basic exploratory displays. A basic barplot of the frequency distribution of number of articles published is shown in the left panel of Figure 11.1. A quick look indicates that the distribution is highly skewed and there is a large number of counts of zero.

Another problem is that the frequencies of 0–2 articles account for over 75% of the total, so that the frequencies of the larger counts get lost in the display. The rootogram corrects for this by plotting frequency on the square-root scale. However, because we are contemplating a model with a log link, the same goal can be achieved by plotting log of frequency, as shown in the right panel of Figure 11.1. To accommodate the zero frequencies, the plot shows log(Frequency+1), avoiding errors from `log(0)`. It can be seen that log frequency decreases steadily up to 7 articles and then levels off approximately.

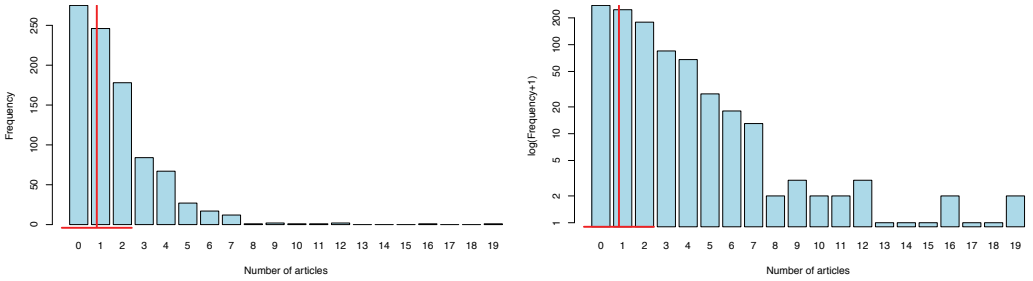


Figure 11.1: Barplots showing the frequency distribution of number of publications by PhD candidates. Left: raw scale; right: a log scale makes the smaller counts more visible. The vertical red lines show the mean and horizontal lines show mean ± 1 standard deviation.

These plots are produced as shown below. The frequency distribution of `articles` can be tabulated by `table()`, but there is a subtle wrinkle here: By default, `table()` excludes the values of `articles` that do not occur in the data (zero frequencies). To include all values in the entire range, it is necessary to treat `articles` as a factor with levels `0:19`.

```
> art.fac <- factor(PhdPubs$articles, levels = 0 : 19)  # include zero frequencies
> art.tab <- table(art.fac)
> art.tab

art.fac
  0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19
275 246 178  84  67  27  17  12   1   2   1   1   2   0   0   0   1   0   0   1
```

Then, the basic plot on the frequency scale is created using `barplot()`, to which some annotations can be added using standard plotting tools, such as the mean or an interval showing the variance (e.g., with a range of one standard deviation).

```
> barplot(art.tab, xlab = "Number of articles", ylab = "Frequency",
+         col = "lightblue")
> abline(v = mean(PhdPubs$articles), col = "red", lwd = 3)
> ci <- mean(PhdPubs$articles) + c(-1, 1) * sd(PhdPubs$articles)
> lines(x = ci, y = c(-4, -4), col = "red", lwd = 3, xpd = TRUE)
```

Similarly, the plot on the log scale in the right panel of Figure 11.1 is produced with `barplot()`, but using `art.tab+1` to start frequency at one and `log="y"` to scale the vertical axis to log.

```
> barplot(art.tab + 1, ylab = "log(Frequency+1)",
+         xlab = "Number of articles", col = "lightblue", log = "y")
```

Other useful exploratory plots for count data include boxplots of the response (on a log scale) and scatterplots against continuous predictors, where jittering the response is often necessary to avoid overplotting and a smooth nonparametric curve can show possible nonlinearity. The `log="y"` option is again handy, and the formula method allows adding a start value to the response. Figure 11.2 illustrates these ideas, for the factor `married` and the covariate `mentor`.

```
> boxplot(articles + 1 ~ married, data = PhdPubs, log = "y",
+         varwidth = TRUE, ylab = "log(articles + 1)", xlab = "married",
+         cex.lab = 1.25)
> plot(jitter(articles + 1) ~ mentor, data = PhdPubs, log = "y",
+      ylab="log(articles + 1)", cex.lab = 1.25)
> lines(lowess(PhdPubs$mentor, PhdPubs$articles + 1), col = "blue",
+       lwd = 3)
```

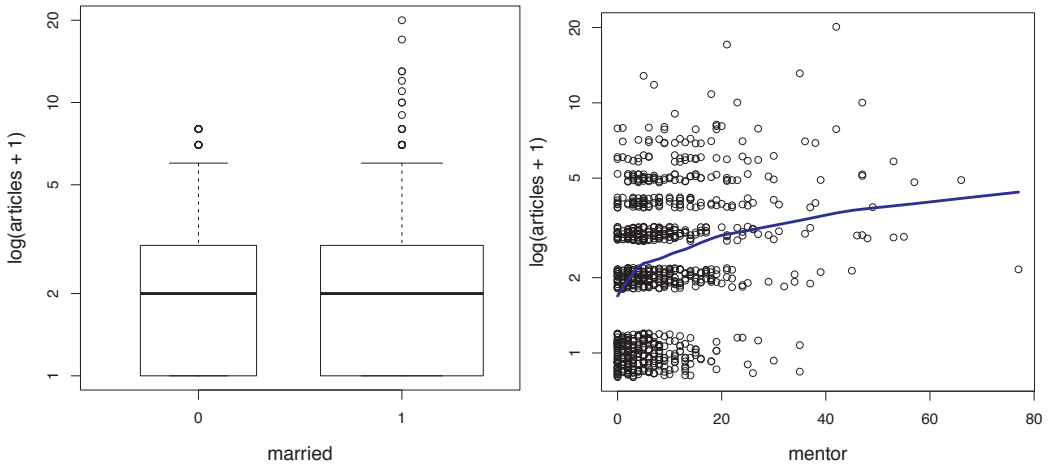


Figure 11.2: Exploratory plots for the number of articles in the PhdPubs data. Left: boxplots for married (1) vs. non-married (0); right: jittered scatterplot vs. mentor publications with a lowess smoothed curve.

It can be seen that the distribution of articles for married and non-married are quite similar, except that for the married students there are quite a few observations with a large number of publications. The relationship between log(articles) and mentor publications seems largely linear except possibly at the very low end. The large number of zero counts at the lower left corner stands out; this would not be seen without jittering.

Plots similar to those in Figure 11.2 can also be produced using ggplot2 with greater flexibility, but perhaps greater effort to get the details right. One key feature is the use of `scale_y_log10()` to plot the response, and all other features on a log scale. The following code gives a plot similar to the right panel of Figure 11.2, but also plots a confidence band around the smoothed curve, and adds a linear regression line of log(articles) on mentor publications. This plot is not shown here, but it is a good exercise to reproduce it for yourself.

```
> ggplot(PhdPubs, aes(mentor, articles + 1)) +
+   geom_jitter(position = position_jitter(h = 0.05)) +
+   stat_smooth(method = "loess", size = 2, fill = "blue", alpha = 0.25) +
+   stat_smooth(method = "lm", color = "red", size = 1.25, se = FALSE) +
+   scale_y_log10(breaks = c(1, 2, 5, 10, 20)) +
+   labs(y = "log(articles + 1)", x = "Mentor publications")
```

To start analysis, we fit the Poisson model using all predictors—`female`, `married`, `kid5`, `phdprestige`, and `mentor`. As recorded in *PhdPubs*, `female` and `married` are both dummy (0/1) variables, and it slightly more convenient for plotting purposes to make them factors.

```
> PhdPubs <- within(PhdPubs, {
+   female <- factor(female)
+   married <- factor(married)
+ })
```

The model is fit as shown below and summarized using `summary()`.

```
> phd.pois <- glm(articles ~ ., data = PhdPubs, family = poisson)
> summary(phd.pois)


Call:
glm(formula = articles ~ ., family = poisson, data = PhdPubs)

Deviance Residuals:
   Min      1Q  Median      3Q     Max
-3.488  -1.538  -0.365   0.577   5.483

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  0.26562    0.09962    2.67   0.0077 **
female1     -0.22442    0.05458   -4.11  3.9e-05 ***
married1     0.15732    0.06125    2.57   0.0102 *
kid5        -0.18491    0.04012   -4.61  4.0e-06 ***
phdprestige  0.02538    0.02527    1.00   0.3153
mentor       0.02523    0.00203   12.43  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 1817.4  on 914  degrees of freedom
Residual deviance: 1633.6  on 909  degrees of freedom
AIC: 3313

Number of Fisher Scoring iterations: 5
```

Significance tests for the individual coefficients show that all are significant, except for `phdprestige`. We ignore this here, and continue to interpret and extend the full main effects model.[8]

The estimated coefficients β for the predictors are shown below. Recall that using the log link means, for example, that being married increases the log of the expected number of articles published by 0.157, holding all other predictors constant. Each additional child of age 5 or less decreases this by 0.185.

```
> round(cbind(beta = coef(phd.pois),
+             expbeta = exp(coef(phd.pois)),
+             pct = 100 * (exp(coef(phd.pois)) - 1)), 3)

              beta expbeta     pct
(Intercept)  0.266   1.304  30.425
female1     -0.224   0.799 -20.102
married1     0.157   1.170  17.037
kid5        -0.185   0.831 -16.882
phdprestige  0.025   1.026   2.570
mentor       0.025   1.026   2.555
```

[8] It is usually less harmful to include a non-significant predictor (which in any case may be a variable useful to control, as `phdprestige` here), than to omit a potentially important predictor, or worse—to fail to account for an important interaction.

It is somewhat easier to interpret the exponentiated coefficients, $\exp(\boldsymbol{\beta})$, as multiplicative effects on the expected number of articles and convert these to percentage change, again holding other predictors constant. For example, expected publications by married candidates are 1.17 times that of non-married, a 17% increase, while each additional child multiplies articles by 0.831, a 16.88% decrease. Alternatively, we recommend visual displays for model interpretation, and effect plots do well in most cases, as shown in Figure 11.3. For a Poisson GLM, an important feature is that the response is plotted on the log scale, so that effects in the model appear as linear functions, while the values of the response (number of articles) are labeled on their original scale, facilitating interpretation. The confidence bands and error bars give 95% confidence intervals around the fitted effects.

```
> library(effects)
> plot(allEffects(phd.pois), band.colors = "blue", lwd = 3,
+      ylab = "Number of articles", main = "")
```

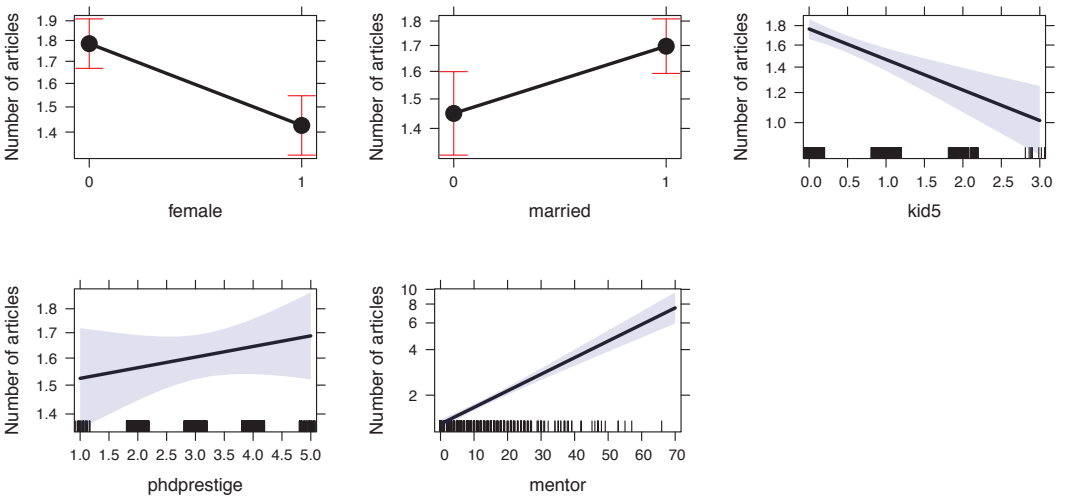


Figure 11.3: Effect plots for the predictors in the Poisson regression model for the PhdPubs data. Jittered values of the continuous predictors are shown at the bottom as rug-plots.

In Figure 11.3 we can see the decrease in published articles with number of young children, but also that the confidence band gets wider with increasing children. The predicted effect here of number of publications by the student's mentor is more dramatic, particularly for those whose mentor was truly prolific. You should note that the panels for the predictors in Figure 11.3 are scaled individually for the range of the fitted main effects. This is often a sensible default and all predictors except `mentor` give a similar range here. To make all of these plots strictly comparable, provide a `ylim` argument, giving the range of the response on the log scale, as below (but not shown here).

```
> plot(allEffects(phd.pois), band.colors = "blue", ylim = c(0, log(10)))
```

All of the above is useful, but still leaves aside the question of how well the Poisson model fits the data. The output from `summary(phd.pois)` above showed that the Poisson model fits quite badly. The residual deviance of 1633.6 with 909 degrees of freedom is highly significant.

△

EXAMPLE 11.2: Mating of horseshoe crabs

Brockmann (1996) studied the mating behavior of female horseshoe crabs in the Gulf of Mexico. In the mating season, crabs arrive on the beach in female/male pairs to lay and fertilize eggs. However, unattached males, called "satellites", also come to the beach, crowd around the nesting couples and compete with attached males for fertilizations, contributing to reproductive success. Some females are ignored by satellite males, and some attract more satellites than others, and the question is: what factors contribute to the number of satellites for each female? Or, perhaps better, how do unattached males choose among available females? This is another example in which zero counts may require special treatment.

The data, given in *CrabSatellites* in the countreg (Zeileis and Kleiber, 2014) package, give the response variable `satellites` for 173 females. Possible predictors are the female's `color` and `spine` condition, given as ordered factors, as well as her `weight` and carapace (shell) `width`.

```
> data("CrabSatellites", package = "countreg")
> str(CrabSatellites)
```

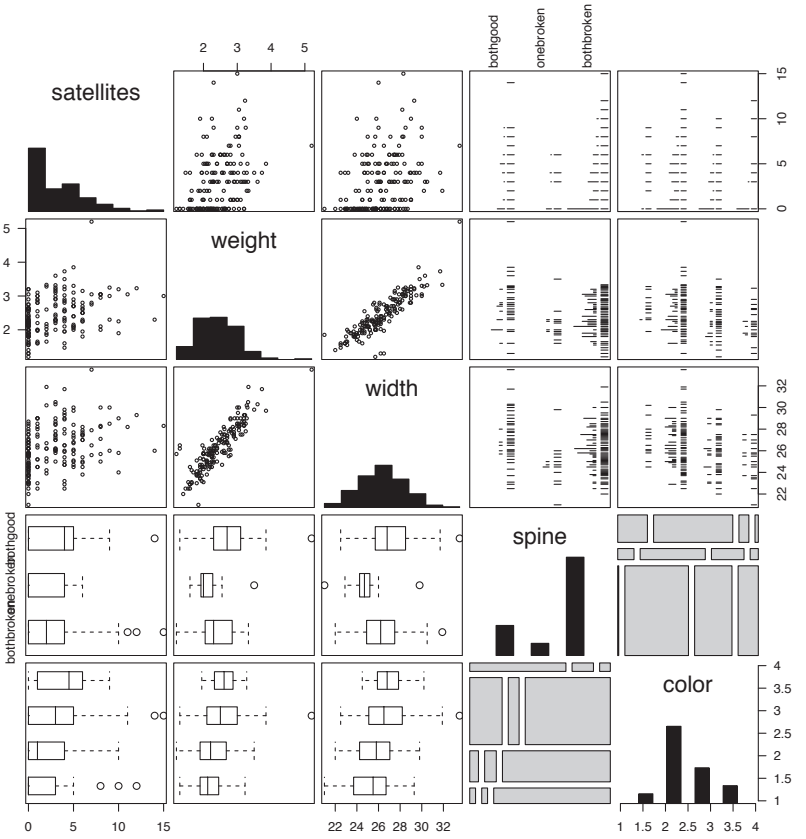


Figure 11.4: Generalized pairs plot for the CrabSatellites data.

```
'data.frame': 173 obs. of  5 variables:
 $ color     : Ord.factor w/ 4 levels "lightmedium"<..: 2 3 3 4 2 1 4 2 2 2 ...
 $ spine     : Ord.factor w/ 3 levels "bothgood"<"onebroken"<..: 3 3 3 2 3 2 3 3 1 3 ...
 $ width     : num  28.3 26 25.6 21 29 25 26.2 24.9 25.7 27.5 ...
 $ weight    : num  3.05 2.6 2.15 1.85 3 2.3 1.3 2.1 2 3.15 ...
 $ satellites: int  8 4 0 0 1 3 0 0 8 6 ...
```

Agresti (2013, Section 4.3) analyzes the number of satellites using count data GLMs, and in his Chapter 5, describes separate logistic regression models for the binary outcome of one or more satellites vs. none. Later in this chapter (Section 11.4) we consider hurdle and zero-inflated models for count data. These have the advantage of modeling the zero counts together with a model for the positive counts.

A useful overview plot of the data is shown using `gpairs()` in Figure 11.4. You can see that the distribution of `satellites` is quite positively skewed, with many zero counts. `width` and `weight` are highly correlated (0.89), and both relate to the size of the female. Their scatterplots in the first row show that larger females attract more satellites. The categorical ordered factors `spine` condition and `color` are strongly associated, with the lightest colored crabs having the best conditions.

```
> library(vcd)
> library(gpairs)
> gpairs(CrabSatellites[, 5 : 1],
+         diag.pars = list(fontsize = 16))
```

Figure 11.5 shows the scatterplots of `satellites` against `width` and `weight` together with smoothed lowess curves.

```
> plot(jitter(satellites) ~ width, data = CrabSatellites,
+   ylab = "Number of satellites (jittered)", xlab = "Carapace width",
+   cex.lab = 1.25)
> with(CrabSatellites, lines(lowess(width, satellites), col = "red", lwd = 2))
> plot(jitter(satellites) ~ weight, data = CrabSatellites,
+   ylab = "Number of satellites (jittered)", xlab = "Weight",
+   cex.lab = 1.25)
> with(CrabSatellites, lines(lowess(weight, satellites), col = "red", lwd = 2))
```

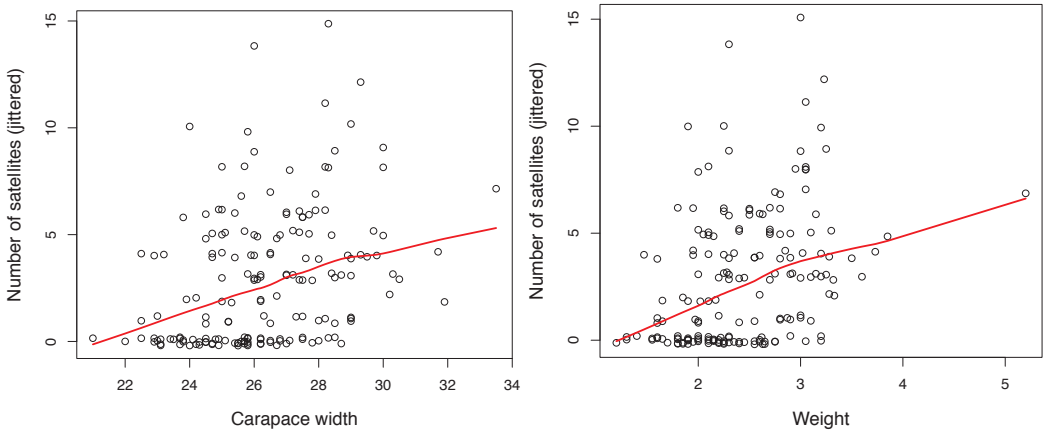


Figure 11.5: Scatterplots of number of satellites vs. width and weight, with lowess smooths.

Both variables show approximately linear relations to the mean number of satellites, so it would not be unreasonable to fit models using the identity link ($\mu \sim x$) rather than the log link ($\mu \sim \log(x)$) with the Poisson family GLM.

In these plots, we reduce the problem of overplotting of the discrete response by jittering, but an alternative technique is to transform a numeric count or continuous predictor to a factor (for visualization purposes only), thereby giving boxplots. A convenience function for this purpose, `cutfac()`, is defined in vcdExtra. It acts like `cut()`, but gives nicer labels for the factor levels and by default chooses convenient breaks among the values based on deciles. Using this, the plots in Figure 11.5 can be re-drawn as boxplots, giving Figure 11.6.

```
> plot(satellites ~ cutfac(width), data = CrabSatellites,
+      ylab = "Number of satellites", xlab = "Carapace width (deciles)")
> plot(satellites ~ cutfac(weight), data = CrabSatellites,
+      ylab = "Number of satellites", xlab = "Weight (deciles)")
```

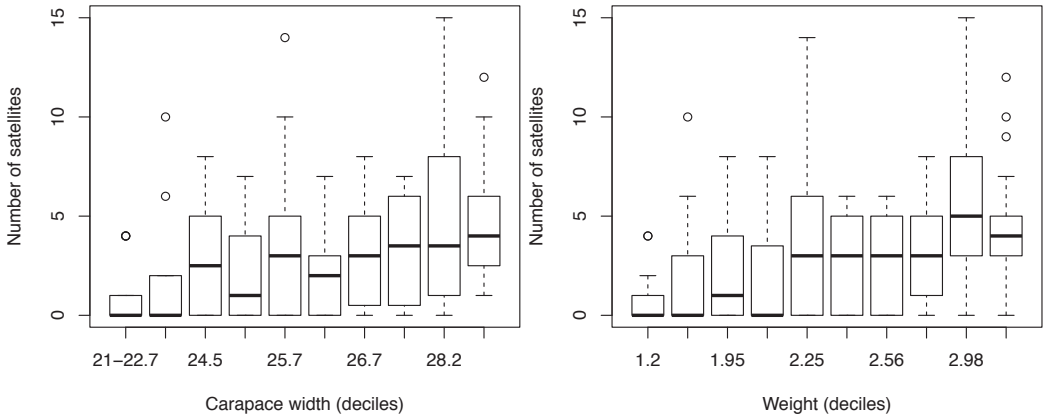


Figure 11.6: Boxplots of number of satellites vs. width and weight.

With this visual overview, we proceed to an initial Poisson GLM model, using all predictors. Note that `color` and `spine` are ordered factors, so `glm()` represents them as polynomial contrasts, as if they were coded numerically.

```
> crabs.pois <- glm(satellites ~ ., data = CrabSatellites,
+                                   family = poisson)
> summary(crabs.pois)


Call:
glm(formula = satellites ~ ., family = poisson, data = CrabSatellites)

Deviance Residuals:
   Min      1Q  Median      3Q     Max
-3.029  -1.863  -0.599   0.933   4.945

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  -0.7057     0.9344   -0.76   0.4501
color.L      -0.4120     0.1567   -2.63   0.0085 **
color.Q       0.1237     0.1231    1.00   0.3150
color.C       0.0481     0.0914    0.53   0.5983
spine.L       0.0618     0.0848    0.73   0.4660
spine.Q       0.1585     0.1609    0.99   0.3244
width         0.0165     0.0489    0.34   0.7358
```

```
weight         0.4971     0.1663    2.99   0.0028 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 632.79  on 172  degrees of freedom
Residual deviance: 549.56  on 165  degrees of freedom
AIC: 920.9

Number of Fisher Scoring iterations: 6
```

The Wald tests for the coefficients show that only the linear effect of color and the effect of width are significant. Effect plots, in Figure 11.7, show the nature of these effects—lighter colored females attract more satellites, as do wider and heavier females.

```
> plot(allEffects(crabs.pois), main = "")
```

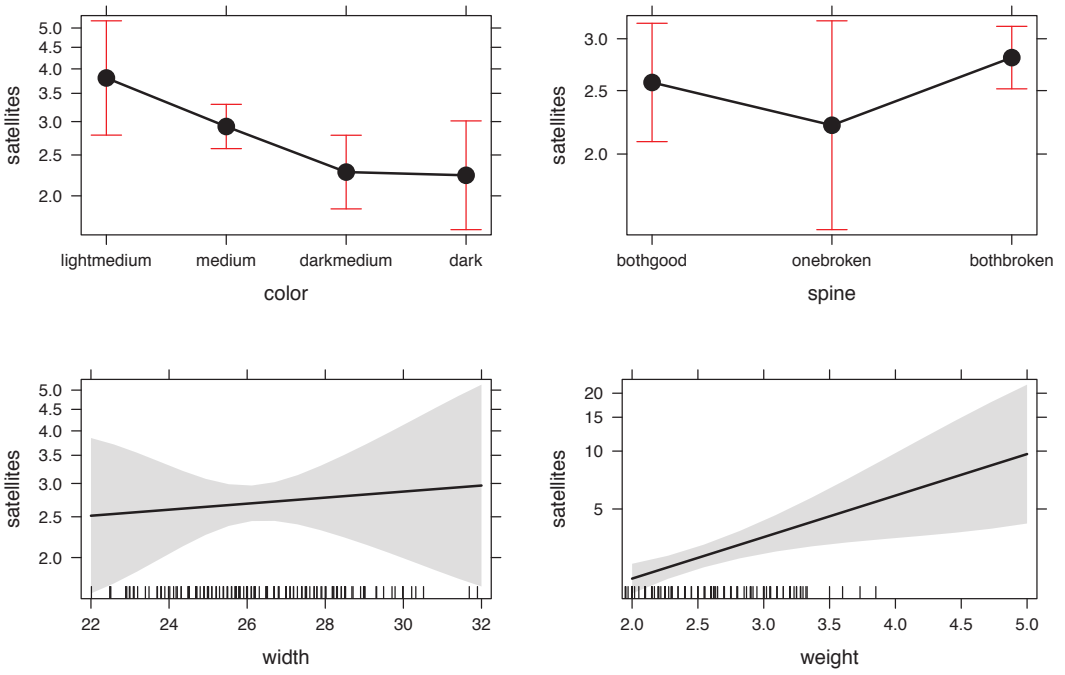


Figure 11.7: Effect plots for the predictors in the Poisson regression model for the CrabSatellites data.

A simpler model can be constructed using `color` as a numeric variable, and either width or weight to represent female size. We choose weight here.[9]

```
> CrabSatellites1 <- transform(CrabSatellites, color = as.numeric(color))
>
> crabs.pois1 <- glm(satellites ~ weight + color, data = CrabSatellites1,
```

[9]Agresti (2013, Section 4.3) and others who have analyzed this example uses carapace width as the main quantitative predictor, possibly because width might be more biologically salient to the single males than weight. This is a case where two highly correlated predictors are each strongly related to the outcome, yet partial tests (controlling for all others) may prefer one over the other.

```
+                     family = poisson)
> summary(crabs.pois1)


Call:
glm(formula = satellites ~ weight + color, family = poisson,
    data = CrabSatellites1)

Deviance Residuals:
   Min      1Q  Median      3Q     Max
-2.978  -1.916  -0.547   0.918   4.834

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)   0.0888     0.2544    0.35    0.727
weight        0.5458     0.0675    8.09    6e-16 ***
color        -0.1728     0.0615   -2.81    0.005 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 632.79  on 172  degrees of freedom
Residual deviance: 552.77  on 170  degrees of freedom
AIC: 914.1

Number of Fisher Scoring iterations: 6
```

From the statistical and graphical analysis so far, the answer to the question posed in this example is clear: unattached male horseshoe crabs prefer light-colored, big fat mamas!

Yet neither of these models fit well, as can be seen from their residual deviances and likelihood-ratio tests.

```
> LRstats(crabs.pois, crabs.pois1)

Likelihood summary table:
            AIC BIC LR Chisq  Df Pr(>Chisq)
crabs.pois  921 946      550 165     <2e-16 ***
crabs.pois1 914 924      553 170     <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Perhaps there is something else to be learned here.

△

11.3 Models for overdispersed count data

In practice, the Poisson model is often very useful for describing the relationship between the mean μ_i and the linear predictors, but typically underestimates the variance in the data. The consequence is that the Poisson standard errors are too small, rendering the Wald tests of coefficients, $z_j = \widehat{\beta}_j / \mathrm{se}(\widehat{\beta}_j)$ (and other hypothesis test statistics) too large, and thus overly liberal.

In applications of the GLM, overdispersion is usually assessed by the likelihood-ratio test of the deviance (or the Pearson statistic) given in Section 11.1.3, but there is a subtle problem here. Lack of fit in a GLM for count data can result either from a mis-specified model for the systematic component (omitted or unmeasured predictors, nonlinear relations, etc.) or from failure of the Poisson mean = variance assumption. Thus, use of these methods requires some high degree of confidence that the systematic part of the model has been correctly specified, so that any lack of fit can be attributed to overdispersion.

One way of dealing with this is to base inference on so-called *sandwich* covariance estimators that are robust against some types of model mis-specification. In R, this is provided by the

`sandwich()` function in the sandwich (Lumley and Zeileis, 2015) package, and can be used with `coeftest(model, vcov = sandwich)` to give overdispersion-corrected hypothesis tests[10] (Zeileis, 2004, 2006). Alternatively, the Poisson model variance assumption can be relaxed in the quasi-Poisson model and the negative-binomial model as discussed below.

11.3.1 The quasi-Poisson model

One obvious solution to the problem of overdispersion for count data is the relaxed assumption that the conditional variance is merely *proportional* to the mean,

$$\mathcal{V}(y_i|\eta_i) = \phi\mu_i \; .$$

Overdispersion is the common case of $\phi > 1$, implying that the conditional variance increases faster than the mean, but the opposite case of underdispersion, $\phi < 1$, is also possible, though relatively rare in practice. This strategy entails estimating the dispersion parameter ϕ from the data, and gives the ***quasi-Poisson model*** for count data.

One possible estimate is the residual deviance divided by degrees of freedom. However, it is more common to use the Pearson statistic, giving a method-of-moments estimate with improved statistical properties:

$$\widehat{\phi} = \frac{X^2_P}{n-p} = \sum_{i=1}^{n} \frac{(y_i - \widehat{\mu}_i)^2}{\widehat{\mu}_i} \, /(n-p) \;\; .$$

It turns out that this model gives the same coefficient estimates as the standard Poisson GLM, but inference is adjusted for over/under dispersion. In particular, following Eqn. (11.1), the standard errors of the model coefficients are multiplied by $\widehat{\phi}^{1/2}$ and so are inflated when overdispersion is present. In R, the quasi-Poisson model with this estimated dispersion parameter is fitted with the `glm()` function, by setting `family=quasipoisson`.

EXAMPLE 11.3: Publications of PhD candidates

For the *PhdPubs* data, the deviance and Pearson estimates of dispersion ϕ can be calculated using the results of the Poisson model saved in the `phd.pois` object. The Pearson estimate, 1.83, indicates that standard errors of coefficients in this model should be multiplied by $\sqrt{1.83} = 1.35$, a 35% increase, to correct for overdispersion.

```
> with(phd.pois, deviance / df.residual)

[1] 1.7971

> sum(residuals(phd.pois, type = "pearson")^2) / phd.pois$df.residual

[1] 1.8304
```

The quasi-Poisson model is then fitted using `glm()` as:

```
> phd.qpois <- glm(articles ~ ., data = PhdPubs, family = quasipoisson)
```

For use in other computation, the dispersion parameter estimate $\widehat{\phi}$ can be obtained as the `dispersion` value of the `summary()` method for a quasi-Poisson model.

[10] More precisely, given that the mean function of the model is correctly specified, the sandwich standard errors guard against misspecifications of the remaining likelihood, including overdispersion and heteroskedasticity.

```
> (phi <- summary(phd.qpois)$dispersion)

[1] 1.8304
```

Note that this value can be compared to the variance/mean ratio of 2.91 calculated for the marginal distribution in Example 11.1; there is considerable improvement taking the predictors into account.

△

11.3.2 The negative-binomial model

The negative-binomial (NB) model for count data was introduced in Section 3.2.3 as a different generalization of the Poisson model that allows for overdispersion. In the context of the GLM, this can be developed as the extended form where the distribution of $y_i \,|\, \boldsymbol{x}_i$ where the mean μ_i for fixed $\boldsymbol{x}_i$ can vary across observations i according to a gamma distribution with mean μ_i and a constant shape parameter, θ, reflecting the additional variation due to heterogeneity.

For a fixed value of θ, the negative-binomial is another special case of the GLM. The expected value of the response is again $\mathcal{E}(y_i) = \mu_i$, but the variance function is $\mathcal{V}(y_i) = \mu_i + \mu_i^2/\theta$, so the variance of y increases more rapidly than that of the Poisson distribution. Some authors (e.g., Agresti (2013), Hilbe (2014)) prefer to parameterize the variance function in terms of $\alpha = 1/\theta$, giving

$$\mathcal{V}(y_i) = \mu_i + \mu_i^2/\theta = \mu_i + \alpha\mu_i^2 \,,$$

so that α is a kind of dispersion parameter. Note that as $\alpha \to 0$, $\mathcal{V}(y_i) \to \mu_i$ and the negative-binomial converges to the Poisson.

The MASS package provides the family function `negative.binomial(theta)` that can be used directly with `glm()` provided that the argument `theta` is specified. One example would be the related geometric distribution (Section 3.2.4) that is the special case of $\theta = 1$. This can be fitted in R by setting `family=negative.binomial(theta=1)` in the call to `glm()`.

Most often, θ is unknown and must be estimated from the data. In this case, the negative-binomial model is not a special case of the GLM, but it is possible to obtain maximum likelihood estimates of both $\boldsymbol{\beta}$ and θ, by iteratively estimating $\boldsymbol{\beta}$ for fixed θ and vice-versa. This method is implemented in the `glm.nb()` in the package MASS.

EXAMPLE 11.4: Mating of horseshoe crabs

For example, for the *CrabSatellites* data, we can fit the general negative-binomial model with θ free.

```
> library(MASS)
> crabs.nbin <- glm.nb(satellites ~ weight + color,
+                      data = CrabSatellites1)
> crabs.nbin$theta

[1] 0.95562
```

The estimated value $\widehat{\theta}$ returned by `glm.nb()` is not very far from 1. Hence, we might also consider fixing $\theta = 1$, as illustrated below.

```
> crabs.nbin1 <- glm(satellites ~ weight + color, data = CrabSatellites1,
+                    family = negative.binomial(1))
```

△

11.3.3 Visualizing the mean–variance relation

The quasi-Poisson and negative-binomial models have different variance functions, and one way to visualize which provides a better fit to the data is to group the data according to the fitted value of the linear predictor, calculate the mean and variance for each group, and then plot the variances against the means. A smoothed curve will then approximate the *empirical* mean–variance relationship. To this, we can add curves showing the mean–variance function implied by various models.[11]

EXAMPLE 11.5: Publications of PhD candidates

For the `PhdPubs` data, the fitted values are obtained with `fitted()` for the Poisson and negative binomial models. Either set can be used to categorize the observations into groups for the purpose of calculating means and variances of the response.

```
> fit.pois <- fitted(phd.pois, type = "response")
> fit.nbin <- fitted(phd.nbin, type = "response")
```

Here we use a simpler version of the `cutfac()` function to group a numeric variable into quantile-based groups. `cutq()` also uses deciles by default, and just uses simple integer values for the factor labels.

```
> cutq <- function(x, q = 10) {
+     quantile <- cut(x, breaks = quantile(x, probs = (0 : q) / q),
+         include.lowest = TRUE, labels = 1 : q)
+     quantile
+ }
```

Using this, we create a variable `group` giving 20 quantile groups of the fitted values, and then use `aggregate()` to find the mean and variance of the number of articles in each group.

```
> group <- cutq(fit.nbin, q = 20)
> qdat <- aggregate(PhdPubs$articles,
+           list(group),
+           FUN = function(x) c(mean = mean(x), var = var(x)))
> qdat <- data.frame(qdat$x)
> qdat <- qdat[order(qdat$mean),]
```

We can then calculate the theoretical variances implied by the quasi-Poisson and negative-binomial models:

```
> phi <- summary(phd.qpois)$dispersion
> qdat$qvar <- phi * qdat$mean
> qdat$nbvar <- qdat$mean + (qdat$mean^2) / phd.nbin$theta
> head(qdat)

     mean     var   qvar  nbvar
1 0.61224 0.78401 1.1206 0.7776
2 1.14894 1.78168 2.1030 1.7312
8 1.24444 2.46162 2.2778 1.9276
4 1.26087 1.70821 2.3079 1.9622
6 1.27273 1.83087 2.3296 1.9873
7 1.29787 4.34413 2.3756 2.0409
```

The plot, shown in Figure 11.8, then simply plots the points and uses `lines()` to plot the model-implied variances.

[11] This idea and the example that follows was suggested by Germán Rodrigues in a Stata example given at `http://data.princeton.edu/wws509/stata/overdispersion.html`.

```
> with(qdat, {
+   plot(var ~ mean, xlab = "Mean number of articles", ylab = "Variance",
+        pch = 16, cex = 1.2, cex.lab = 1.2)
+   abline(h = mean(PhdPubs$articles), col = gray(.40), lty = "dotted")
+   lines(mean, qvar, col = "red", lwd = 2)
+   lines(mean, nbvar, col = "blue", lwd = 2)
+   lines(lowess(mean, var), lwd = 2, lty = "dashed")
+   text(3, mean(PhdPubs$articles), "Poisson", col = gray(.40))
+   text(3, 5, "quasi-Poisson", col = "red")
+   text(3, 6.7, "negbin", col = "blue")
+   text(3, 8.5, "lowess")
+ })
```

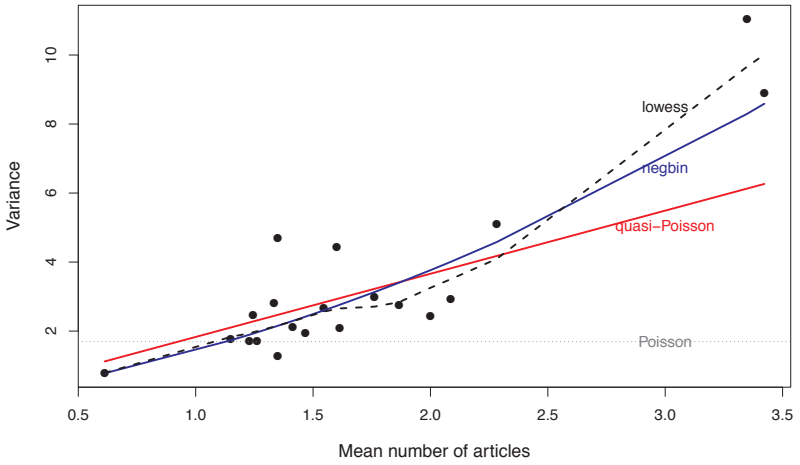


Figure 11.8: Mean–variance functions for the PhdPubs data. Points show the observed means and variances for 20 quantile groups based on the fitted values in the negative-binomial model. The labeled lines and curves show the variance functions implied by various models.

We can see from this plot that the variances implied by the quasi-Poisson and negative-binomial models are in reasonable accord with the data and with each other up to a mean of about 2.5. They diverge substantially at the upper end, for the 20–30% of the most productive candidates, where the quadratic variance function of the negative-binomial provides a better fit.

Finally, we can also compare the standard errors of coefficients for the various methods designed to correct for overdispersion. These are extracted as the diagonal elements of the `vcov()` and `sandwich()` methods from the model objects.

```
> library(sandwich)
> phd.SE <- sqrt(cbind(
+   pois = diag(vcov(phd.pois)),
+   sand = diag(sandwich(phd.pois)),
+   qpois = diag(vcov(phd.qpois)),
+   nbin = diag(vcov(phd.nbin))))
> round(phd.SE, 4)

             pois   sand  qpois   nbin
(Intercept) 0.0996 0.1382 0.1348 0.1327
female1     0.0546 0.0714 0.0738 0.0726
married1    0.0613 0.0823 0.0829 0.0819
kid5        0.0401 0.0560 0.0543 0.0528
```

```
phdprestige 0.0253 0.0392 0.0342 0.0343
mentor      0.0020 0.0039 0.0027 0.0032
```

For this example, the sandwich, quasi-Poisson, and negative-binomial methods give similar results, all about 40% larger on average than those from the Poisson model. △

11.3.4 Testing overdispersion

The forms of overdispersion seen in these examples and in Figure 11.8 give rise to a statistical test (Cameron and Trivedi 1990; Cameron and Trivedi 1998, Section 3.4) for the null hypothesis of Poisson variation, $H_0 : \mathcal{V}(y) = \mu$, against an alternative that the variance has a particular form depending on the mean,

$$\mathcal{V}(y) = \mu + \alpha \times f(\mu) \, ,$$

where $f(\mu)$ is a given transformation function of the mean.

Overdispersion corresponds to $\alpha > 0$ and underdispersion to $\alpha < 0$. The coefficient α can be estimated by an auxiliary OLS regression (without an intercept), i.e., of the form

```
lm(var ~ -1 + f(mean))
```

and tested with the corresponding t (or z) statistic, which is asymptotically standard normal under the null hypothesis.

Common specifications of the transformation function are $f(\mu) = \mu$ and $f(\mu) = \mu^2$. The first corresponds to an NB model with a linear variance function (called NB1 by various authors) or a quasi-Poisson model with dispersion parameter ϕ, i.e.,

$$\mathcal{V}(y) = (1 + \alpha)\mu = \phi\mu \, .$$

The second is the more traditional form with quadratic variance function described in Section 11.3.2 (called NB2 by some authors).

These tests are carried out using the `dispersiontest()` function in the AER (Kleiber and Zeileis, 2015) package, the companion software of Kleiber and Zeileis (2008). The first argument is a Poisson GLM model; the second specifies the alternative hypothesis, either as an integer power of μ or a function of the mean.

```
> library(AER)
> dispersiontest(phd.pois)


Overdispersion test

data:  phd.pois
z = 5.73, p-value = 4.9e-09
alternative hypothesis: true dispersion is greater than 1
sample estimates:
dispersion
    1.8259

> dispersiontest(phd.pois, 2)


Overdispersion test

data:  phd.pois
z = 6.46, p-value = 5.3e-11
alternative hypothesis: true alpha is greater than 0
sample estimates:
  alpha
0.50877
```

These tests use a specified alternative hypothesis, so there is no way to compare directly which of the NB1 or NB2 models is better or worse, except by using methods such as AIC or BIC described in Section 11.1.4.

11.3.5 Visualizing goodness-of-fit

Even with correction for overdispersion, goodness-of-fit tests provide only an overall summary of model fit. Some specialized tests for particular forms of overdispersion are also available (e.g., see Cameron and Trivedi (1998, Chapter 5)), but these only identify general problems and cannot provide detailed indications of the possible source of these problems.

In Chapter 3, we illustrated the use of rootograms for visualizing goodness-of-fit to a wide variety of discrete distributions using the `plot()` method for class "goodfit" objects with the vcd package. However, those methods were developed for one-way discrete distributions without explanatory variables.

Kleiber and Zeileis (2014) have generalized this idea to the wider class of GLM-related count regression models considered here. The countreg package provides a new implementation of `rootogram()` with methods for all of these models (and others not mentioned). We illustrate these plots for the models considered to this point, and then extend this use for models allowing for excess zero counts in Section 11.4.

EXAMPLE 11.6: Publications of PhD candidates

For the *PhdPubs* data, Figure 11.9 shows hanging rootograms for the Poisson and negative-binomial models produced using `countreg::rootogram`[12] on the fitted model objects. We are looking both for general patterns of under/over fit, as well as counts that stand out as poorly fitted against the background.

```
> library(countreg)
> countreg::rootogram(phd.pois, max = 12,
+                     main = "PhDPubs: Poisson")
> countreg::rootogram(phd.nbin, max = 12,
+                     main = "PhDPubs: Negative-Binomial")
```

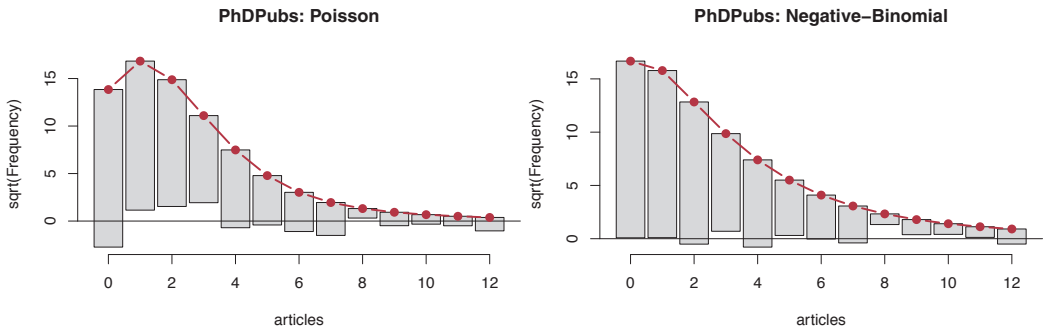


Figure 11.9: Hanging rootograms for the PhdPubs data.

The Poisson model shows a systematic, wave-like pattern with excess zeros, too few observed frequencies for counts of 1–3, but generally greater frequencies for counts of 4 or more. The negative-binomial model clearly fits much better, though there is a peculiar tendency among the smaller frequencies for 8 or more articles. △

[12] At the time of this writing, `rootogram` in countreg conflicts with the version in vcd, so we qualify the use here with the package name.

EXAMPLE 11.7: Mating of horseshoe crabs

Figure 11.10 shows similar plots for the same two models fit to the number of crab satellites. The fit of the Poisson model clearly reveals the excess of zero male satellites. For the negative-binomial, the rootogram no longer exhibits same wave-like pattern, however, the underfitting of the count for 0 and overfitting for counts 1–2 is characteristic of data with excess zeros.

```
> countreg::rootogram(crabs.pois, max = 15,
+                     main = "CrabSatellites: Poisson")
> countreg::rootogram(crabs.nbin, max = 15,
+                     main = "CrabSatellites: Negative-Binomial")
```

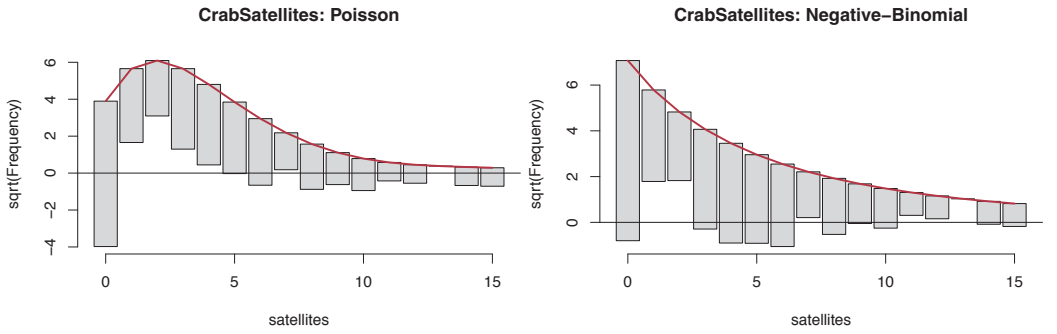


Figure 11.10: Hanging rootograms for the CrabSatellites data.

△

11.4 Models for excess zero counts

In addition to overdispersion, many sets of empirical data exhibit a greater prevalence of zero counts than can be accommodated by the Poisson or negative-binomial models. We saw this in the `PhdPubs` data set, where there were many candidates who had not published at all, and in the `CrabSatellites` data where a large number of females attracted no unattached males. Other examples abound in many different fields: studies of the use of health care services often find that many people never visit a hospital in some time frame; similarly, the distribution of insurance claims often shows large numbers who make no claims (Yip and Yau, 2005) because of under-reporting of small claims, policy deductible provisions, and desire to avoid premium increases.

Beyond simply identifying this as a problem of lack-of-fit, understanding the reasons for excess zero counts can make a contribution to a more complete explanation of the phenomenon of interest, and this requires both new statistical models and visualization techniques illustrated in this section.

In the first example, Long (1997) argued that the PhD candidates might fall into two distinct groups: "publishers" (perhaps striving for an academic career) and "non-publishers" (seeking other career paths). Of the 275 observations having `articles==0`, some might not have published due to chance or unmeasured factors. One reasonable form of explanation is that the observed zero counts reflect a mixture of the two latent classes—those who simply have not yet published and those who will likely never publish. A statistical formulation of this idea leads to the class of *zero-inflated* models described below.

A different form of explanation is that there may be some special circumstance or "hurdle" required to achieve a positive count, like publishing the master's thesis (such as being driven internally by a personality trait or externally by pressure from a mentor). This idea leads to the class of *hurdle*

models that entertain and fit (simultaneously) two separate models: one for the occurrence of the zero counts, and one for the positive counts. These two approaches are illustrated in Figure 11.11

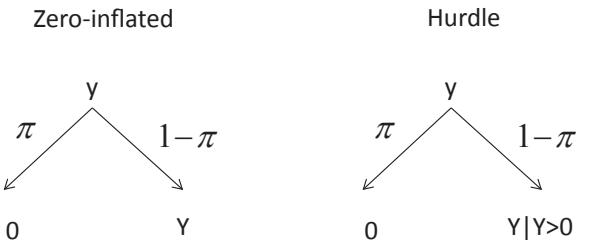


Figure 11.11: Models for excess zeros. The observed response y is derived from a latent or parent distribution for Y yielding zero counts with probability π.

11.4.1 Zero-inflated models

Zero-inflated models, introduced by Lambert (1992) as the ***zero-inflated Poisson*** (ZIP) model, provide an attractive solution to the problem of dealing with an overabundance of zero counts. It postulates that the observed counts arise from a mixture of two latent classes of observations: some structural zeros for whom y_i will always be 0, and the rest, sometimes giving random zeros. The ZIP model is comprised of two components:

- A model for the binary event of membership in the unobserved (latent) class of those for whom the count is necessarily zero (e.g., "non-publishers"). This is typically taken as a logistic regression for the probability π_i that observation i is in this class, with predictors $\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots, \boldsymbol{z}_q$, giving
$$\text{logit}(\pi_i) = \boldsymbol{z}_i^\mathsf{T}\boldsymbol{\gamma} = \gamma_0 + \gamma_1 z_{i1} + \gamma_2 z_{i2} + \cdots + \gamma_q z_{iq} \,. \tag{11.7}$$
- A Poisson model for the other class (e.g., "publishers"), for whom the observed count may be 0 or positive. This model typically uses the usual log link to predict the mean, using predictors $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_p$, so
$$\log_e \, \mu(\boldsymbol{x}_i) = \boldsymbol{x}_i^\mathsf{T}\boldsymbol{\beta} = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_q x_{ip} \,. \tag{11.8}$$

In application, it is permissible and not uncommon to use the same set of predictors $x = z$ in both submodels, but the notation indicates that this is not required. Some simple special cases arise when the model for the always-zero latent class is an intercept-only model, $\text{logit}(\pi_i) = \gamma_0$, implying the same probability for all individuals, and (less commonly) when the Poisson mean model is intercept-only with no predictors, but there might be excess zero counts.

With this setup, one can show that the probability of observing counts of $y_i = 0$ and $y_i > 0$ are

$$\begin{aligned} \Pr(y_i = 0 \,|\, \boldsymbol{x}, \boldsymbol{z}) &= \pi_i + (1 - \pi_i) e^{-\mu_i} \\ \Pr(y_i \,|\, \boldsymbol{x}, \boldsymbol{z}) &= (1 - \pi_i) \times \left[\frac{{\mu_i}^{y_i} e^{-\mu_i}}{y_i!} \right], \qquad y_i \geq 0 \,, \end{aligned} \tag{11.9}$$

where the term in brackets in the second equation is the Poisson probability $\Pr(y = y_i)$ with rate parameter $\text{Pois}(\mu_i)$. In these equations, $\pi_i = \text{logit}^{-1}(\boldsymbol{z}_i^\mathsf{T}\boldsymbol{\gamma})$ depends on the $\boldsymbol{z}$ through Eqn. (11.7), and $\mu_i = \exp(\boldsymbol{x}^\mathsf{T}\boldsymbol{\beta})$ depends on the $\boldsymbol{x}$ through Eqn. (11.8).

The conditional expectation and variance of y_i then have the forms

$$\begin{aligned} \mathcal{E}(y_i) &= (1 - \pi_i)\, \mu_i \\ \mathcal{V}(y_i) &= (1 - \pi_i)\, \mu_i (1 + \mu_i \pi_i) \,. \end{aligned}$$

Thus, when $\pi_i > 0$, the mean of y is always less than μ_i, and the variance of y is greater than its mean by a dispersion factor of $(1 + \mu_i \pi_i)$.

There is nothing special about the use of the Poisson distribution here. The model for the count variable could also be taken as the negative-binomial, giving a *zero-inflated negative-binomial* (ZINB) model using $\text{NBin}(\mu, \theta)$ or a *zero-inflated geometric* model using $\text{NBin}(\mu, \theta = 1)$.

EXAMPLE 11.8: Simulating zero-inflated data

A simple way of understanding the effects of zero-inflation on count data is to simulate data from their distribution and plot it. For the standard Poisson and negative-binomial, random values can be generated using `rpois()` and `rnegbin()` (in MASS), respectively. Their zero-inflated counterparts are implemented in the VGAM package as `rzipois()` and `rzinegbin()`.

To illustrate this use, we generate two random data sets using `rzipois()` having constant mean $\mu = 3$. The first is a standard Poisson ($\pi = 0$), while the second has a constant probability $\pi = 0.3$ of an excess zero.

```
> library(VGAM)
> set.seed(1234)
> data1 <- rzipois(200, 3, 0)
> data2 <- rzipois(200, 3, .3)
```

Barplots of the frequencies in these data sets are shown in Figure 11.12. The sample mean in `data1` is 2.925, quite close to $\mu = 3$. In the zero-inflated `data2`, the mean is only 2.25 due to the excess zeros.

```
> tdata1 <- table(data1)
> barplot(tdata1, xlab = "Count", ylab = "Frequency",
+         main = "Poisson(3)")
> tdata2 <- table(data2)
> barplot(tdata2, xlab = "Count", ylab = "Frequency",
+         main = expression("ZI Poisson(3, " * pi * "= .3)"))
```

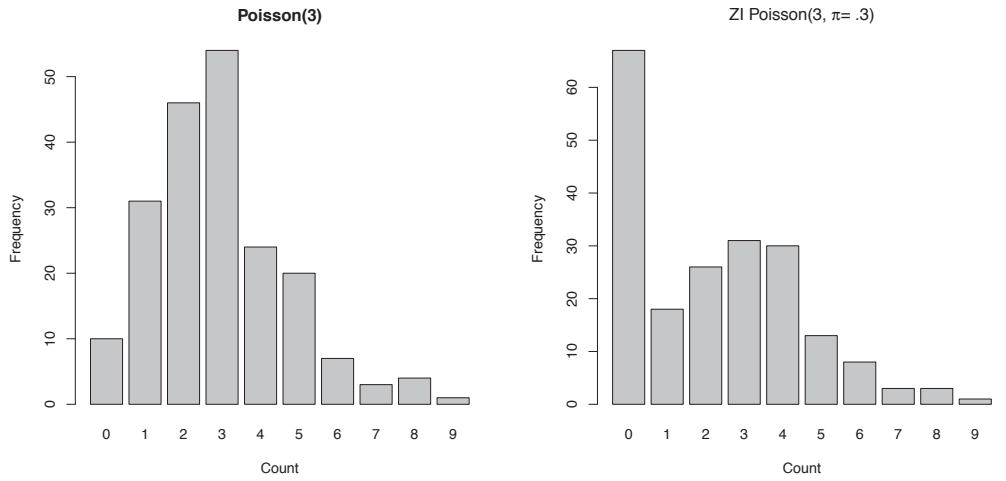


Figure 11.12: Bar plots of simulated data from Poisson and zero-inflated Poisson distributions.

△

There are several packages in R capable of fitting zero-inflated models. The most mature and

complete of these is `zeroinfl()` in the countreg package (a successor to the pscl package). The function `zeroinfl()` is modeled after `glm()`, but provides an extended syntax for the model formula.

If the `formula` argument is supplied in the form `y ~ x1 + x2 + ...`, it not only describes the count regression of y on $x_1, x_2, \ldots$, but also implies that the *same* set of regressors, $z_j = x_j$, is used for the zero count binary submodel. The extended syntax uses the notation `y ~ x1 + x2 + ... | z1 + z2 + ...` to specify the x variables separately, conditional on (`|`) the always-zero count model `y ~ z1 + z2 + ...`. The model for the not-always-zero class can be specified using the `dist` argument, with possible values `"poisson"`, `"negbin"`, and `"geometric"`.

11.4.2 Hurdle models

A different class of models capable of accounting for excess zero counts is the ***hurdle model*** (also called the ***zero-altered model***) proposed initially by Cragg (1971) and developed further by Mullahy (1986). This model also uses a separate logistic regression submodel to distinguish counts of $y = 0$ from larger counts, $y > 0$. The submodel for the positive counts is expressed as a (left) *truncated* Poisson or negative-binomial model, excluding the zero counts. As an example, consider a study of behavioral health in which one outcome is the number of cigarettes smoked in one month. All the zero counts will come from non-smokers and smokers will nearly always smoke a positive number.

This differs from the set of ZIP models in that classes of $y = 0$ and $y > 0$ are now considered fully observed, rather than latent. Conceptually, there is one process and submodel accounting for the zero counts and a separate process accounting for the positive counts, once the "hurdle" of $y = 0$ has been passed. In other words, for ZIP models, the first process generates only extra zeros beyond those of the regular Poisson distribution. For hurdle models, the first process generates all of the zeros. The probability equations corresponding to Eqn. (11.9) are:

$$\begin{aligned} \Pr(y_i = 0 \,|\, \boldsymbol{x}, \boldsymbol{z}) &= \pi_i \\ \Pr(y_i \,|\, \boldsymbol{x}, \boldsymbol{z}) &= \frac{(1-\pi_i)}{1-e^{-\mu_i}} \times \left[\frac{{\mu_i}^{y_i} e^{-\mu_i}}{y_i!} \right], \qquad y_i \geq 0\,. \end{aligned} \tag{11.10}$$

The hurdle model can be fitted in R using the `hurdle()` function from the countreg package. The syntax for the model formula is the same extended form provided by `zeroinfl()`, where `y ~ x1 + x2` uses the same regressors for the zero and positive count submodels, while `y ~ x1 + x2 | z1 + z2` uses `y ~ z1 + z2` for the zero hurdle model. Similarly, the count distribution can be given as `"poisson"`, `"negbin"`, or `"geometric"` with the `dist` argument. For `hurdle()`, the distribution for zero model can be specified with a `zero.dist` argument. The default is `"binomial"` (with a logit `link`), but other right-censored distributions can also be specified.

11.4.3 Visualizing zero counts

Both the zero-inflated and hurdle models treat the zero counts $y = 0$ specially with separate submodels, so the binary event of $y = 0$ vs. $y > 0$ can be visualized using any of the techniques illustrated in Chapter 7. See Section 7.2.3, Section 7.3.1, and Section 7.3.2 for some examples that plot both the binary observations and a model summary or smoothed curve to show the relationships with one or more regressors. To apply these ideas in the current context, simply define or plot a logical variable corresponding to the expression `y==0`, giving values of `TRUE` or `FALSE`.

A different, and simpler idea is illustrated here using what is called a ***spineplot*** Hummel (1996) when a predictor x is a discrete factor or ***spinogram*** when x is continuous. Both are forms of mosaic plots with special formatting of spacing and shading, and in this context they plot $\Pr(y = 0|x)$ against $\Pr(x)$; when x is numerical, it is first made discrete, as in a histogram.

Then, in the spine plot or spinogram, the widths of the bars correspond to the relative frequencies of x and heights of the bars correspond to the conditional relative frequencies of $y = 0$ in every x group. In R, spine plots are implemented in the function `spineplot()`; however, this is what you get by default if you use `plot(y==0 ~ x)` to plot the binary factor against any regressor x.

A related graphical method is the ***conditional density plot*** (Hofmann and Theus, 2005). The conditional probabilities $\Pr(y = 0|x)$ are derived using a smoothing approach (via `density()`) over x rather than by making x discrete. These plots are provided by `cdplot()` in the graphics package and a similar `cd_plot()` in vcd. The smoothing method for the density estimate is controlled by a `bw` (bandwidth) method and other arguments.

EXAMPLE 11.9: Mating of horseshoe crabs

For the *CrabSatellites* data, we can examine the relationship of the zero counts (females who attract no unattached male satellites) to the predictors using spinograms or conditional density plots. Here, we consider `weight` and `color` (treated numerically) as predictors.

Spinograms for the occurrence of zero satellites against `weight` and `color` are shown in Figure 11.13, where we have used quantiles of those distributions to define the breaks on the horizontal axis. Using `ylevels=2:1` reverses the order of the vertical categories. You can easily see that the zeros decrease steadily with weight and increase with darkness.

```
> plot(factor(satellites == 0) ~ weight, data = CrabSatellites,
+       breaks = quantile(weight, probs = seq(0,1,.2)), ylevels = 2:1,
+       ylab = "No satellites")
> plot(factor(satellites == 0) ~ color, data = CrabSatellites,
+       breaks = quantile(color, probs = seq(0,1,.33)),  ylevels = 2:1,
+       ylab = "No satellites")
```

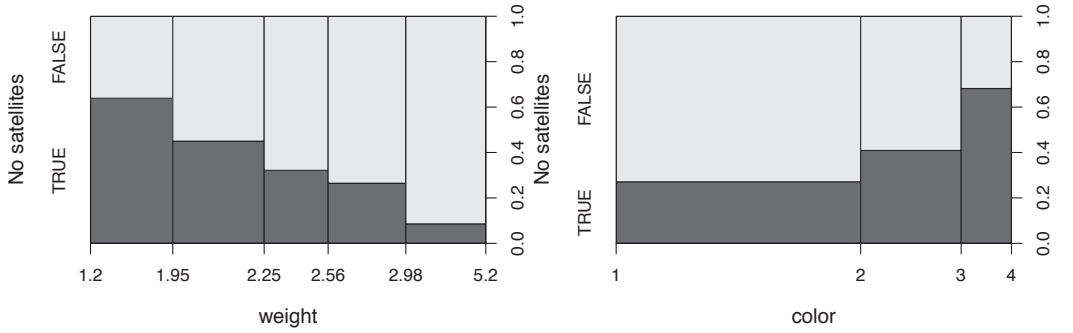


Figure 11.13: Spinograms for the CrabSatellites data. The variables weight (left) and color (right) have been made discrete using quantiles of their distributions.

Similar plots in the form of conditional density plots are shown in Figure 11.14, with a similar interpretation.

```
> cdplot(factor(satellites == 0) ~ weight, data = CrabSatellites,
+         ylevels = 2:1, ylab = "No satellites")
> cdplot(factor(satellites == 0) ~ color, data = CrabSatellites,
+         ylevels = 2:1, , ylab = "No satellites")
```

△

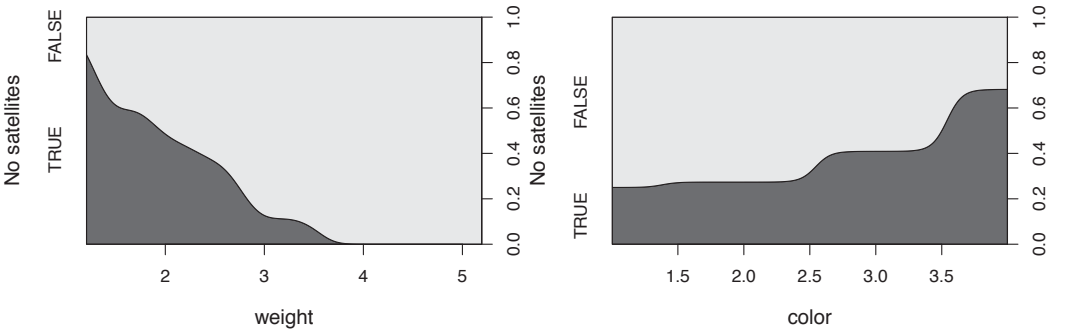


Figure 11.14: Conditional density plots for the CrabSatellites data. The region shaded below shows the conditional probability density estimate for a count of zero.

11.5 Case studies

In this section, we introduce two extended examples, designed to illustrate aspects of exploratory analysis, visualization, model fitting, and interpretation for count data GLMs. The first (Section 11.5.1) concerns another well-known data set from ethology, where (a) excess zeros require special treatment, (b) the occurrence of zero counts has substantive meaning, and (c) an interaction between two factors is important.

The second case study (Section 11.5.2) uses a larger, also well-known data set from health economics, with more predictors and more potential interactions. The emphasis shifts here from fitting and comparing models with different distributional forms and link functions to selecting terms for an adequate descriptive and explanatory model. Another feature of these examples is that the relatively large sample size in this data supports a wider range of model complexity than is available in smaller samples.

11.5.1 Cod parasites

The cod fishery is extremely important to the economy of Norway, so anything that affects the health of the cod population and its ecosystem can have severe consequences. The red king crab *Paralithodes camtschaticus* was deliberately introduced by Russian scientists to the Barents Sea in the 1960s and 1970s from its native area in the North Pacific. The carapace of these crabs is used by the leech *Johanssonia arctica* to deposit its eggs. This leech in turn is a vector for the blood parasite *Trypanosoma murmanensis* that can infect marine fish, including cod.

Hemmingsen et al. (2005) examined cod for trypanosome infections during annual cruises along the coast of Finnmark in North Norway over three successive years and in four different areas (A1: Sørøya; A2: Magerøya; A3: Tanafjord; A4: Varangerfjord). They show that trypanosome infections are strongest in the area Varangerfjord where the density of red king crabs is highest. Thus, there is evidence that the introduction of the foreign red king crabs had an indirect detrimental effect on the health of the native cod population. This situation stands out because it is not an introduced *parasite* that is dangerous for a native host, but rather an introduced *host* that promotes transmission of two endemic parasites. They call the connections among these factors "an unholy trinity."[13]

[13] The four areas A1–A4 are arranged from east to west, with Varangerfjord (A4) closest to the Russian Kola Peninsula where the red king crabs initially migrated. A more specific test of the "Russian hypothesis" could be developed by treating area as an ordered factor and testing the linear component. We leave this analysis to an exercise for the reader.

EXAMPLE 11.10: Cod parasites

The data from Hemmingsen et al. (2005) is contained in `CodParasites` in the countreg package. It gives the results for 1,254 cod caught by one ship in annual autumn cruises from 1999–2001. The main response variable, `intensity`, records the counted number of *Trypanosoma* parasites found in blood samples from these fish. To distinguish between infected vs. non-infected fish, a secondary response, `prevalence`, is also recorded, corresponding to the expression

```
> CodParasites$prevalence <-
+     ifelse(CodParasites$intensity == 0, "no", "yes")
```

Thus, `intensity` is the basic count response variable, and `prevalence` reflects the zero count that would be assessed in zero-inflated and hurdle models. In substantive terms, in a hurdle model, `prevalence` corresponds to whether a fish is infected or not; once infected, `intensity` gives the degree of infection. In a zero-inflated model, infected could be considered a latent variable; there are extra zeros from non-infected fish, but some infected fish are measured as "normal" zeros.

Hemmingsen et al. (2005) consider only three explanatory predictors: `area`, `year` (both factors) and `length` of the fish.[14] A quick numerical summary of the univariate properties of these variables is shown below. The intensity values are indeed extremely skewed, with a median of 0 and a maximum of 257. However, there are some missing values (`NA`s) among the response variables and a few in the length variable.

```
> data("CodParasites", package = "countreg")
> summary(CodParasites[, c(1 : 4, 7)])

   intensity       prevalence             area        year          length
 Min.   :  0.00   no  :654   soroya       :272   1999:567   Min.   : 17.0
 1st Qu.:  0.00   yes :543   mageroya     :255   2000:230   1st Qu.: 44.0
 Median :  0.00   NA's: 57   tanafjord    :415   2001:457   Median : 54.0
 Mean   :  6.18              varangerfjord:312              Mean   : 53.4
 3rd Qu.:  4.00                                            3rd Qu.: 62.0
 Max.   :257.00                                            Max.   :101.0
 NA's   :57                                                NA's   :6
```

Even better, a quick univariate and bivariate summary of these variables can be shown in a generalized pairs plot (Figure 11.15).

```
> library(vcd)
> library(gpairs)
> gpairs(CodParasites[, c(1 :4, 7)],
+         diag.pars = list(fontsize = 16),
+         mosaic.pars = list(gp = shading_Friendly))
```

In this plot, among the categorical variables, prevalence is strongly associated with area, but also with year. As well, there seems to be an association between area and year, meaning the number of cod samples collected in different areas varied over time. In the univariate plots on the diagonal, intensity stands out as extremely skewed, and the distribution of length appears reasonably symmetric.

Before fitting any models, some more detailed exploratory plots are helpful for understanding the relationship of both prevalence and intensity to the predictors. The general idea is to make separate plots of prevalence and intensity and to try to show both the data and some simple summaries. In their Table 1, Hemmingsen et al. (2005) counted the missing observations as infected and we do the same to get a similar contingency table.

[14] Other potential predictors include weight, sex, age, and developmental stage, as well as the depth at which the fish were caught.

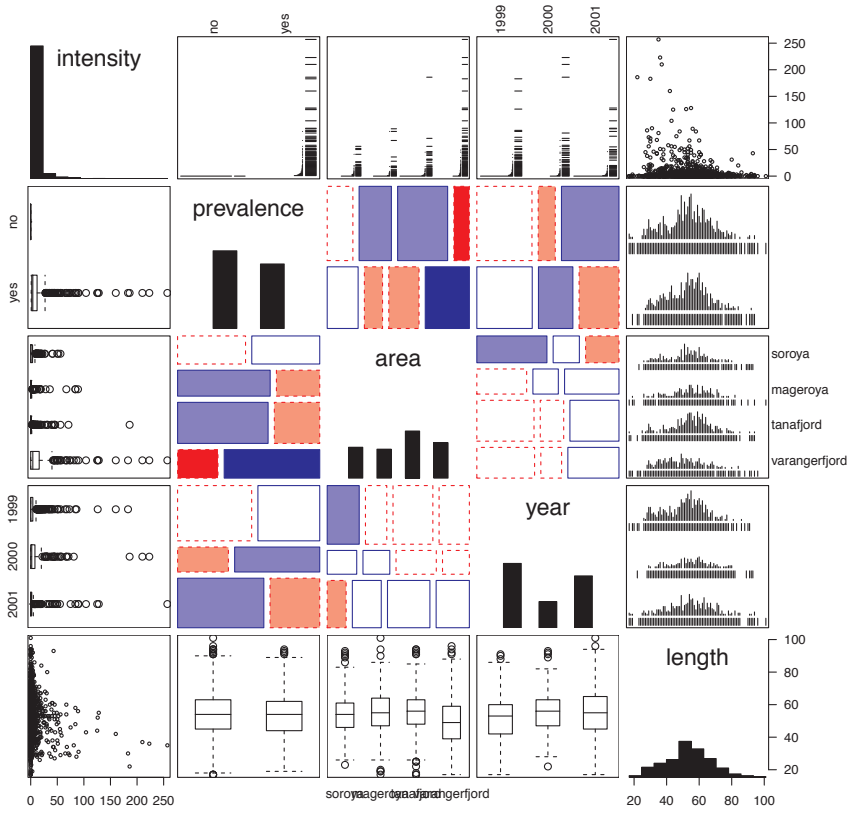


Figure 11.15: Generalized pairs plot for the CodParasites data.

```
> cp.tab <- xtabs(~ area + year + factor(is.na(prevalence) |
+                                        prevalence == "yes"),
+                 data = CodParasites)
> dimnames(cp.tab)[3] <- list(c("No", "Yes"))
> names(dimnames(cp.tab))[3] <- "prevalence"
```

For the factors `area` and `year`, we can visualize prevalence as before (Example 11.9) using spineplots, but, for two (or more) factors, doubledecker and mosaic plots are better because they are more flexible and keep the factors distinct. The doubledecker plot (Figure 11.16) highlights the infected fish, and shows that prevalence is indeed highest in all years in Varangerfjord.

```
> doubledecker(prevalence ~ area + year, data = cp.tab,
+              margins = c(1, 5, 3, 1))
```

A similar plot can be drawn shading the tiles according to a model for the expected counts. It makes sense here to consider the null loglinear model for prevalence as a response, independent of the combinations of area and year. This plot (Figure 11.17) shows further that prevalence differs substantially over the area-year combinations, so we should expect an interaction in the model for zero counts. As well, Varangerfjord stands out as having consistently greater prevalence in all years than expected under this model.

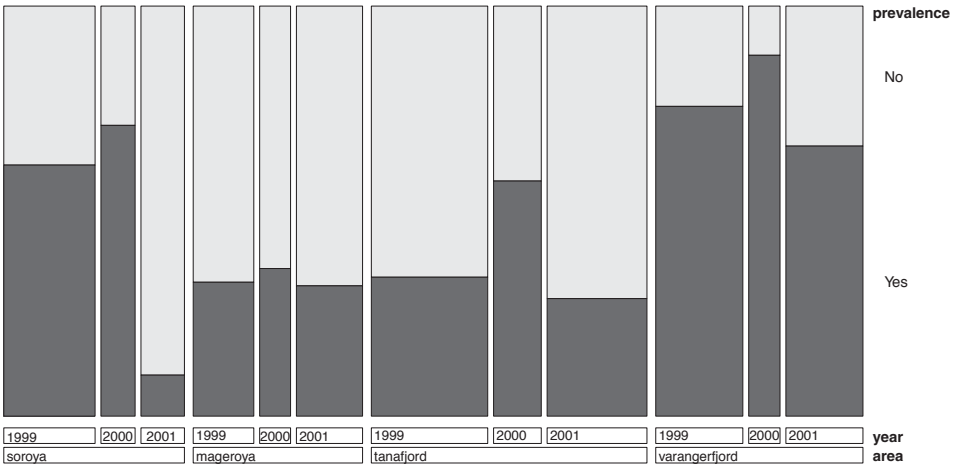


Figure 11.16: Doubledecker plot for prevalence against area and year in the CodParasites data. The cases of infected fish are highlighted.

```
> doubledecker(prevalence ~ area + year, data = cp.tab,
+              gp = shading_hcl, expected = ~ year:area + prevalence,
+              margins = c(1, 5, 3, 1))
```

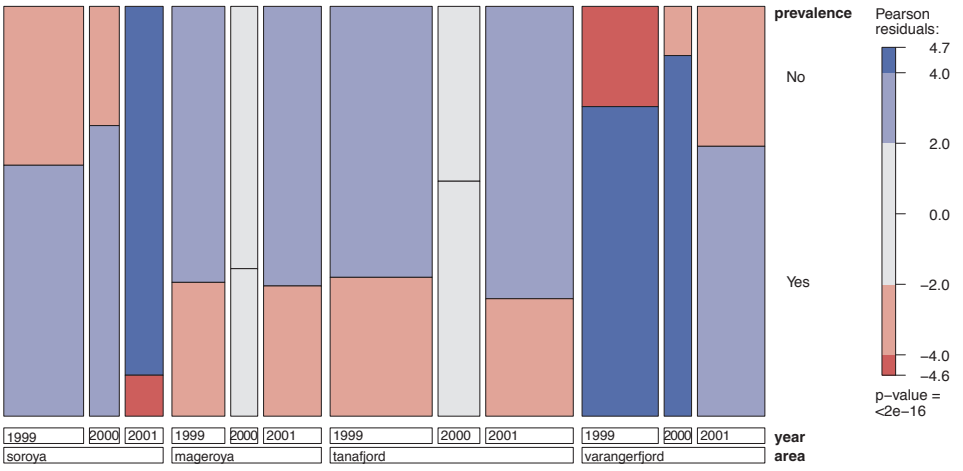


Figure 11.17: Mosaic plot for prevalence against area and year in the CodParasites data, in the doubledecker format. Shading reflects departure from a model in which prevalence is independent of area and year jointly.

The effect of fish `length` on `prevalence` can be most easily seen by treating the factor as a numeric (0/1) variable and smoothing, as shown in Figure 11.18. The loess smoothed curve shows an apparent U-shaped relationship; however, the plotted observations and the confidence bands make clear that there is very little data in the extremes of `length`.

```
> library(ggplot2)
> ggplot(CodParasites, aes(x = length, y = as.numeric(prevalence) - 1)) +
+   geom_jitter(position = position_jitter(height = .05), alpha = 0.25) +
```

```
+    geom_rug(position = "jitter", sides = "b") +
+    stat_smooth(method = "loess", color = "red",
+                fill = "red", size = 1.5) +
+    labs(y = "prevalence")
```

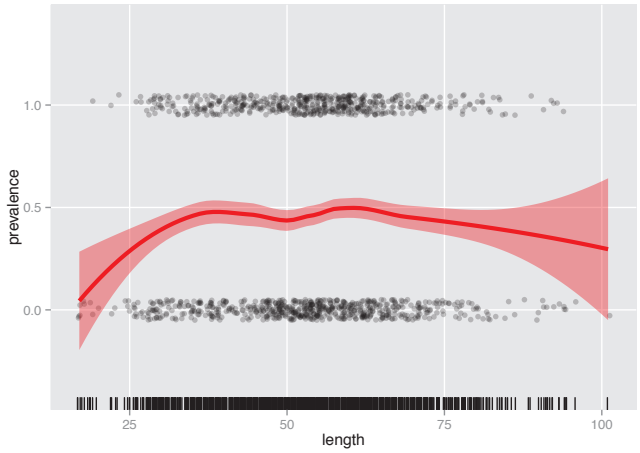


Figure 11.18: Jittered scatterplot of prevalence against length of fish, with loess smooth.

For the positive counts of `intensity`, boxplots by area and year show the distributions of parasites, and it is again useful to display these on a log scale. In Figure 11.19, we have used ggplot2, with `geom_boxplot()` and `geom_jitter()` to also plot the individual observations. Note that `facet_grid()` makes it easy to organize the display with separate panels for each area, a technique that could extend to additional factors.

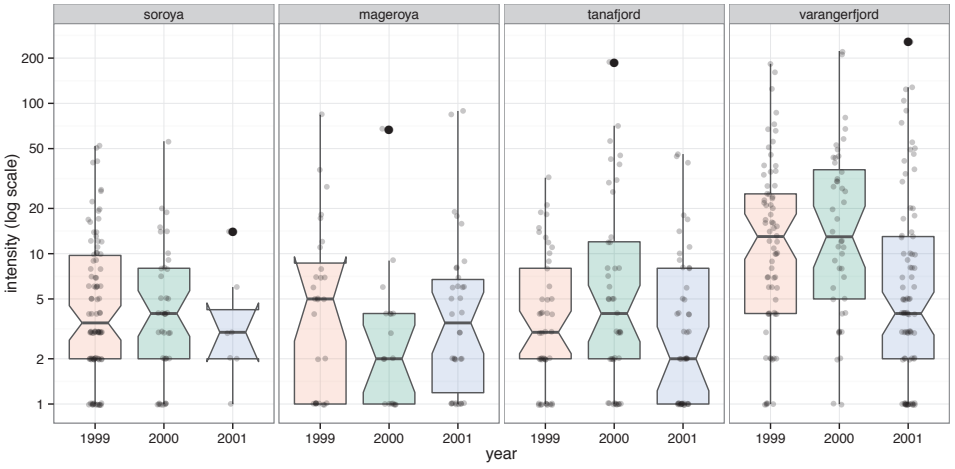


Figure 11.19: Notched boxplots for log (intensity) of parasites by area and year in the CodParasites data. Significant differences in the medians are signaled when the notches of two groups do not overlap.

```
> # plot only positive values of intensity
> CPpos <- subset(CodParasites, intensity > 0)
> ggplot(CPpos, aes(x = year, y = intensity)) +
+   geom_boxplot(outlier.size = 3, notch = TRUE, aes(fill = year),
+                alpha = 0.2) +
+   geom_jitter(position = position_jitter(width = 0.1), alpha = 0.25) +
+   facet_grid(. ~ area) +
+   scale_y_log10(breaks = c(1, 2, 5, 10, 20, 50, 100, 200)) +
+   theme(legend.position = "none") +
+   labs(y = "intensity (log scale)")
```

Most of these distributions are positively skewed and there are a few high outliers, but probably not more than would be expected in a sample of this size. The positive counts (degree of infection) are also higher in all years in Varangerfjord than other areas. You can also see that the intensity values were generally lower in 2001 than other years.

For the effect of length of fish, we want to know if log (intensity) is reasonably linear on length. A jittered scatterplot produced with ggplot2 is shown in Figure 11.20. The smoothed loess curve together with the linear regression line show no indication of nonlinearity.

```
> ggplot(CPpos, aes(x = length, y = intensity)) +
+   geom_jitter(position = position_jitter(height = .1), alpha = 0.25) +
+   geom_rug(position = "jitter", sides = "b") +
+   scale_y_log10(breaks = c(1, 2, 5, 10, 20, 50, 100, 200)) +
+   stat_smooth(method = "loess", color = "red", fill = "red", size = 2) +
+   stat_smooth(method = "lm", size = 1.5)
```

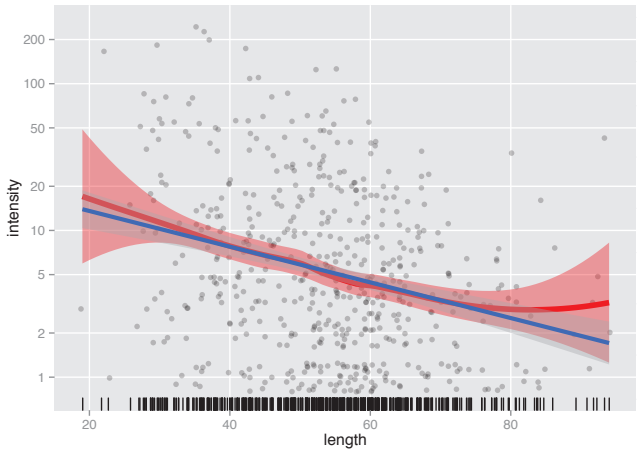


Figure 11.20: Jittered scatterplot of log (intensity) for the positive counts against length of fish, with loess smooth and linear regression line.

△

11.5.1.1 Fitting models

The simple summary of these exploratory analyses is that both the zero component (prevalence) and non-zero component (intensity) involve an interaction of `area` and `year`, and that at least intensity depends on `length`. We proceed to fit some count data models.

EXAMPLE 11.11: Cod parasites

For a baseline reference, we first fit the standard Poisson and negative-binomial models, not allowing for excess zeros.

```
> library(MASS)
> library(countreg)
> cp_p  <- glm(intensity ~ length + area * year,
+              data = CodParasites, family = poisson)
> cp_nb <- glm.nb(intensity ~ length + area * year,
+                 data = CodParasites)
```

Next, we fit analogous hurdle and zero-inflated models, in each case allowing the non-zero count component to be either Poisson or negative-binomial. The zero components are fit as logistic regressions with the same predictors and the logit link.

```
> cp_hp  <- hurdle(intensity ~ length + area * year,
+                  data = CodParasites, dist = "poisson")
> cp_hnb <- hurdle(intensity ~ length + area * year,
+                  data = CodParasites, dist = "negbin")
> cp_zip <- zeroinfl(intensity ~ length + area * year,
+                    data = CodParasites, dist = "poisson")
> cp_znb <- zeroinfl(intensity ~ length + area * year,
+                    data = CodParasites, dist = "negbin")
```

Following Section 11.3.5, we can compare the fit of these models using rootograms. The details of fit of these six models are shown in Figure 11.21.

```
> countreg::rootogram(cp_p, max = 50, main = "Poisson")
> countreg::rootogram(cp_nb, max = 50, main = "Negative Binomial")
> countreg::rootogram(cp_hp, max = 50, main = "Hurdle Poisson")
> countreg::rootogram(cp_hnb, max = 50, main = "Hurdle Negative Binomial")
> countreg::rootogram(cp_zip, max = 50, main = "Zero-inflated Poisson")
> countreg::rootogram(cp_znb, max = 50,
+                               main = "Zero-inflated Negative Binomial")
```

The basic Poisson model of course fits terribly due to the excess zero counts. The hurdle Poisson and zero-inflated Poisson fit the zero counts perfectly, but at the expense of underfitting the counts for low-intensity values. All of the negative binomial models show a reasonable fit (at the scale shown in this plot), and none show a systematic pattern of under/overfitting.

These models are all in different GLM and extended-GLM families, and there are no `anova()` methods for hurdle and zero-inflated models. Each pair of Poisson and negative-binomial models are a nested set, because the Poisson is a special case of the negative-binomial where $\theta \to \infty$, and so can be compared using likelihood-ratio tests available with `lrtest()` from lmtest. However, this cannot be used to compare models of different classes, such as a hurdle model vs. a zero-inflated model. (In Figure 11.21, each pair in the same row are nested models, while all other pairs are non-nested.) Yet, they all have `logLik()` methods to calculate their log likelihood, and so `AIC()` and `BIC()` can be used.

```
> LRstats(cp_p, cp_nb, cp_hp, cp_hnb, cp_zip, cp_znb, sortby = "BIC")

Likelihood summary table:
         AIC   BIC LR Chisq   Df Pr(>Chisq)
cp_p   20378 20444    20352 1178     <2e-16 ***
cp_hp  13688 13820    13636 1165     <2e-16 ***
cp_zip 13687 13819    13635 1165     <2e-16 ***
cp_nb   5031  5102     5003 1178     <2e-16 ***
cp_znb  4955  5092     4901 1164     <2e-16 ***
cp_hnb  4937  5074     4883 1164     <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

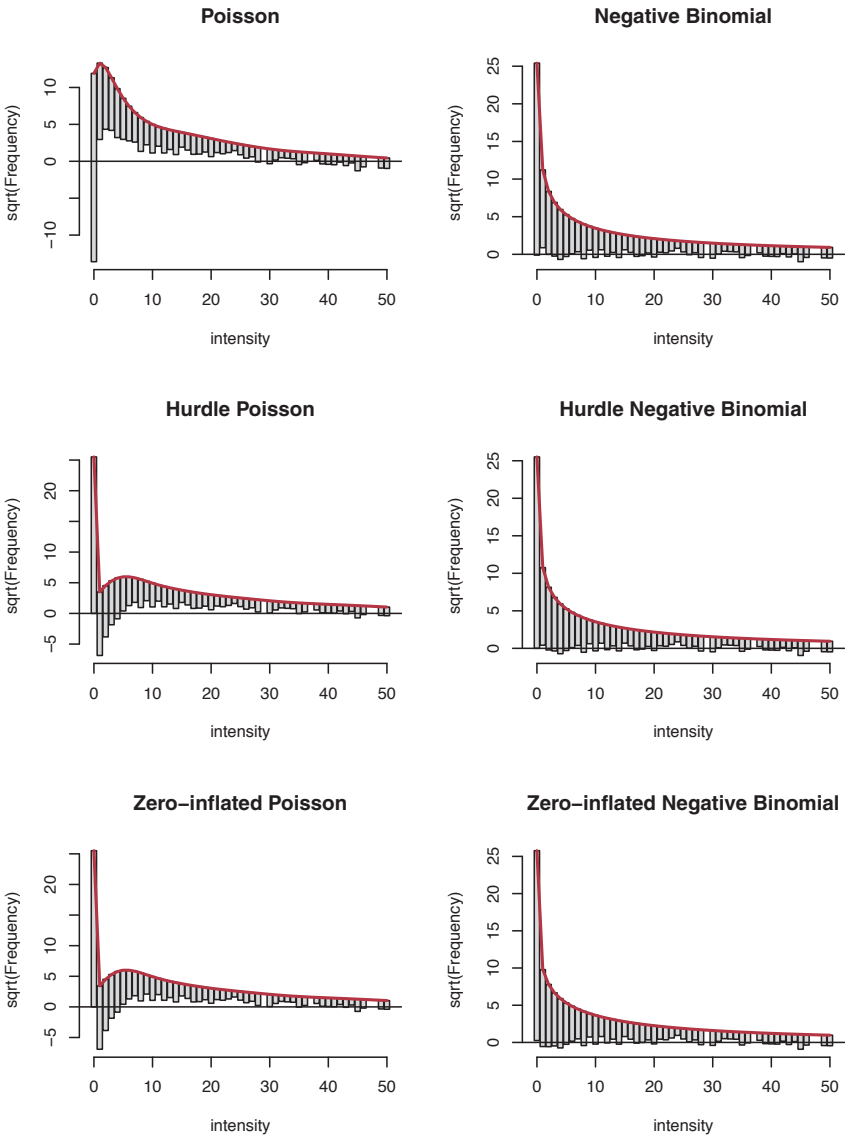


Figure 11.21: Rootograms for six models fit to the CodParasites data.

These show that all the Poisson models fit quite badly, and among the negative-binomial models, the hurdle version, `cp_hnb`, is preferred by both AIC and BIC. If you want to carry out formal tests, `lrtest()` can be used to compare a given Poisson model to its negative-binomial counterpart, which are nested. For example, the test below compares the hurdle Poisson to the hurdle negative-binomial and confirms that the latter is a significant improvement.

```
> library(lmtest)
> lrtest(cp_hp, cp_hnb)

Likelihood ratio test

Model 1: intensity ~ length + area * year
Model 2: intensity ~ length + area * year
  #Df LogLik Df Chisq Pr(>Chisq)
```

```
1  26  -6818
2  27  -2442  1  8752     <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Of greater interest is the difference among the negative-binomial models that are not nested. As described in Section 11.1.4, these can be compared using Voung's test[15]

```
> library(pscl)
> vuong(cp_nb, cp_hnb)      # nb vs. hurdle nb

Vuong Non-Nested Hypothesis Test-Statistic:
(test-statistic is asymptotically distributed N(0,1) under the
 null that the models are indistinguishible)
-------------------------------------------------------------
              Vuong z-statistic             H_A  p-value
Raw                   -5.4873 model2 > model1 2.04e-08
AIC-corrected         -4.2943 model2 > model1 8.76e-06
BIC-corrected         -1.2625 model2 > model1    0.103

> vuong(cp_hnb, cp_znb)     # hurdle nb vs znb

Vuong Non-Nested Hypothesis Test-Statistic:
(test-statistic is asymptotically distributed N(0,1) under the
 null that the models are indistinguishible)
-------------------------------------------------------------
              Vuong z-statistic             H_A p-value
Raw                    1.7941 model1 > model2  0.0364
AIC-corrected          1.7941 model1 > model2  0.0364
BIC-corrected          1.7941 model1 > model2  0.0364
```

The negative-binomial model is considered to be a closer fit than the hurdle version (because it is more parsimonious), while the hurdle NB model has a significantly better fit than the zero-inflated NB model. For this example, we continue to work with the hurdle NB model. The tests for individual coefficients in this model are shown below.

```
> summary(cp_hnb)


Call:
hurdle(formula = intensity ~ length + area * year, data = CodParasites,
    dist = "negbin")

Pearson residuals:
   Min     1Q Median     3Q    Max
-0.696 -0.407 -0.336 -0.108 11.114

Count model coefficients (truncated negbin with log link):
                         Estimate Std. Error z value Pr(>|z|)
(Intercept)               3.37580    0.39947    8.45  < 2e-16 ***
length                   -0.03748    0.00587   -6.38  1.7e-10 ***
areamageroya              0.37898    0.38105    0.99   0.3199
areatanafjord            -0.50480    0.31238   -1.62   0.1061
areavarangerfjord         0.89159    0.29161    3.06   0.0022 **
year2000                 -0.03957    0.32857   -0.12   0.9041
year2001                 -0.75388    0.68925   -1.09   0.2741
areamageroya:year2000    -0.63981    0.61667   -1.04   0.2995
areatanafjord:year2000    1.19387    0.49479    2.41   0.0158 *
areavarangerfjord:year2000  0.51074    0.47719    1.07   0.2845
areamageroya:year2001     0.70444    0.82036    0.86   0.3905
areatanafjord:year2001    0.90824    0.77685    1.17   0.2424
areavarangerfjord:year2001  0.59838    0.74738    0.80   0.4233
```

[15]Note that the Poisson (NB) and ZIP (ZINB) models are, in fact, nested (against popular belief). The Poisson (NB) and HP (HNB) may or may not be nested, depending on which binary zero hurdle is employed. The HNB (HP) and ZINB (ZIP) models may be nested for certain types of covariates.

```
Log(theta)                 -1.49866    0.23904   -6.27  3.6e-10 ***
Zero hurdle model coefficients (binomial with logit link):
                         Estimate Std. Error z value Pr(>|z|)
(Intercept)               0.08526    0.29505    0.29    0.773
length                    0.00693    0.00465    1.49    0.136
areamageroya             -1.32137    0.28526   -4.63  3.6e-06 ***
areatanafjord            -1.44918    0.24388   -5.94  2.8e-09 ***
areavarangerfjord         0.30073    0.27111    1.11    0.267
year2000                  0.39507    0.34382    1.15    0.251
year2001                 -2.65201    0.43340   -6.12  9.4e-10 ***
areamageroya:year2000    -0.08034    0.50797   -0.16    0.874
areatanafjord:year2000    0.87058    0.45027    1.93    0.053 .
areavarangerfjord:year2000  0.86462    0.59239    1.46    0.144
areamageroya:year2001     2.73749    0.53291    5.14  2.8e-07 ***
areatanafjord:year2001    2.71899    0.49949    5.44  5.2e-08 ***
areavarangerfjord:year2001  2.54144    0.51825    4.90  9.4e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Theta: count = 0.223
Number of iterations in BFGS optimization: 25
Log-likelihood: -2.44e+03 on 27 Df
```

From the above and from Figure 11.18, it appears that `length` is not important as a linear effect in the submodel for prevalence. A revised model excludes this from the zero formula.

```
> cp_hnb1 <- hurdle(intensity ~ length + area * year | area * year,
+                   data = CodParasites, dist = "negbin")
```

A likelihood-ratio test shows no advantage for the smaller model; however, Vuong's test leads to the conclusion that this reduced model is preferable:

```
> lrtest(cp_hnb, cp_hnb1)

Likelihood ratio test

Model 1: intensity ~ length + area * year
Model 2: intensity ~ length + area * year | area * year
  #Df LogLik Df Chisq Pr(>Chisq)
1  27  -2442
2  26  -2443 -1  2.23       0.14

> vuong(cp_hnb, cp_hnb1)

Vuong Non-Nested Hypothesis Test-Statistic:
(test-statistic is asymptotically distributed N(0,1) under the
 null that the models are indistinguishible)
-------------------------------------------------------------
              Vuong z-statistic             H_A p-value
Raw                    0.741801 model1 > model2  0.2291
AIC-corrected          0.076907 model1 > model2  0.4693
BIC-corrected         -1.612770 model2 > model1  0.0534
```

△

11.5.1.2 Model interpretation: Effect plots

Interpreting these models from their coefficients is very difficult because an interaction is present and there are separate submodels for the zero and count components. This task is much easier with effects plots. The effects package has methods for any GLM, but cannot handle the extended forms of the zero-inflated and hurdle models.

When the same predictors are used in both submodels, and a standard GLM such as the negative-binomial provides a reasonable fit, you can use the standard effects functions to visualize the (total)

expected count, which for the zeros would include both the extra zeros and those that derive from the count submodel. For visual interpretation, these will be sufficiently similar, even though the hurdle and zero-inflated models differ with respect to explaining overdispersion and/or excess zeros.

Alternatively, if you want to visualize and interpret the zero and nonzero components separately, perhaps with different predictors, you can fit the implied submodels separately, and then use effects functions for the effects in each. These ideas are illustrated in the next example.

EXAMPLE 11.12: Cod parasites

The expected counts for `intensity`, including both zero and positive counts, can be plotted using effects for the `cp_nb` NB model. Figure 11.21 gives some confidence that the fitted values are similar to those in the hurdle and zero-inflated versions.

We use `allEffects()` to calculate the effects for the high-order terms—the main effect of `length` and the interaction of `area` and `year`. These could be plotted together by plotting the resulting `eff.nb` object, but we plot them separately to control the plot details. In these plots, the argument `type="response"` gives plots on the response scale, and we use `ylim` to equate the ranges to make the plots directly comparable. The code below produces Figure 11.22.

```
> library(effects)
> eff.nb <- allEffects(cp_nb)
> plot(eff.nb[1], type = "response", ylim = c(0,30),
+       main  ="NB model: length effect")
>
> plot(eff.nb[2], type = "response", ylim = c(0,30),
+       multiline = TRUE, ci.style = "bars",
+       key.args = list(x = .05, y = .95, columns = 1),
+       colors = c("black", "red", "blue") ,
+       symbols = 15 : 17, cex = 2,
+       main = "NB model: area*year effect")
```

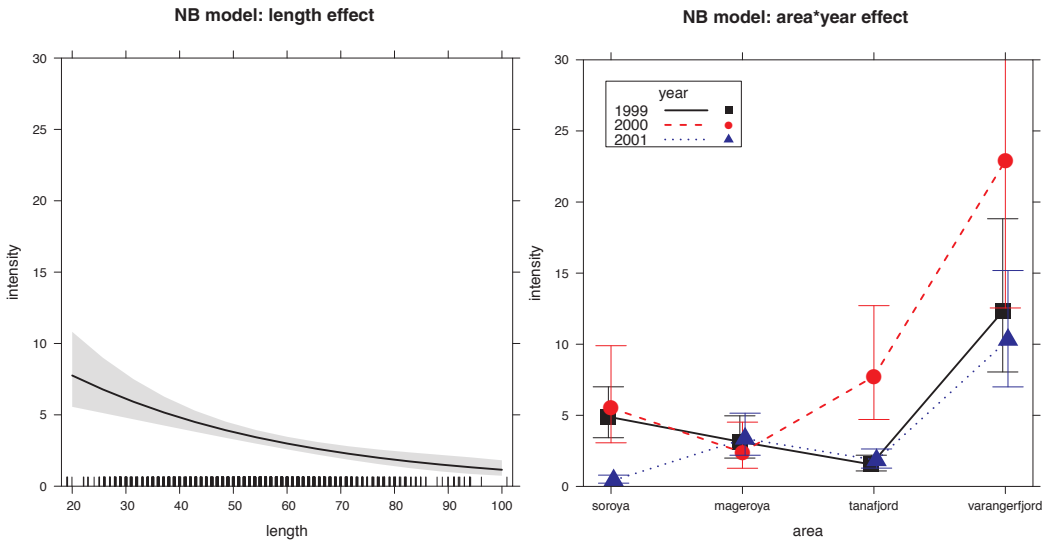


Figure 11.22: Effect plots for total intensity of parasites from the negative-binomial model.

This helps to interpret the nature of the area by year effect. The pattern of mean expected intensity of cod parasites is similar in 1999 and 2001, except for the Sørøya area. The results in year 2000 differ mainly in greater intensity in Tanafjord and Varangerfjord. Varangerfjord shows larger infection counts overall, but particularly in year 2000. The effect plot for length on this scale is roughly comparable to the variation in areas and years.

In this example, the submodels for zero and positive counts have substantively different interpretations. To visualize the fitted effects in these submodels using effects, first fit the equivalent submodels separately using GLM methods. The following models for prevalence, using the binomial family, and the positive counts for intensity, using glm.nb(), give similar fitted results to those obtained from the hurdle negative-binomial model, cp_hnb discussed earlier.

```
> cp_zero  <- glm(prevalence ~ length + area * year,
+                 data = CodParasites, family = binomial)
> cp_nzero <- glm.nb(intensity ~ length + area * year,
+                    data = CodParasites, subset = intensity > 0)
```

We could construct effect plots for each of these submodels, but interest here is largely on the binomial model for the zero counts, cp_zero. Effect plots for the terms in this model are shown in Figure 11.23. Again, we set the ylim values to equate the vertical ranges to make the plots comparable.

```
> eff.zero <- allEffects(cp_zero)
> plot(eff.zero[1], ylim=c(-2.5, 2.5),
+      main="Hurdle zero model: length effect")
>
> plot(eff.zero[2],  ylim=c(-2.5, 2.5),
+      multiline=TRUE,
+      key.args=list(x=.05, y=.95, columns=1),
+      colors=c("black", "red", "blue"),
+      symbols=15:17, cex=2,
+      main="Hurdle zero model: area*year effect")
```

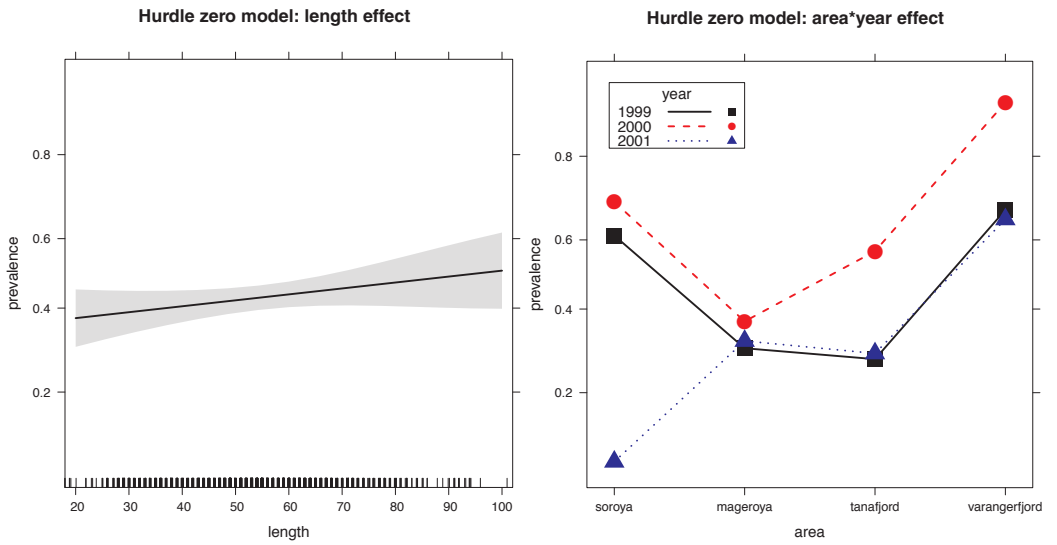


Figure 11.23: Effect plots for prevalence of parasites analogous to the hurdle negative-binomial model, fitted using a binomial GLM model.

The effect of length on prevalence is slightly increasing, but we saw earlier that this is not significant. For the area–year interaction, the three curves have similar shapes, except for the aberrant value for Sørøya in 2001 and the closeness of the values at Magerøya in all years. Overall, prevalence was highest in 2000, and also in the Varangerfjord samples.

△

11.5.2 Demand for medical care by the elderly

A large cross-sectional study was carried out by the U.S. National Medical Expenditure Survey (NMES) in 1987–1988 to assess the demand for medical care, as measured by the number of physician/non-physician office visits and the number of hospital outpatient visits to a physician/non-physician. The survey was based upon a representative national probability sample of the civilian non-institutionalized population and individuals admitted to long-term care facilities during 1987. A subsample of 4,406 individuals aged 66 and over, all of whom are covered by Medicare, is contained in the *NMES1988* data set in the AER package. These data were previously analyzed by Deb and Trivedi (1997) and Zeileis et al. (2008), from which this account borrows. The objective of the study and these analyses is to create a descriptive, and hopefully predictive, model for the demand for medical care in this elderly population.

EXAMPLE 11.13: Demand for medical care

The potential response variables in the *NMES1988* data set form a 2×2 set of the combinations of *place of visit* (office vs. hospital) and (physician vs. non-physician) *practitioner*. Here, we focus on the highest total frequency variable `visits`, recording office visits to a physician. There are quite a few potential predictors, but here we consider only the following:

- `hospital`: number of hospital stays[16]
- `health`: a factor indicating self-perceived health status, with categories `"poor"`, `"average"` (reference category), `"excellent"`
- `chronic`: number of chronic conditions
- `gender`
- `school`: number of years of education
- `insurance`: a factor. Is the individual covered by private insurance?

For convenience, these variables are extracted to a reduced data set, `nmes`.

```
> data("NMES1988", package = "AER")
> nmes <- NMES1988[, c(1, 6:8, 13, 15, 18)]
```

A quick overview of the response variable, `visits`, is shown as simple (unbinned) histograms on the frequency and log(frequency) scales in Figure 11.24. The zero counts are not as extreme as we have seen in other examples. On the log scale, there is a small, but noticeable spike at 0, followed by a progressive, nearly linear decline, up to about 30 visits.

```
> plot(table(nmes$visits),
+      xlab = "Physician office visits", ylab = "Frequency")
> plot(log(table(nmes$visits)),
+      xlab = "Physician office visits", ylab = "log(Frequency)")
```

However, as a benchmark, without taking any predictors into account, there is very substantial overdispersion relative to a Poisson distribution, the variance being nearly 8 times the mean.

```
> with(nmes, c(mean = mean(visits),
+              var = var(visits),
+              ratio = var(visits) / mean(visits)))

   mean     var   ratio
 5.7744 45.6871  7.9120
```

[16] It is arguable that number of hospitalizations should be regarded as a dependent variable, reflecting another aspect of demand for medical care, rather than as a predictor. We include it here as a predictor to control for its relationship to the outcome `visits`.

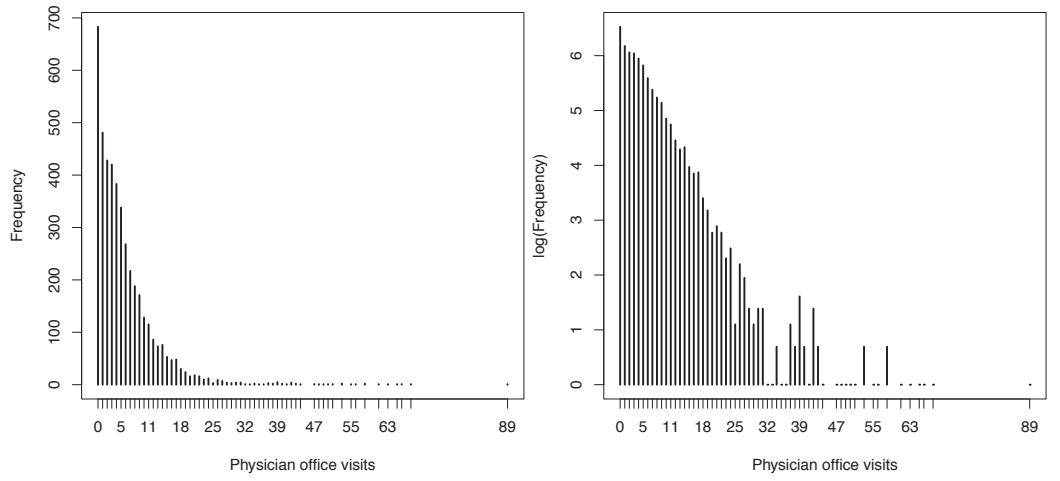


Figure 11.24: Frequency distributions of the number of physician office visits.

As before, it is useful to precede the formal analysis with a variety of exploratory plots. Figure 11.25 shows a few of these as boxplots, using `cutfac()` to make predictors discrete, and plotting `visits` on a log scale, started at 1. All of these show the expected relationships, e.g., number of office visits increases with numbers of chronic conditions and hospital stays, but decreases with better perceived health status.

```
> plot(log(visits + 1) ~ cutfac(chronic), data = nmes,
+      ylab = "Physician office visits (log scale)",
+      xlab = "Number of chronic conditions", main = "chronic")
> plot(log(visits + 1) ~ health, data = nmes, varwidth = TRUE,
+      ylab = "Physician office visits (log scale)",
+      xlab = "Self-perceived health status", main = "health")
> plot(log(visits + 1) ~ cutfac(hospital, c(0:2, 8)), data = nmes,
+      ylab = "Physician office visits (log scale)",
+      xlab = "Number of hospital stays", main = "hospital")
```

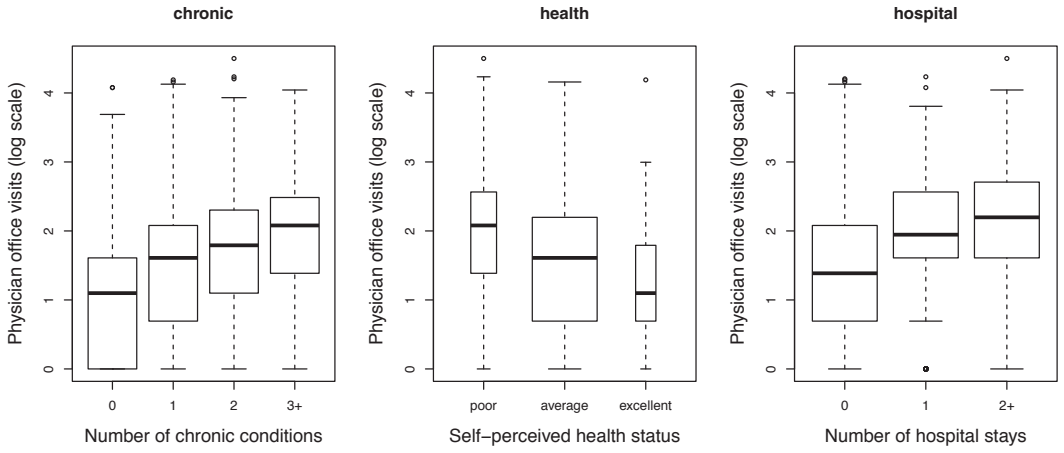


Figure 11.25: Number of physician office visits plotted against some of the predictors.

Similar plots for `insurance` and `gender` show that those with private insurance have more office visits and women slightly more than men.

The relationship with number of years of education could be shown in boxplots by the use of `cutfac(school)`, or with `spineplot()` by making both variables discrete. However, it is more informative (shows the data) to depict this in a smoothed and jittered scatterplot, as in Figure 11.26.

```
> library(ggplot2)
> ggplot(nmes, aes(x = school, y = visits + 1)) +
+   geom_jitter(alpha = 0.25) +
+   stat_smooth(method = "loess", color = "red", fill = "red",
+               size = 1.5, alpha = 0.3) +
+   labs(x = "Number of years of education",
+        y = "log(Physician office visits + 1)") +
+   scale_y_log10(breaks = c(1, 2, 5, 10, 20, 50, 100))
```

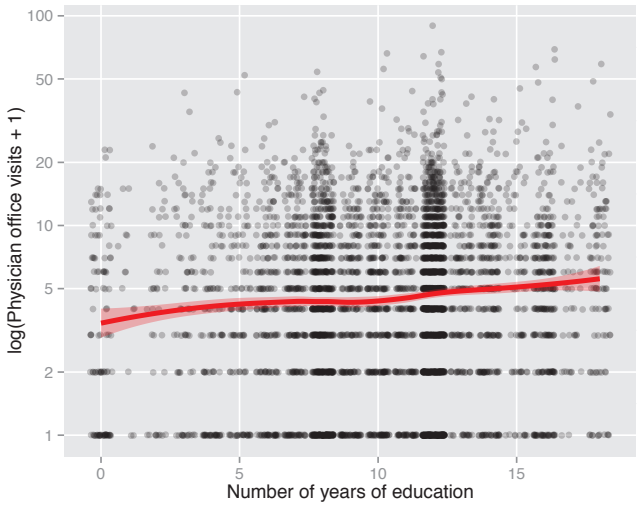


Figure 11.26: Jittered scatterplot of physician office visits against number of years of education, with nonparametric (loess) smooth.

As you might expect, there is a small but steady increase in mean office visits with years of education. It is somewhat surprising that there are quite a few individuals with 0 years of education; jittering also shows the greater density of observations at 8 and 12 years.

As in previous examples, a variety of other exploratory plots would be helpful in understanding the relationships among these variables *jointly*, particularly how office visits depends on combinations of two (or more) predictors. Some natural candidates would include mosaic and doubledecker plots (using `cutfac(visits)`), e.g., as in Figure 11.17, and conditional or faceted versions of the boxplots shown in Figure 11.25, each stratified by one (or more) additional predictors. These activities are left as exercises for the reader.

△

11.5.2.1 Fitting models

Most previous analyses of these data have focused on exploring and comparing different types of count data regression models. Deb and Trivedi (1997) compared the adequacy of fit of the negative-binomial, a hurdle NB, and models using finite mixtures of NB models. Zeileis et al. (2008) used

this data to illustrate hurdle and zero-inflated models using the countreg package, while Cameron and Trivedi (1998, 2013) explored a variety of competing models, including 1- and 2-parameter NB models and C-component finite mixture models that can be thought of as generalizations of the 2-component models described in Section 11.4.

In most cases, the full set of available predictors was used, and models were compared using the standard methods for model selection: likelihood-ratio tests for nested models, AIC, BIC, and so forth. An exception is Kleiber and Zeileis (2014), who used a reduced set of predictors similar to those employed here, and illustrated the use of rootograms and plots of predicted values for visualizing and comparing fitted models.

This is where model comparison and selection for count data models (and other GLMs) adds another layer of complexity beyond what needs to be considered for classical (Gaussian) linear models, standard logistic regression models, and the special case of loglinear models treated earlier. Thus, when we consider and compare different distribution types or link functions, we have to be reasonably confident that the systematic part of the model has been correctly specified (as we noted in Section 11.3), and is the *same* in all competing models, so that any differences can be attributed to the distribution type. However, lack-of-fit may still arise because the systematic part of the model is incorrect.

In short, we cannot easily compare apples to oranges (different distributions with different regressors), but we also have to make sure we have a good apple to begin with. The important questions are:

- Have all important predictors and control variables been included in the model?
- Are quantitative predictors represented on the correct scale (via transformations or nonlinear terms) so their effects are reasonably additive for the linear predictor?
- Are there important interactions among the explanatory variables?

EXAMPLE 11.14: Demand for medical care

In this example, we start with the all main-effects model of the predictors in the `nmes` data, similar to that considered by Zeileis et al. (2008). We first fit the basic Poisson and NB models, as points of reference.

```
> nmes.pois <-    glm(visits ~ ., data = nmes, family = poisson)
> nmes.nbin <- glm.nb(visits ~ ., data = nmes)
```

A quick check with `lmtest()` shows that the NB model is clearly superior to the standard Poisson regression model as we expect (and also to the quasi-Poisson).

```
> library(lmtest)
> lrtest(nmes.pois, nmes.nbin)

Likelihood ratio test

Model 1: visits ~ hospital + health + chronic + gender + school + insurance
Model 2: visits ~ hospital + health + chronic + gender + school + insurance
  #Df LogLik Df Chisq Pr(>Chisq)
1   8 -17972
2   9 -12171  1 11602     <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The model summary for the NB model below shows the coefficients area all significant. Moreover, the signs of the coefficients are all as we would expect from our exploratory plots. For example, log(visits) increases with number of hospital stays, chronic conditions, and education, and is greater for females and those with private health insurance. So, what's not to like?

```
> summary(nmes.nbin)


Call:
glm.nb(formula = visits ~ ., data = nmes, init.theta = 1.206603534,
    link = log)

Deviance Residuals:
   Min      1Q  Median      3Q     Max
-3.047  -0.995  -0.295   0.296   5.818

Coefficients:
                Estimate Std. Error z value Pr(>|z|)
(Intercept)      0.92926    0.05459   17.02  < 2e-16 ***
hospital         0.21777    0.02018   10.79  < 2e-16 ***
healthpoor       0.30501    0.04851    6.29  3.2e-10 ***
healthexcellent -0.34181    0.06092   -5.61  2.0e-08 ***
chronic          0.17492    0.01209   14.47  < 2e-16 ***
gendermale      -0.12649    0.03122   -4.05  5.1e-05 ***
school           0.02682    0.00439    6.10  1.0e-09 ***
insuranceyes     0.22440    0.03946    5.69  1.3e-08 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for Negative Binomial(1.2066) family taken to be 1)

    Null deviance: 5743.7  on 4405  degrees of freedom
Residual deviance: 5044.5  on 4398  degrees of freedom
AIC: 24359

Number of Fisher Scoring iterations: 1


              Theta:  1.2066
          Std. Err.:  0.0336

 2 x log-likelihood:  -24341.1070
```

This all-main-effects model is relatively simple to interpret, but a more important question is whether it adequately explains the relations of the predictors to the outcome, `visits`.

Significant interactions among the predictors could substantially change the interpretation of the model, and in the end, could affect policy recommendations based on this analysis. This question turns out to be far more interesting and important than the subtle differences among models for handling overdispersion and zero counts.

One simple way to consider whether there are important interactions among the predictors that better explain patient visits is to get simple tests of the additional contribution of each two-way (or higher-way) interaction using the `add1()` function. The formula argument in the call below specifies to test the addition of all two-way terms.

```
> add1(nmes.nbin, . ~ .^2, test = "Chisq")

Single term additions

Model:
visits ~ hospital + health + chronic + gender + school + insurance
                  Df Deviance   AIC  LRT Pr(>Chi)
<none>                  5045 24357
hospital:health    2     5025 24341 19.9  4.7e-05 ***
hospital:chronic   1     5009 24324 35.2  3.0e-09 ***
hospital:gender    1     5044 24358  0.8   0.3650
hospital:school    1     5041 24355  4.0   0.0453 *
```

```
hospital:insurance  1     5036 24351  8.0   0.0046 **
health:chronic      2     5005 24322 39.5  2.6e-09 ***
health:gender       2     5040 24357  4.3   0.1172
health:school       2     5030 24347 14.3   0.0008 ***
health:insurance    2     5032 24348 12.9   0.0016 **
chronic:gender      1     5045 24359  0.0   0.9008
chronic:school      1     5043 24357  1.9   0.1705
chronic:insurance   1     5039 24354  5.1   0.0246 *
gender:school       1     5040 24354  4.8   0.0290 *
gender:insurance    1     5042 24357  2.5   0.1169
school:insurance    1     5037 24352  7.2   0.0072 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

From this, we decide to add all two-way interactions among `health`, `hosp`, and `numchron`, and also the two-way interaction `health:school`. Other significant interactions could also be explored, but we don't do this here.

```
> nmes.nbin2 <- update(nmes.nbin,
+                      . ~ . + (health + chronic + hospital)^2
+                            + health : school)
```

This model clearly fits much better than the main effects model, as shown by a likelihood ratio test. The same conclusion would result from `anova()`.

```
> lrtest(nmes.nbin, nmes.nbin2)

Likelihood ratio test

Model 1: visits ~ hospital + health + chronic + gender + school + insurance
Model 2: visits ~ hospital + health + chronic + gender + school + insurance
    + health:chronic + hospital:health + hospital:chronic + health:school
  #Df LogLik Df Chisq Pr(>Chisq)
1   9 -12171
2  16 -12133  7  74.3     2e-13 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

△

11.5.2.2 Model interpretation: Effect plots

Complex models with more than a few predictors are difficult to understand and explain, even more so when there are interactions among the predictors. As we have noted previously, effect plots (Fox, 1987, Fox and Andersen, 2006) provide a ready solution.

They have the advantage that each plot shows the correct *partial* relation between the response and the variables in the term shown, controlling (adjusting) for all other variables in the model, as opposed to *marginal* plots that ignore all other variables. From these, it is possible to read an interpretation of a given model effect directly from the effect plot graphs, knowing that all variables not shown in a given graph have been controlled (adjusted for) by setting them equal to average or typical values.

A disadvantage is that these plots show only the predicted (fitted) effects under the *given model* (and not the *data*). If relationships of the response to predictors are nonlinear, or important interactions are not included in the model, you won't see this in an effect plot. We illustrate this point using the results of the main effect NB model, `nmes.nbin`, as shown in Figure 11.27.

EXAMPLE 11.15: Demand for medical care

```
> library(effects)
> plot(allEffects(nmes.nbin), ylab = "Office visits")
```

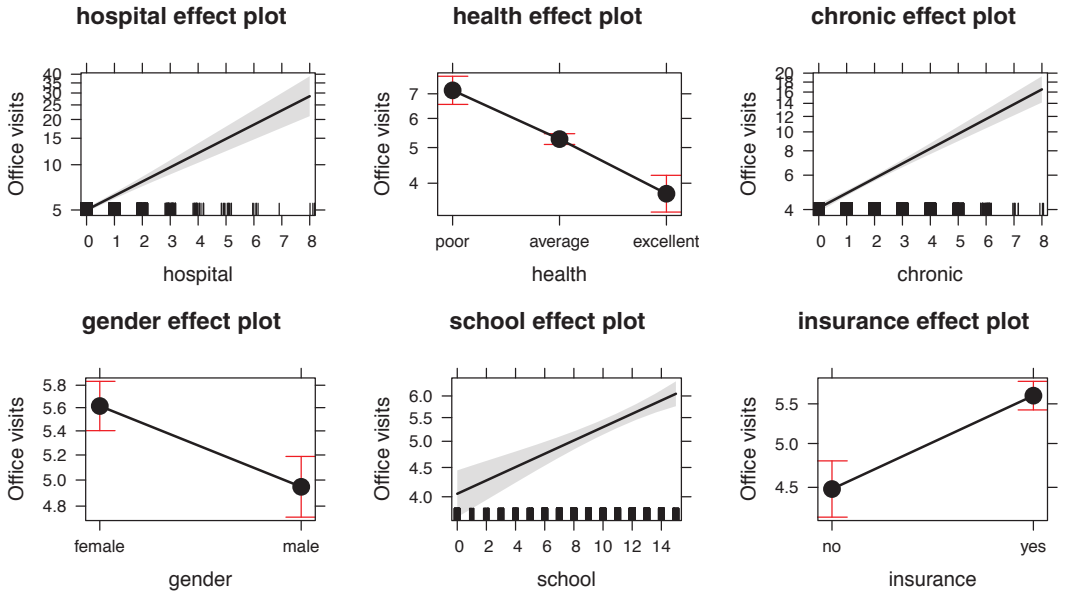


Figure 11.27: Effect plots for the main effects of each predictor in the negative binomial model nmes.nbin.

All of these panels show the expected relations of the predictors to the `visits` response, and the confidence bands and error bars provide visual tests of the sizes of differences. But they don't tell the full story, because the presence of an important interaction (such as `health:chronic`) means that the effect of one predictor (`health`) differs over the values of the other (`chronic`).

We can see this clearly in effect plots for the model `nmes.nbin2` with interactions. For display purposes, it is convenient here to calculate the fitted effects for model terms over a smaller but representative subset of the levels of the integer-valued predictors, using the `xlevels=` argument to `allEffects()`.

```
> eff_nbin2 <- allEffects(nmes.nbin2,
+   xlevels = list(hospital = c(0 : 3, 6, 8),
+   chronic = c(0:3, 6, 8),
+   school = seq(0, 20, 5)))
```

The result of `allEffects()`, `eff_nbin2`, is a "efflist" object, a list of effects for each *high-order term* in the model. Note that only the terms `gender` and `insurance`, not involved in any interaction, appear as main effects here.

```
> names(eff_nbin2)

[1] "gender"           "insurance"        "health:chronic"
[4] "hospital:health"  "hospital:chronic" "health:school"
```

Plotting the entire "efflist" object gives a collection of plots, one for each high-order term, as in Figure 11.27, and is handy for a first look. However, the `plot()` methods for effects objects offer greater flexibility when you plot terms individually using additional options. For example, Figure 11.28 plots the effect for the interaction of health and number of chronic conditions with a few optional arguments. See `help(plot.eff, package="effects")` for the available options.

```
> plot(eff_nbin2, "health:chronic", layout = c(3, 1),
+       ylab = "Office visits", colors = "blue")
```

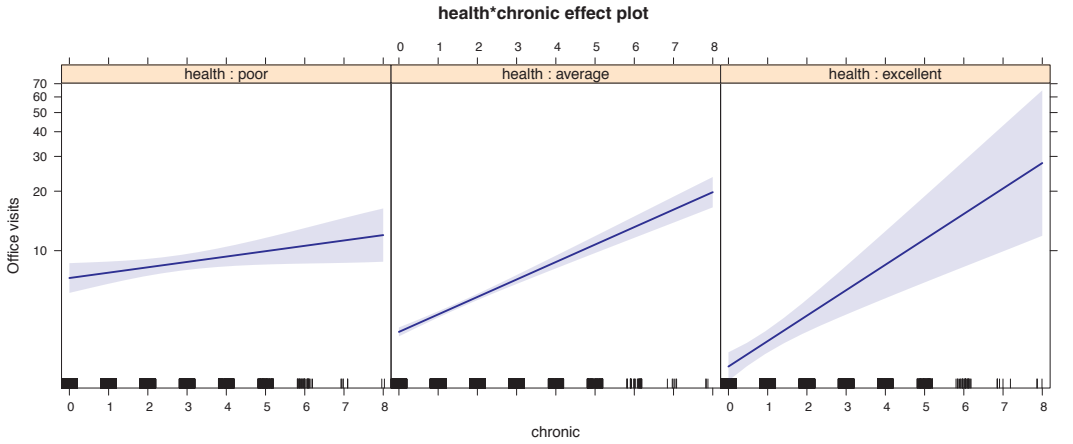


Figure 11.28: Effect plot for the interaction of health and number of chronic conditions in the model nmes.nbin2.

The default style shown in Figure 11.28 is a conditional or faceted plot, graphing the response against the X variable with the greatest number of levels, with separate panels for the levels of the other predictor. Alternatively, the effects for a given term can be shown overlaid in a single plot, using the `multiline=TRUE` argument, as shown in Figure 11.29 for the two interactions involving health status. Not only is this style more compact, but it also makes direct comparison of the trends for the other variable easier.

```
> plot(eff_nbin2,
+       "health:chronic", multiline = TRUE, ci.style = "bands",
+       ylab = "Office visits", xlab  ="# Chronic conditions",
+       key.args = list(x = 0.05, y = .80, corner = c(0, 0), columns = 1))
>
> plot(eff_nbin2,
+       "hospital:health", multiline = TRUE, ci.style = "bands",
+       ylab = "Office visits", xlab = "Hospital stays",
+       key.args = list(x = 0.05, y = .80, corner = c(0, 0), columns = 1))
```

From both Figure 11.28 and the left panel of Figure 11.29, it can be seen that for people with poor health status, the relationship of chronic conditions to office visits is relatively flat. For those who view their health status as excellent, their use of office visits is much more strongly related to their number of chronic conditions.

The interaction of perceived health status with number of hospital stays (right panel of Figure 11.29) shows that the difference in office visits according to health status is mainly important only for those with 0 or 1 hospital stays.

The remaining two interaction effects are plotted in Figure 11.30. The interaction of hospital