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

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

`ftable()` and `structable()` have `as.matrix()` methods[2] that convert their result into a matrix suitable as input to `ca()`.

With data in the form of a frequency data frame, you can easily create interactive coding using `interaction()` or simply use `paste()` to join the levels of stacked variables together.

To illustrate, create a 4-way table of random Poisson counts (with constant mean, $\lambda = 15$) of types of Pet, classified by Age, Color and Sex.

```
> set.seed(1234)
> dim <- c(3, 2, 2, 2)
> tab <- array(rpois(prod(dim), 15), dim = dim)
> dimnames(tab) <- list(Pet = c("dog", "cat", "bird"),
+                       Age = c("young", "old"),
+                       Color = c("black", "white"),
+                       Sex = c("male", "female"))
```

You can use `ftable()` to print this, with a formula that assigns `Pet` and `Age` to the columns and `Color` and `Sex` to the rows.

```
> ftable(Pet + Age ~ Color + Sex, tab)

             Pet   dog       cat      bird
             Age young old young old young old
Color Sex
black male        10  12    16  16    16  12
      female       8  12    13  15    11  13
white male        18  11    12  18    13  20
      female      13  13    16  15    12  15
```

Then, `as.matrix()` creates a matrix with the levels of the stacked variables combined with some separator character. Using `ca(pet.mat)` would then calculate the CA solution for the stacked table, analyzing only the associations in the loglinear model [Pet Age][Color Sex].[3]

```
> (pet.mat <- as.matrix(ftable(Pet + Age ~ Color + Sex, tab), sep = '.'))

             Pet.Age
Color.Sex     dog.young dog.old cat.young cat.old bird.young bird.old
  black.male         10      12        16      16         16       12
  black.female        8      12        13      15         11       13
  white.male         18      11        12      18         13       20
  white.female       13      13        16      15         12       15
```

With data in a frequency data frame, a similar result (as a frequency table) can be obtained using `interaction()` as shown below. The result of `xtabs()` looks the same as `pet.mat`.

```
> tab.df <- as.data.frame(as.table(tab))
> tab.df <- within(tab.df,
+   {Pet.Age = interaction(Pet, Age)
```

[2] This requires at least R version 3.1.0 or vcd 1.3-2 or later.

[3] The result would not be at all interesting here. Why?

singular value decomposition of the matrix $\boldsymbol{B}$, which produces scores for the categories of *all* variables so that the greatest proportion of the bivariate, pairwise associations in all blocks (including the diagonal blocks) is accounted for in a small number of dimensions.

In this respect, MCA resembles multivariate methods for quantitative data based on the joint bivariate correlation or covariance matrix ($\boldsymbol{\Sigma}$) and there is some justification for regarding the Burt matrix as the categorical analog of $\boldsymbol{\Sigma}$.[5]

There is a close connection between this analysis and the bivariate mosaic matrix (Section 5.6): The mosaic matrix displays the residuals from independence for each pair of variables, and thus provides a visual representation of the Burt matrix. The one-way margins shown (by default) in the diagonal cells reflect the diagonal matrices $\boldsymbol{N}_i$ in Eqn. (6.7). The total amount of shading in all the individual mosaics portrays the total pairwise associations decomposed by MCA. See Friendly (1999a) for further details.

For interpretation of MCA plots, we note the following relations (Greenacre, 1984, Section 5.2):[6]

- The inertia contributed by a given variable increases with the number of response categories.
- The centroid of the categories for each discrete variable is at the origin of the display.
- For a particular variable, the inertia contributed by a given category increases as the marginal frequency in that category *decreases*. Low frequency points therefore appear further from the origin.
- The category points for a binary variable lie on a line through the origin. The distance of each point to the origin is inversely related to the marginal frequency.

EXAMPLE 6.8: Marital status and pre- and extramarital sex

The data on the relation between marital status and reported premarital and extramarital sex was explored earlier using mosaic displays in Example 5.9 and Example 5.13.

Using the ca package, an MCA analysis of the `PreSex` data is carried out using `mjca()`. This function typically takes a data frame in *case form* containing the factor variables, but converts a table to this form. This example analyzes the Burt matrix calculated from the `PreSex` data, specified as `lambda="Burt"`

```
> data("PreSex", package = "vcd")
> PreSex <- aperm(PreSex, 4:1)   # order variables G, P, E, M
> presex.mca <- mjca(PreSex, lambda = "Burt")
> summary(presex.mca)

Principal inertias (eigenvalues):

 dim    value      %    cum%   scree plot
 1      0.149930  53.6  53.6   *************
 2      0.067201  24.0  77.6   ******
 3      0.035396  12.6  90.2   ***
 4      0.027365   9.8 100.0   **
        -------- -----
 Total: 0.279892 100.0
...
```

The output from `summary()` seems to show that 77.6% of the total inertia is accounted for in two dimensions. A basic, default plot of the MCA solution is provided by the `plot()` method for "mjca" objects.

[5] For multivariate normal data, however, the mean vector and covariance matrix are sufficient statistics, so all higher-way relations are captured in the covariance matrix. This is not true of the Burt matrix. Moreover, the covariance matrix is typically expressed in terms of mean-centered variables, while the Burt matrix involves the marginal frequencies. A more accurate statement is that the uncentered covariance matrix is analogous to the Burt matrix.

[6] This book, now out of print, is available for free download at `http://www.carme-n.org/`.

(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)
```

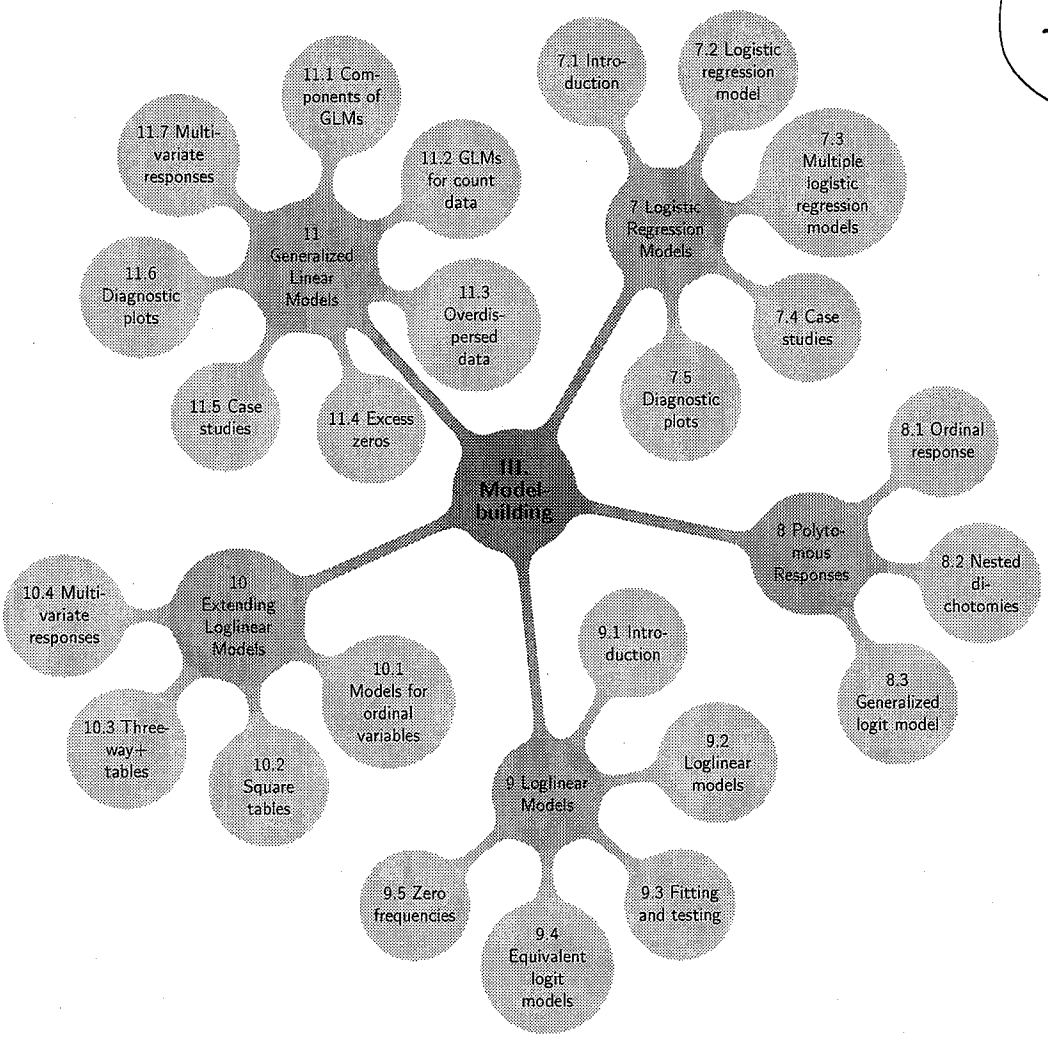
(a) Use `mjca()` to carry out an MCA on the four-way table `accident.tab`.
(b) Construct an informative 2D plot of the solution, and interpret in terms of how the variable `result` varies in relation to the other factors.

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?

Part III

Model-building Methods



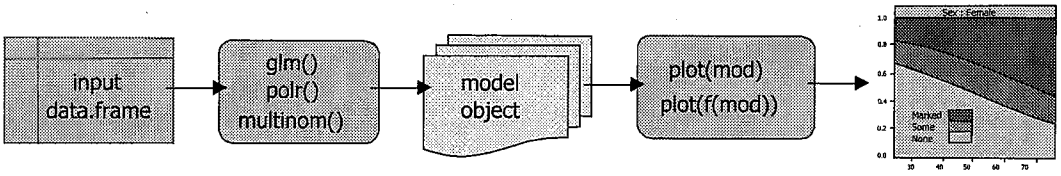


Figure 7.1: Overview of fitting and graphing for model-based methods in R.

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.

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 models***, 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.

```
> 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)
> 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)
```

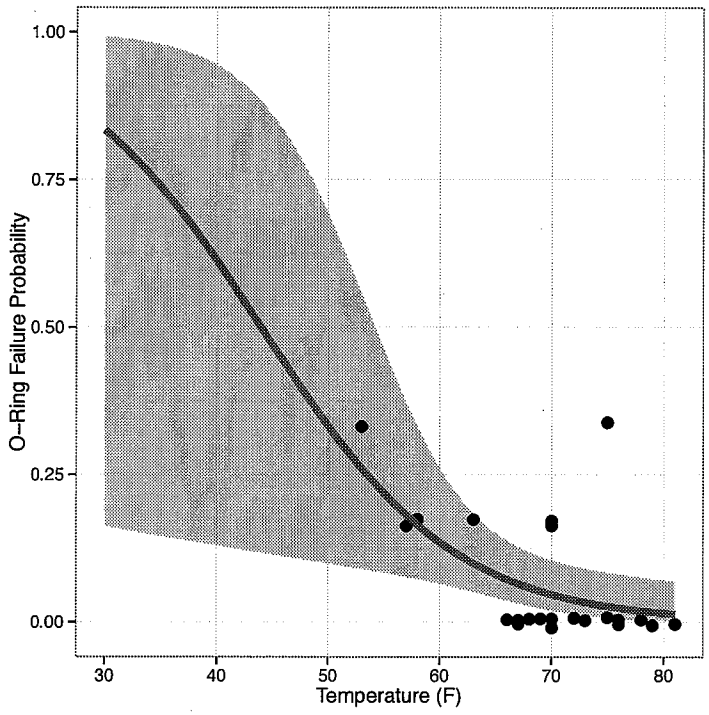


Figure 7.5: Space shuttle data, with fitted logistic regression model.

△

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

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

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))
```

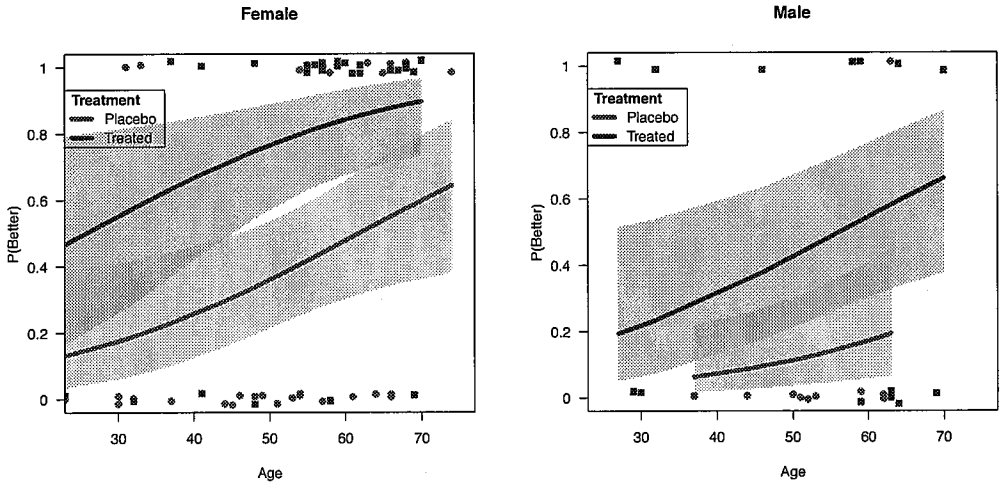


Figure 7.9: Full-model plot of Arthritis data, showing fitted probabilities by Treatment and Sex.

△

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.

```
> 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.

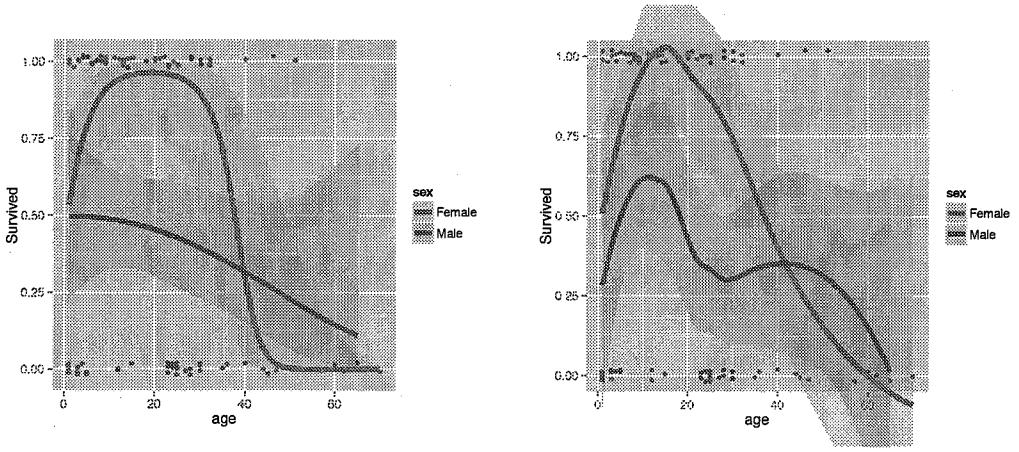


Figure 7.16: Conditional plots of the Donner data, showing the relationship of survival to age and sex. Left: The smoothed curves and confidence bands show the result of fitting separate quadratic logistic regressions on age for males and females. Right: Separate loess smooths are fit to the data for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

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

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.

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)))
```

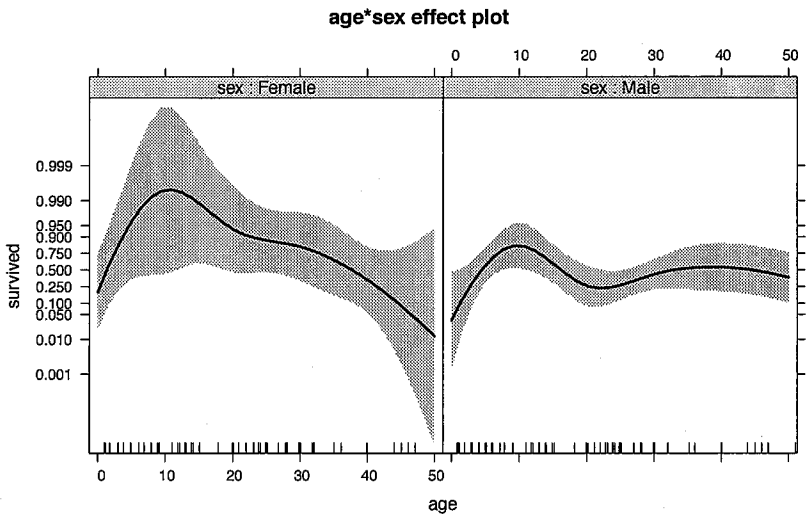


Figure 7.17: Effect plot for the spline model `donner.mod6` fit to the Donner data.

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.

```
> 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.

```
> 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)))
```

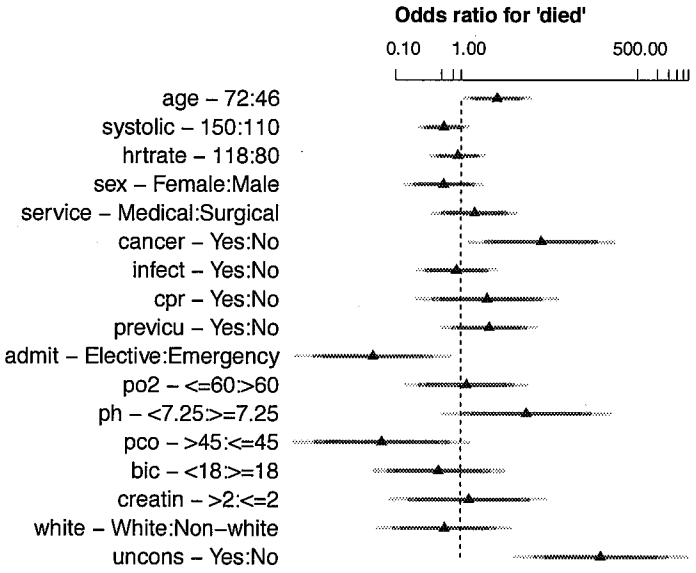


Figure 7.21: Odds ratios for the terms in the model for the ICU data. Each line shows the odds ratio for a term, together with lines for 90, 95, and 99% confidence intervals in progressively darker shades.

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.

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.

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,

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

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()`:

- 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
```

```
   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)))
```

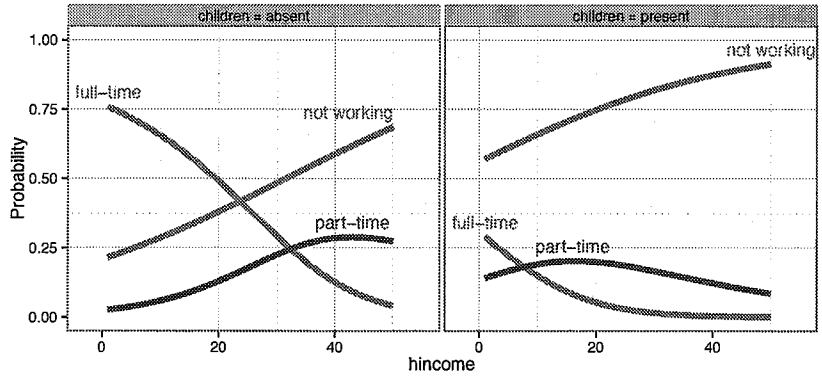


Figure 8.10: Fitted probabilities from the models for nested dichotomies fit to the data on women's labor force participation.

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.

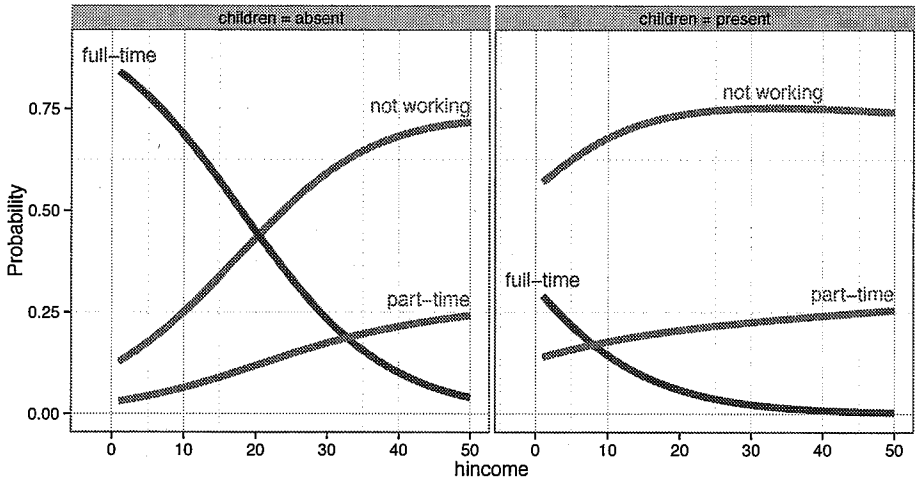


Figure 8.12: Fitted probabilities from the generalized logit model fit to the data on women's labor force participation.

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))
```

△

(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.

```
> 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 $\Sigma \boldsymbol{n} \to \infty$), reasonably well-approximated when all expected frequencies are large.[1] The (residual) degrees of freedom are the

[1] Except in bizarre or borderline cases, these tests provide the same conclusions when expected frequencies are at least moderate (all $\widehat{m} > 5$). 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.

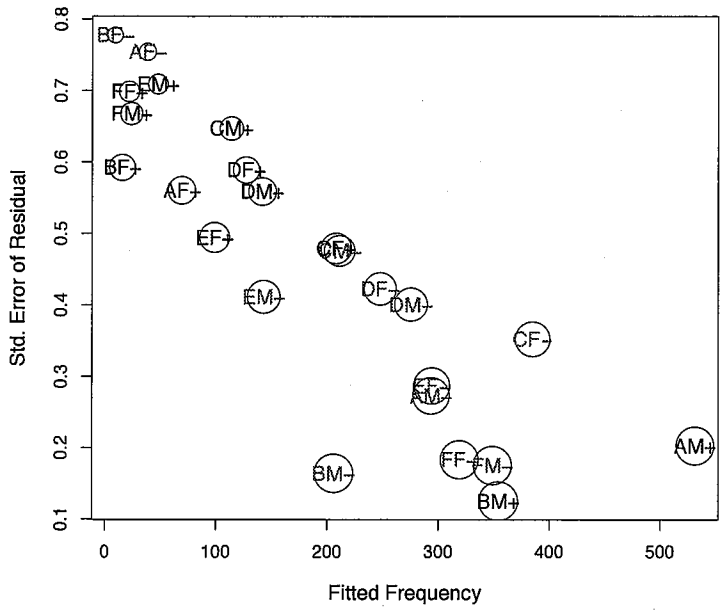


Figure 9.1: Standard errors of residuals, $\sqrt{1-h_i}$ decrease with expected frequencies. This plot shows why ordinary Pearson and deviance residuals may be misleading. The symbol size in the plot is proportional to leverage, h_i. Labels abbreviate Department, Gender, and Admit, colored by Admit.

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.

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`

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)
```

[4]Using weights gives the same fitted values, but not the same LR tests for model fit.

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

$$df = (\#\text{ cells} - \#\text{ of parameters}) - (\#\text{ zero cells} - \#\text{ of NA parameters}), \qquad (9.18)$$

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.

The `Health` data is created as a frequency data frame as follows.

[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.

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)

```
> 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.

```
> vcdExtra::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 = c(left = 65))
```

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)
> vcdExtra::LRstats(health.glm2)
```

```
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)

                       Estimate Std. Error
Mult(mental, .).ses1   1.111360    0.29921
```

```
> 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.

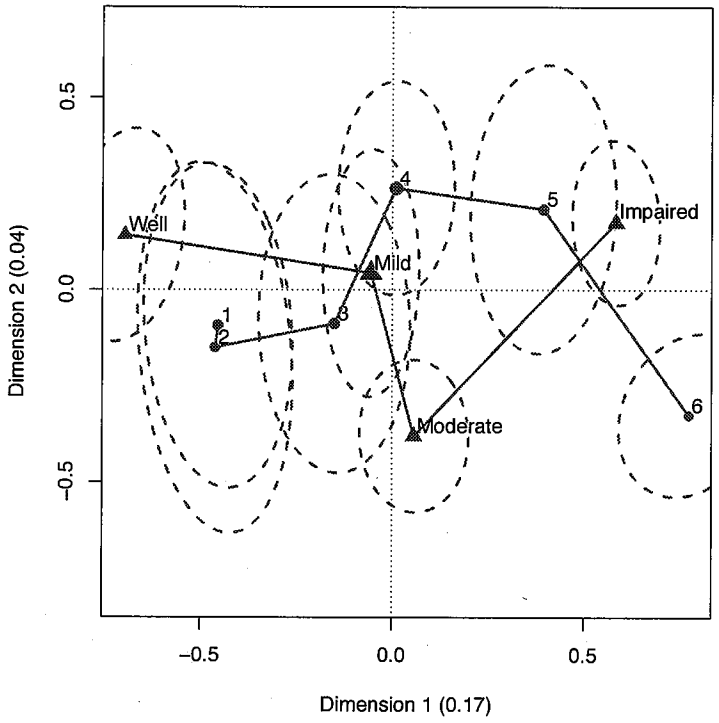


Figure 10.7: Scaled category scores for the RC(2) model fit and plotted using the logmult package. The 68% confidence ellipses correspond to bivariate ±1 confidence intervals for the category parameters.

[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.

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,

$$G^2(MH) \equiv G^2(S \mid QS) = G^2(S) - G^2(QS) \,. \tag{10.8}$$

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.

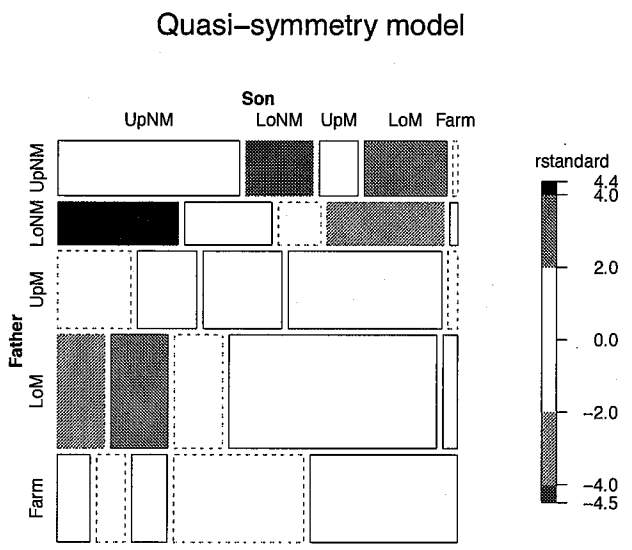


Figure 10.11: Mosaic display for the model of quasi-symmetry fit to the Hauser79 data.

```
+    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,

$$\lambda_{ij}^{AB} = \begin{cases} \prod_{k=j}^{k=i-1} \nu_k & : \; i > j \\ \prod_{k=i}^{k=j-1} \nu_k & : \; i < j \end{cases} .$$

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
```

```
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)
+   })
```

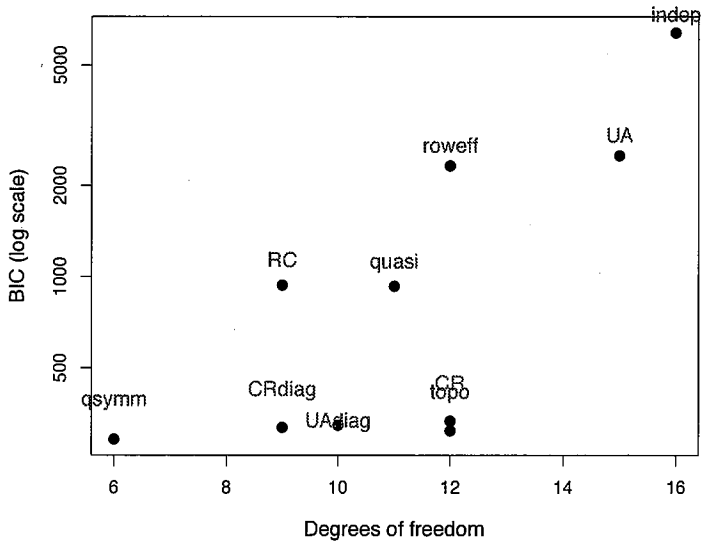


Figure 10.13: Model comparison plot for the models fit to the Hauser79 data.

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.

△

```
> 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)
> vcdExtra::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,
+        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()`.

```
> vis.hetdiag <- update(vis.indep, . ~ . + gender * Diag(right, left) +
+                       Symm(right, left))
```

```
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
```

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))
```

[9] The default, `zero=3` gives the model shown in Eqn. (10.16), with the odds ratio constant.

```
> 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)
+    )
```

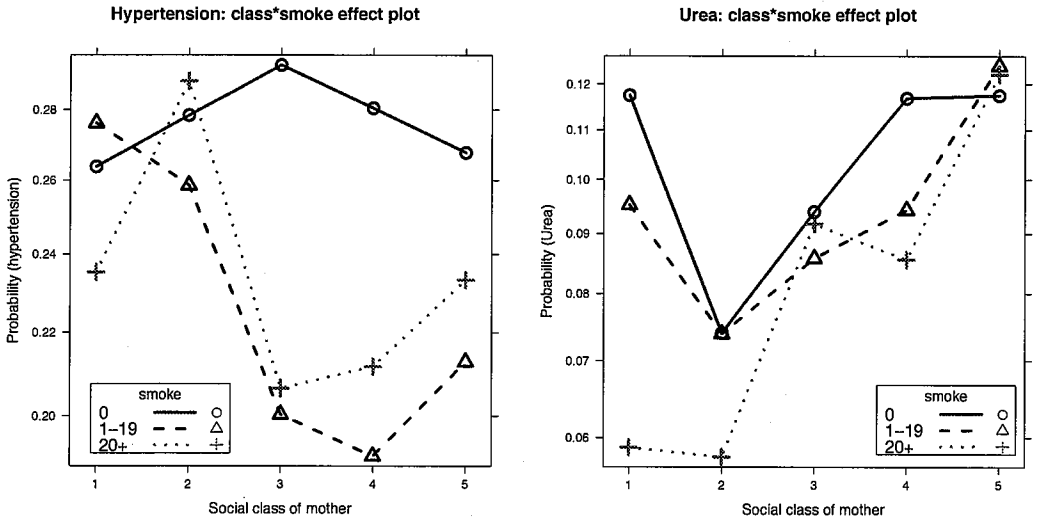


Figure 10.25: Effect plots for hypertension and urea, by social class of mother and smoking.

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))
```

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"
```

can be carried out using the Wald test statistic, $z_j = \widehat{\beta}_j / \mathrm{se}(\widehat{\beta}_j)$. 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 $\boldsymbol{c} = \mathbf{0}$. 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

$$\boldsymbol{L} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{0} & \boldsymbol{I} \end{bmatrix} , \qquad \boldsymbol{c} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} .$$

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.

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
```

It is somewhat easier to interpret the exponentiated coefficients, $\exp(\beta)$, as multiplicative effects

[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.

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 = "")
```

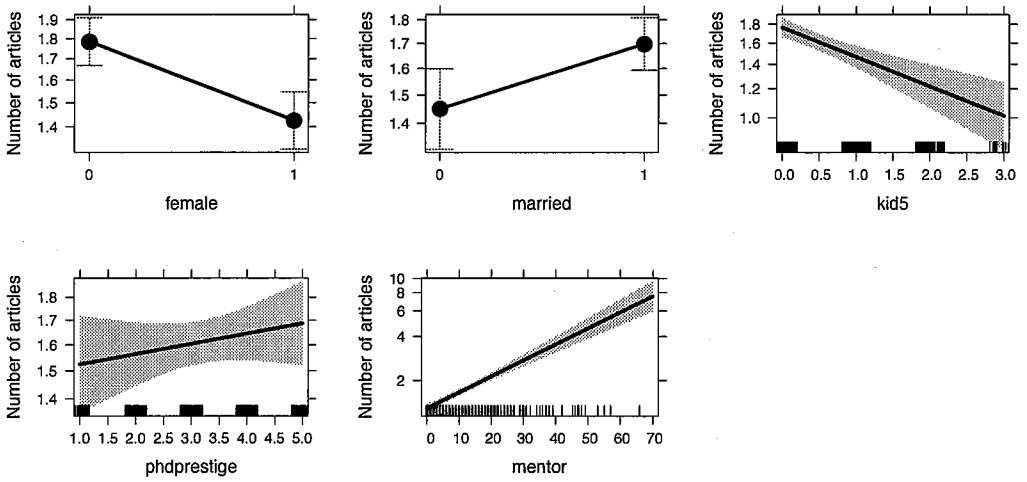


Figure 11.3: Effect plots for the predictors in the Poisson regression model for the PhdPubs data. Jittered values of the continuous predictors are shown at the bottom as rug-plots.

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)))
```

```
'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))
```

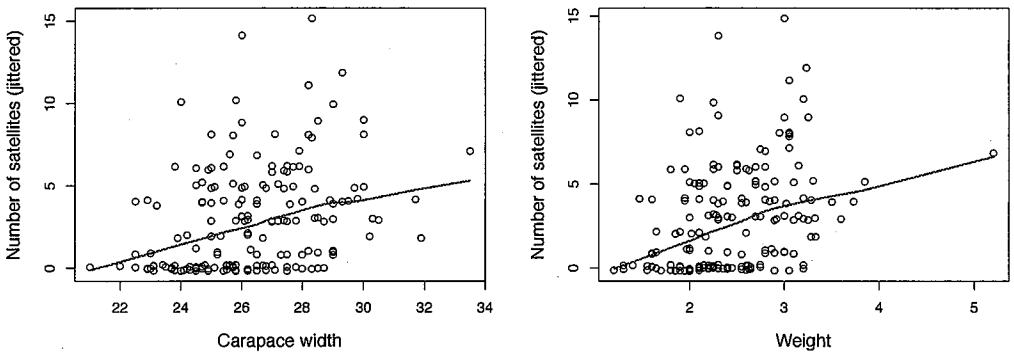


Figure 11.5: Scatterplots of number of satellites vs. width and weight, with lowess smooths.

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)")
```

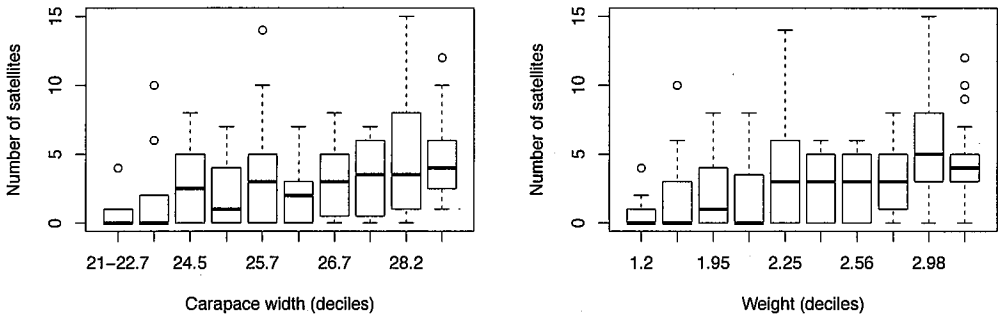


Figure 11.6: Boxplots of number of satellites vs. width and weight.

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 **
```

`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.

```
> (phi <- summary(phd.qpois)$dispersion)
[1] 1.8304
```

[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.

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 $y_i \mid \boldsymbol{x}_i$ 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 β and θ, by iteratively estimating β 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

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 ***spine plot*** 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

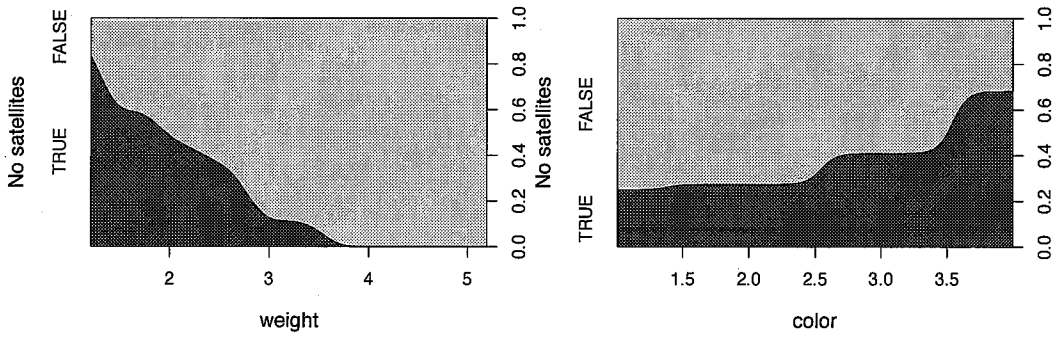


Figure 11.14: Conditional density plots for the CrabSatellites data. The region shaded below shows the conditional probability density estimate for a count of zero.

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.

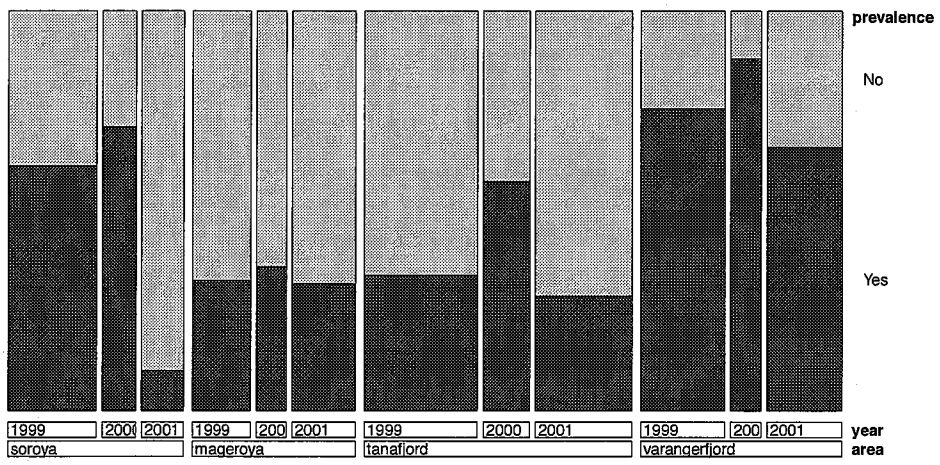


Figure 11.16: Doubledecker plot for prevalence against area and year in the CodParasites data. The cases of infected fish are highlighted.

```
> doubledecker(prevalence ~ area + year, data = cp.tab,
+              gp = shading_hcl, expected = ~ year:area + prevalence,
+              margins = c(1, 5, 3, 1))
```

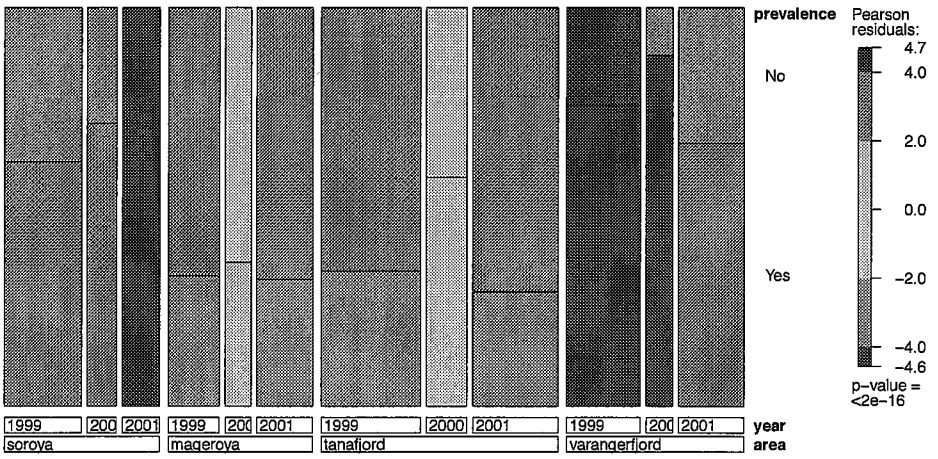


Figure 11.17: Mosaic plot for prevalence against area and year in the CodParasites data, in the doubledecker format. Shading reflects departure from a model in which prevalence is independent of area and year jointly.

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")
```

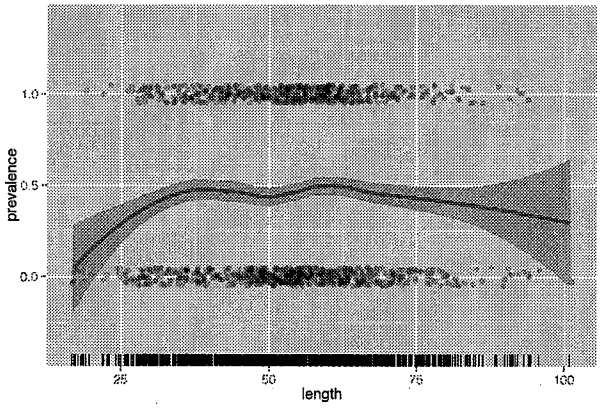


Figure 11.18: Jittered scatterplot of prevalence against length of fish, with loess smooth.

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.

```
> # plot only positive values of intensity
> CPpos <- subset(CodParasites, intensity > 0)
> ggplot(CPpos, aes(x = year, y = intensity)) +
```

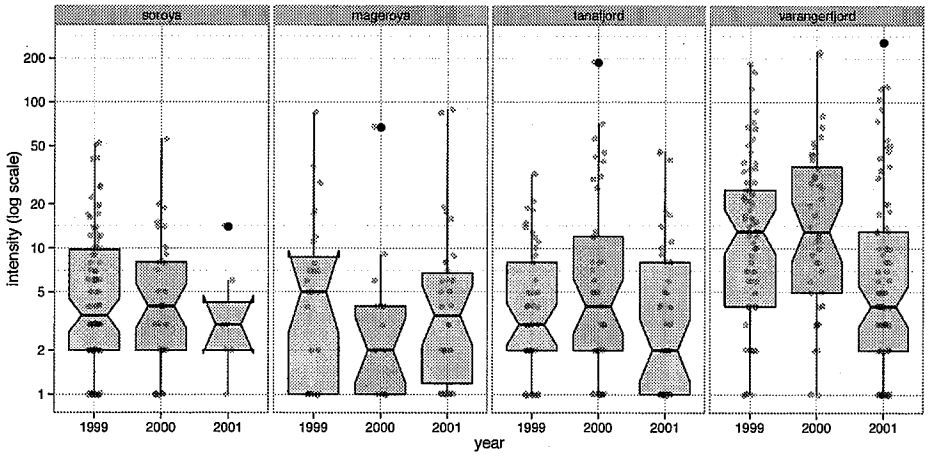


Figure 11.19: Notched boxplots for log (intensity) of parasites by area and year in the CodParasites data. Significant differences in the medians are signaled when the notches of two groups do not overlap.

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?

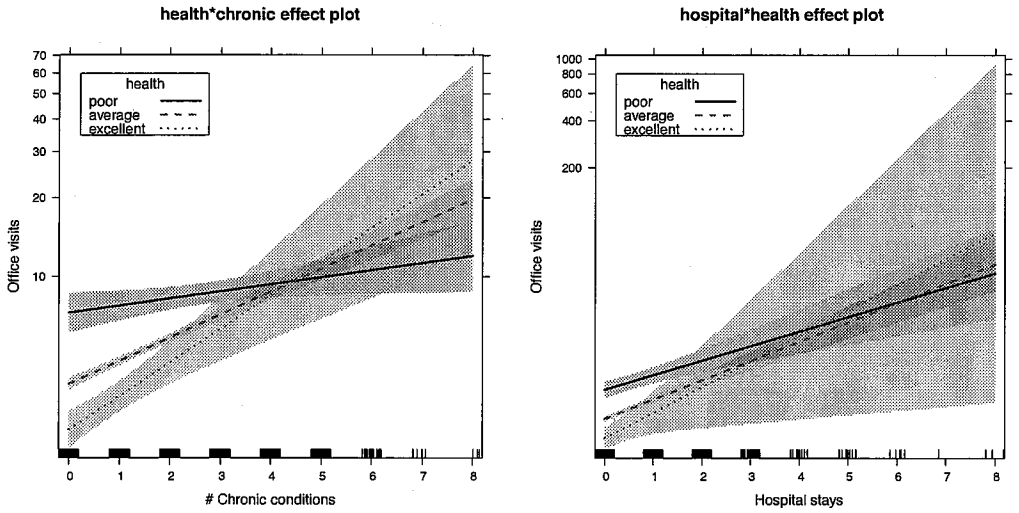


Figure 11.29: Effect plots for the interactions of chronic conditions and hospital stays with perceived health status in the model nmes.nbin2.

stays and office visits. As the number of chronic conditions increases, the relation with hospital stays decreases in slope.

```
> plot(eff_nbin2, "hospital:chronic", multiline = TRUE, ci.style = "bands",
+      ylab = "Office visits", xlab = "Hospital stays",
+      key.args = list(x = 0.05, y = .70, corner = c(0, 0), columns = 1))
>
> plot(eff_nbin2, "health:school", multiline = TRUE,  ci.style = "bands",
+      ylab = "Office visits", xlab = "Years of education",
+      key.args = list(x = 0.65, y = .1, corner = c(0, 0), columns = 1))
```

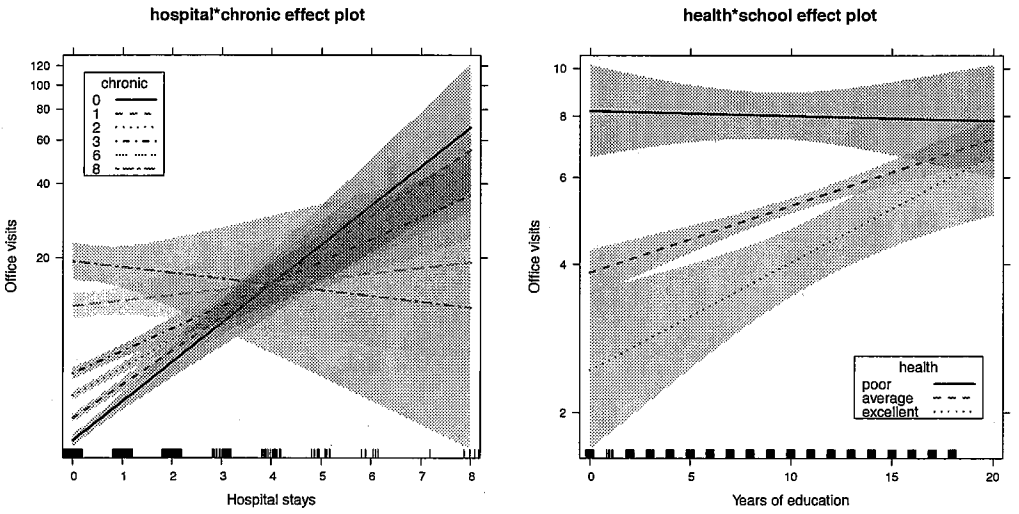


Figure 11.30: Effect plots for the interactions of chronic conditions and hospital stays and for health status with years of education in the model nmes.nbin2.

However, effect plots for this model quickly reveal a *substantive* limitation of this approach using polynomial terms. Figure 11.31 shows one such plot for the interaction of health and number of chronic conditions that you should compare with Figure 11.28.

```
> eff_nbin3 <- allEffects(nmes.nbin3,
+   xlevels = list(hospital = c(0 : 3, 6, 8),
+                  chronic = c (0 : 3, 6, 8),
+                  school = seq(0, 20, 5)))
> plot(eff_nbin3, "health : chronic", layout = c(3, 1))
```

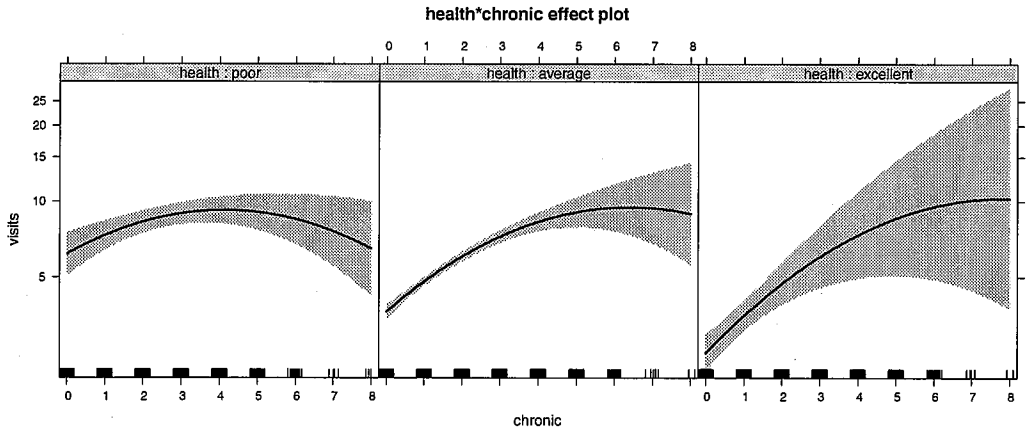


Figure 11.31: Effect plot for the interaction of health and number of chronic conditions in the quadratic model nmes.nbin3.

The quadratic fits for each level of health in Figure 11.31 imply that office visits increase with chronic conditions up to a point and then decrease—with a quadratic, what goes up must come down, the same way it went up! This makes no sense here, particularly for those with poor health status. As well, the confidence bands in this figure are uncomfortably wide, particularly at higher levels of chronic conditions, compared to those in Figure 11.28. The quadratic model is thus preferable statistically and descriptively, but serves less well for explanatory, substantive, and predictive goals.

An alternative approach to handle nonlinearity is to use regression splines (as in Example 7.9) or a ***generalized additive model*** (Hastie and Tibshirani, 1990) for these terms. The latter specifies the linear predictor as a sum of smooth functions,

$$g(\mathcal{E}(y)) = \beta_0 + f_1(x_1) + f_2(x_2) + \cdots + f_m(x_m)\,.$$

where each $f_j(x_j)$ may be a function with a specified parametric form (for example, a polynomial) or may be specified non-parametrically, simply as "smooth functions," to be estimated by non-parametric means.

In R, a very general implementation of the generalized additive model (GAM) is provided by `gam()` in the mgcv (Wood, 2015) package and described in detail by Wood (2006). Particular features of the package are facilities for automatic smoothness selection (Wood, 2004), and the provision of a variety of smooths of more than one variable. This example just scratches the surface of GAM methodology.

In the context of the NB model we are considering here, the analog of model `nmes.nbin3` fitted using `gam()` is `nmes.gamnb` shown below. The negative-binomial distribution can be specified using `family=nb()` when the parameter θ is also estimated from the data (as with

For a class "glm" object, the function `residuals(object, type)` returns the unstandardized residuals for `type="pearson"` or `type="deviance"`.[18] The standardized versions are obtained using `rstandard()`, again with a `type` argument for the Pearson or deviance flavor. `rstudent()` calculates the studentized deletion residuals.

11.6.1.3 Influence

As discussed in Section 7.5 in the context of logistic regression, influence measures attempt to evaluate the effect that an observation exerts on the parameters, fitted values, or goodness-of-fit statistics by comparing a statistic calculated for all the data with the value obtained omitting each observation in turn. Again, approximations are used to estimate these effects without laboriously refitting the model n times.

Overall measures of influence include

- Cook's distance (Eqn. (7.10)), a squared measure of the difference $\widehat{\beta} - \widehat{\beta}_{(-i)}$ in all p coefficients in the model. The approximation used in `cooks.distance()` is

$$C_i = \frac{\widetilde{r}_i h_i}{\widehat{\phi}\, p\, (1 - h_i)} .$$

  This follows Williams (1987), but scales the result by the estimated dispersion $\widehat{\phi}$ as an approximate $F_{p,n-p}$ statistic rather than χ^2_p.
- DFFITS, the standardized signed measure of the difference of the fitted value $\widehat{\mu}_i$ using all the data and the value $\widehat{\mu}_{(-i)}$ omitting observation i.

EXAMPLE 11.17: Publications of PhD candidates

For models that inherit methods from the "glm" class (including NB models fit using `glm.nb()`), the simplest initial diagnostic plots are provided by the `plot()` method. Figure 11.33 shows the default *regression quartet* of plots for the negative-binomial model `phd.nbin` examined in earlier examples. By default, the `id.n=3` most noteworthy observations are labeled with their row names from the original data set.

```
> plot(phd.nbin)
```

The plot of residuals against predicted values in the upper left panel of Figure 11.33 should show no overall systematic trend for a well-fitting model. The smoothed loess curve in red suggests that this is not the case.

Several functions in the car package make these plots more flexibly and with greater control of the details. Figure 11.34 shows the plot of residuals against predicted values two ways. The right panel explains the peculiar pattern of diagonal band of points. These correspond to the different discrete values of the response variable, number of articles published.

```
> library(car)
> residualPlot(phd.nbin, type = "rstandard", col.smooth = "red", id.n = 3)
> residualPlot(phd.nbin, type = "rstandard",
+              groups = PhdPubs$articles, key = FALSE, linear = FALSE,
+              smoother = NULL)
```

Other useful plots show the residuals against each predictor. For a good-fitting model, the average residual should not vary systematically with the predictor. As shown in Figure 11.35, `residualPlot()` draws a lowess smooth, and also computes a curvature test for each of the plots by adding a quadratic term and testing the quadratic to be zero.

[18] Other types include raw response residuals (`type="response"`), working residuals (`type="working"`), and partial residuals (`type="partial"`).

EXAMPLE 11.19: Demand for medical care

In the examples in Section 11.5.2 we considered a variety of models for the number of office visits to physicians (`visits`) as the primary outcome variable in the study of demand for medical care by the elderly. We noted that other indicators of demand included office visits to non-physicians and hospital visits to both physicians and non-physicians. A more complete analysis of this data would consider all four response indicators together.

A special feature of this example is that the four response variables constitute a 2×2 set of the combinations of *place of visit* (office vs. hospital) and (physician vs. non-physician) *practitioner*. These are all counts, and could be transformed to two binary responses according to place and practitioner. Instead, we treat them individually here.

We start by selecting the variables to consider from the *NMES1988* data, giving a new working data set `nmes2`.

```
> data("NMES1988", package = "AER")
> nmes2 <- NMES1988[, c(1 : 4, 6 : 8, 13, 15, 18)]
> names(nmes2)[1 : 4]     # responses

[1] "visits"   "nvisits"  "ovisits"  "novisits"

> names(nmes2)[-(1 : 4)]  # predictors

[1] "hospital"  "health"    "chronic"   "gender"    "school"
[6] "insurance"
```

11.7.1 Analyzing correlations: HE plots

For purely descriptive purposes, a useful starting point is often an analysis of the $\log(\boldsymbol{y})$ on the predictor variables using the classical MLM, a rough analog of a multivariate Poisson regression with a log link. Inferential statistics will be biased, but we can use the result to visualize the pairwise linear relations that exist among all responses and all predictors compactly using hypothesis-error (HE) plots (Friendly, 2007).

Zero counts cause problems because the log of zero is undefined, so we add 1 to each y_{ij} in the call to `lm()`. The result is an object of class "mlm".

```
> clog <- function(x) log(x + 1)
> nmes.mlm <- lm(clog(cbind(visits, nvisits, ovisits, novisits)) ~ .,
+                data = nmes2)
```

An HE plot provides a visualization of the covariances of effects for the linear hypothesis (H) for each term in an MLM in relation to error covariances (E) using data ellipsoids in the space of dimension q, the number of response variables. The size of each H ellipsoid in relation to the E ellipsoid indicates the strength of the linear relations between the responses and the individual predictors.[21] The orientation of each H ellipsoid shows the direction of the correlations for that term with the response variables. For 1 degree of freedom terms (a covariate or factor with two levels), the corresponding H ellipsoid collapses to a line.

The heplots (Fox and Friendly, 2014) package contains functions for 2D plots (`heplot()`) of pairs of y variables, 3D plots (`heplot3d()`), and all pairwise plots (`pairs()`). We illustrate this here using `pairs()` for the MLM model, giving the plot shown in Figure 11.40.

[21] When the errors, $\boldsymbol{E}$ in Eqn. (11.16), are approximately multivariate normal, the H ellipsoid provides a visual test of significance: the H ellipsoid projects outside the E ellipsoid *if and only if* Roy's test is significant at a chosen α level.

```
> library(heplots)
> vlabels <- c("Physician\noffice visits", "Non-physician\n office visits",
+              "Physician\nhospital visits", "Non-physician\nhospital visits")
> pairs(nmes.mlm, factor.means = "health", fill = TRUE, var.labels = vlabels)
```

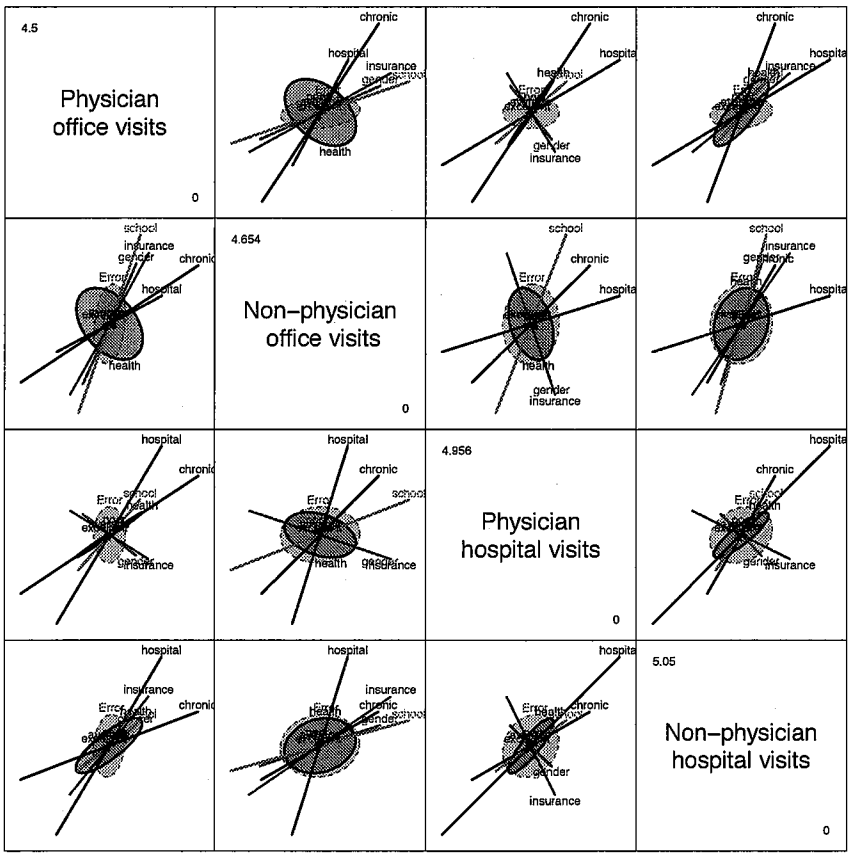


Figure 11.40: Pairwise HE plots for all responses in the nmes2 data.

The top row in Figure 11.40 shows the relationship of physician office visits to the other types of medical services. It can be seen that chronic conditions and hospital stays are positively correlated with both responses, as they also are in all other pairwise plots. Having private health insurance is positively related to some of these outcomes, and negatively to others. Except for difficulties with overlapping labels and the obvious violation of statistical assumptions of the MLM here, such plots give reasonably useful overviews on the relationships among the y and x variables.

11.7.2 Analyzing associations: Odds ratios and fourfold plots

In the analysis below, we first attempt to understand the association among these response variables and how these associations relate to the explanatory variables. It is natural to think of this in terms of the (log) odds ratio of a visit to a physician vs. a non-physician, given that the place is in an office as opposed to a hospital. Following this, we consider some multivariate negative binomial models relating these counts to the explanatory variables.

In order to treat the four response variables as a single response (`visit`), distinguished by `type`, it is necessary to reshape the data from a wide format to a long format with four rows for each input observation.

```
> vars <- colnames(nmes2)[1 : 4]
> nmes.long <- reshape(nmes2,
+   varying = vars,
+   v.names = "visit",
+   timevar = "type",
+   times = vars,
+   direction = "long",
+   new.row.names = 1 : (4 * nrow(nmes2)))
```

Then, the `type` variable can be used to create two new variables, `practitioner` and `place`, corresponding to the distinctions among visits. While we are at it, we create factors for two of the predictors.

```
> nmes.long <- nmes.long[order(nmes.long$id),]
> nmes.long <- transform(nmes.long,
+   practitioner = ifelse(type %in% c("visits", "ovisits"),
+                         "physician", "nonphysician"),
+   place = ifelse(type %in% c("visits", "nvisits"), "office", "hospital"),
+   hospf = cutfac(hospital, c(0 : 2, 8)),
+   chronicf = cutfac(chronic))
```

Then, we can use `xtabs()` to create a frequency table of `practitioner` and `place` classified by any one or more of these factors. For example, the total number of visits of the four types is given by

```
> xtabs(visit ~ practitioner + place, data = nmes.long)

             place
practitioner   hospital office
  nonphysician     2362   7129
  physician        3308  25442
```

From this, we can calculate the odds ratio and visualize the association with a fourfold or mosaic plot. More generally, by including more factors in the call to `xtabs()`, we can calculate and visualize how the *conditional* association varies with these factors. For example, Figure 11.41 shows fourfold plots conditioned by health status. It can be seen that there is a strong positive association, except for those with excellent health: people are more likely to see a physician in an office visit, and a non-physician in a hopsiptal visit. The corresponding log odds ratios are shown numerically using `loddsratio()`.

```
> library(vcdExtra)
> fourfold(xtabs(visit ~ practitioner + place + health, data = nmes.long),
+          mfrow=c(1,3))
> loddsratio(xtabs(visit ~ practitioner + place + health, data = nmes.long))

log odds ratios for practitioner and place by health

     poor   average excellent
 1.140166  0.972777  0.032266
```

Going further, we can condition by more factors. Figure 11.42 shows the fourfold plots conditioned by the number of chronic conditions (in the rows) and the combinations of gender and private insurance (columns).

```
> tab <- xtabs(visit ~ practitioner + place + gender + insurance + chronicf,
+              data=nmes.long)
> fourfold(tab, mfcol=c(4,4), varnames=FALSE)
```

The systematic patterns seen here are worth exploring further by graphing the log odds ratios directly. The call `as.data.frame(loddsratio(tab))` converts the result of `loddsratio(tab)`

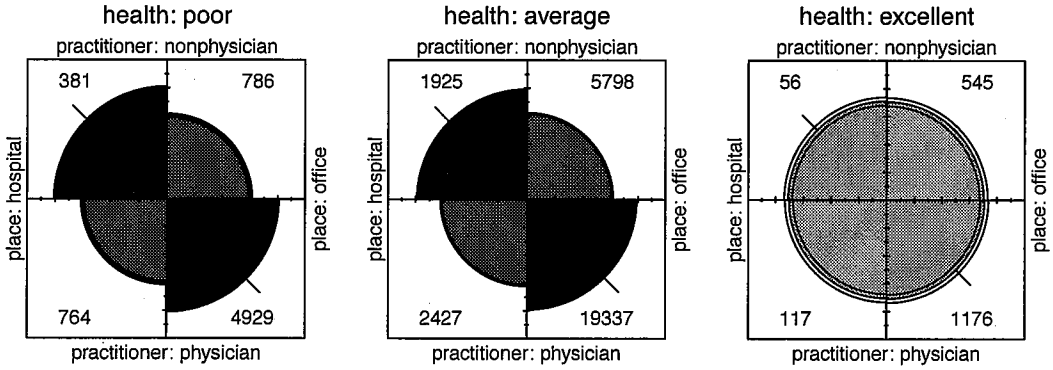


Figure 11.41: Fourfold displays for the association between practitioner and place in the nmes.long data, conditioned on health status.

to a data frame with factors for these variables, and variables LOR and ASE containing the estimated log odds ratio ($\widehat{\theta}$) and its asymptotic standard error (ASE($\widehat{\theta}$)). Figure 11.43 shows the plot of these values as line graphs with associated ± 1 error bars produced using ggplot2.[22]

```
> lodds.df <- as.data.frame(loddsratio(tab))
> library(ggplot2)
> ggplot(lodds.df, aes(x = chronicf, y = LOR,
+                      ymin = LOR - 1.96 * ASE, ymax = LOR + 1.96 * ASE,
+                      group = insurance, color = insurance)) +
+   geom_line(size = 1.2) + geom_point(size = 3) +
+   geom_linerange(size = 1.2) +
+   geom_errorbar(width = 0.2) +
+   geom_hline(yintercept = 0, linetype = "longdash") +
+   geom_hline(yintercept = mean(lodds.df$LOR), linetype = "dotdash") +
+   facet_grid(. ~ gender, labeller = label_both) +
+   labs(x = "Number of chronic conditions",
+        y = "log odds ratio (physician|place)") +
+   theme(legend.position = c(0.1, 0.9))
```

It can be seen that for those with private insurance, the log odds ratios are uniformly positive, but males and females exhibit a somewhat different pattern over number of chronic conditions. Among those with no private insurance, the log odds ratios generally increase over number of chronic conditions, except for females with 3 or more such conditions.

Beyond this descriptive analysis, you can test hypotheses about the effects of the predictors on the log odds ratios using a simple ANOVA model. Under the null hypothesis, $H_0 : \theta_{ijk\ldots} = 0$, the $\widehat{\theta}$ are each distributed normally, $\mathcal{N}(0, \text{ASE}(\widehat{\theta}))$, so a weighted ANOVA can be used to test for differences according to the predictors. This analysis gives the results below.

```
> lodds.mod <- lm(LOR ~ (gender + insurance + chronicf)^2,
+                 weights = 1 / ASE^2, data = lodds.df)
> anova(lodds.mod)

Analysis of Variance Table

Response: LOR
                  Df Sum Sq Mean Sq F value Pr(>F)
```

[22] A similar plot can be obtained using `cotabplot(~practitioner + place + chronicf + insurance | gender, tab, panel = cotab_loddsratio)`.

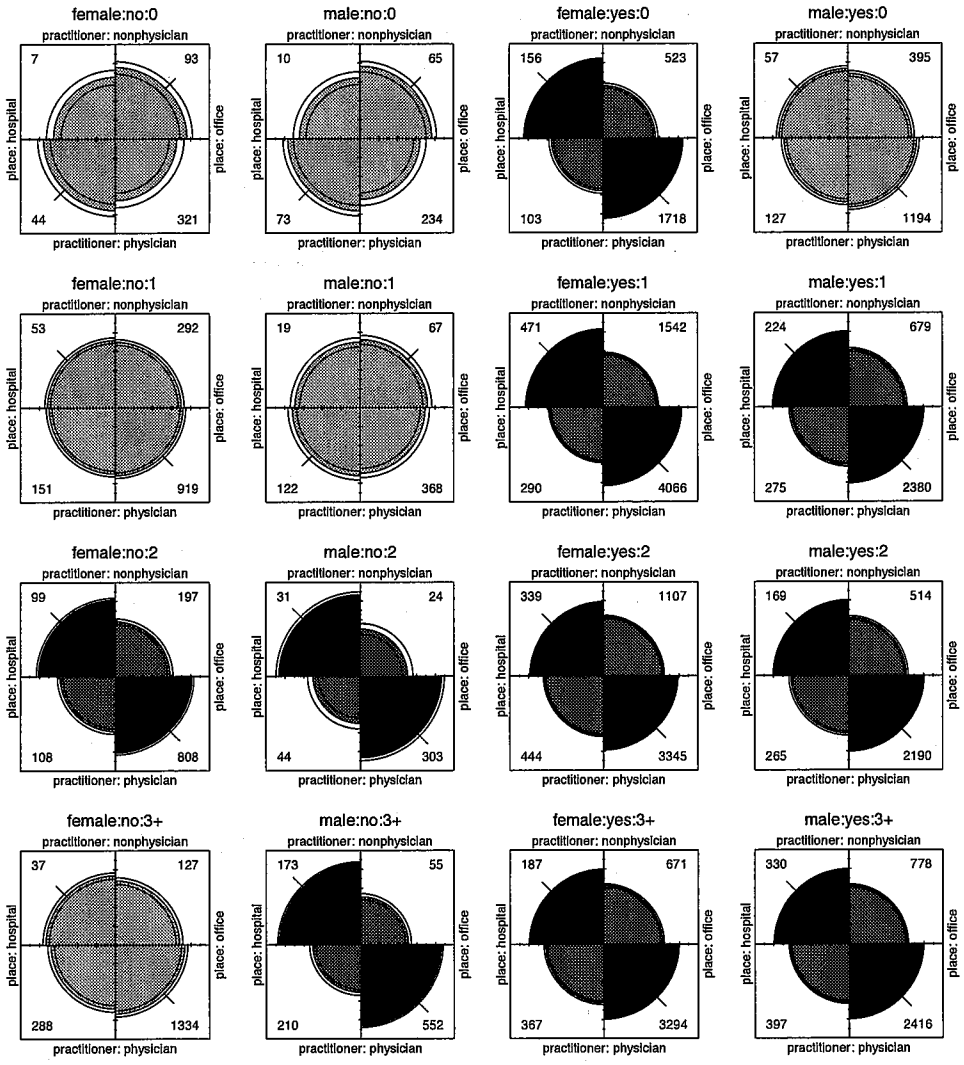


Figure 11.42: Fourfold displays for the association between practitioner and place in the nmes.long data, conditioned on gender, insurance and number of chronic conditions. Rows are levels of chronic; columns are the combinations of gender and insurance.

```
gender              1    0.8     0.8    0.17  0.707
insurance           1    5.3     5.3    1.17  0.358
chronicf            3    4.6     1.5    0.34  0.802
gender:insurance    1   32.5    32.5    7.20  0.075 .
gender:chronicf     3   54.1    18.0    3.99  0.143
insurance:chronicf  3  114.1    38.0    8.43  0.057 .
Residuals           3   13.5     4.5
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

As might be expected from the graph in Figure 11.43, having private insurance is a primary determinant of the decision to seek an office visit with a physician, but this effect interacts slightly according to number of chronic conditions and gender.

△

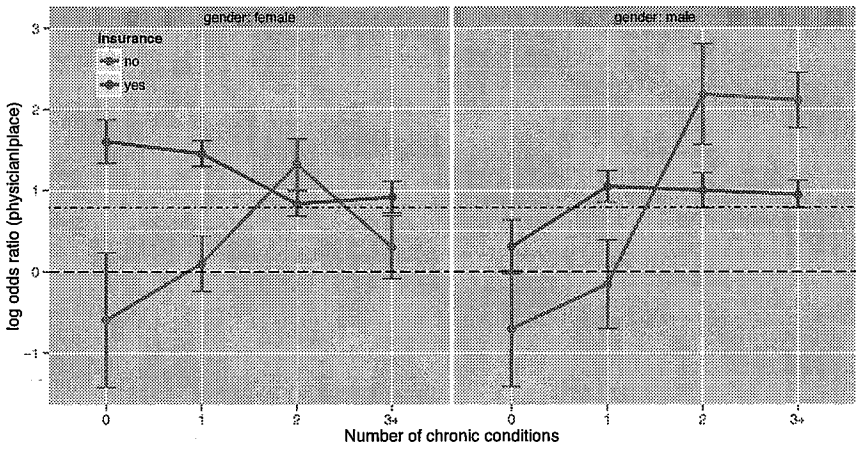


Figure 11.43: Plot of log odds ratios with 1 standard error bars for the association between practitioner and place, conditioned on gender, insurance, and number of chronic conditions. The horizontal lines show the null model (longdash) and the mean (dot–dash) of the log odds ratios.

11.7.2.1 Fitting and testing multivariate count data models

With a multivariate response, `vglm()` in the VGAM package estimates the separate coefficients for each response jointly. A special feature of this formulation is that constraints can be imposed to force the coefficients for a given term in a model to to be the same for all responses. A likelihood-ratio test against the unconstrained model can then be used to test for differences in the effects of predictors across the response variables.

This is achieved by formulating the linear predictor as a sum of terms,

$$\eta(\boldsymbol{x}) = \sum_{k=1}^{p} \boldsymbol{H}_k \boldsymbol{\beta}_k \boldsymbol{x}_k \; ,$$

where $\boldsymbol{H}_1, \ldots, \boldsymbol{H}_p$ are *known* full-rank constraint matrices. With no constraints, the $\boldsymbol{H}_k$ are identity matrices $\boldsymbol{I}_q$ for all terms. With `vglm()`, the constraint matrices for a given model are returned using `constraints()`, and can be set for a new, restricted model using the `constraints` argument. To constrain the coefficients for a term k to be equal for all responses, use $\boldsymbol{H}_k = 1_q$, a unit vector.

More general Wald tests of hypotheses can be carried out without refitting using `linearHypothesis()` in the car package. These include (a) joint tests that a subset of predictors for a given response have null effects; (b) across-response tests of equality of coefficients for one or more model terms.

EXAMPLE 11.20: Demand for medical care

In the examples in Section 11.5.2, we described a series of increasingly complex models for physician office visits, including interactions and nonlinear terms. The multivariate case is computationally more intensive, and estimation can break down in complex models. We can illustrate the main ideas here using the multivariate analog of the simple main effects model discussed in Example 11.14.

Using `vglm()`, the response variables are specified as the matrix form $\boldsymbol{Y}$ using `cbind()` on the left-hand side of the model formula. The right-hand side, `~ .` here specifies all other variables as predictors. `family = negbinomial` uses the NB model for each y_j, with an intercept-only model for the dispersion parameters by default.

  - a *link function*, $g(\bullet)$, connecting the linear predictor η_i to the mean, $\mu_i = \mathcal{E}(y_i)$, of the response variable, so that $g(\mu_i) = \eta_i$. The link function formalizes the more traditional approach of analyzing an ad-hoc transformation of y, such as $\log(y)$, $\sqrt{y}$, y^2, or Box-Cox (Box and Cox, 1964) transformations y^λ to determine an empirical optimal power transformation.
  - a *random component*, specifying the conditional distribution of $y_i \mid \boldsymbol{x}_i$ as any member of the exponential family, including the normal, binomial, Poisson, gamma, and other distributions.

- For the analysis of discrete response variables, and count data in particular, a key feature of the GLM is recognition of a *variance function* for the conditional variance of y_i, not forced to be constant, but rather allowed to depend on the mean μ_i and possibly a dispersion parameter, ϕ.

- From this background, we focus on GLMs for discrete count data response variables that extend considerably the loglinear models for contingency tables treated in Chapter 9. The Poisson distribution with a log link function is an equivalent starting point, however, count data GLMs often exhibit overdispersion in relation to the Poisson assumption that the conditional variance is the same as the mean, $\mathcal{V}(y_i \mid \eta_i) = \mu_i$.

  - One simple approach to this problem is the quasi-Poisson model, that estimates the dispersion parameter ϕ from the data, and uses this to correct standard errors and inferential tests.
  - Another is the wider class of negative-binomial models that allow a more flexible mean-variance function such as $\mathcal{V}(y_i \mid \eta_i) = \mu_i + \alpha\mu_i^2$.

- In practical application, many sets of empirical count data also exhibit a greater prevalence of zero counts than can be fit well using (quasi-) Poisson or negative-binomial models. Two simple extensions beyond the GLM class are

  - zero-inflated models, that posit a latent class of observations that always yield $y_i = 0$ counts, among the rest that have a Poisson or negative-binomial distribution including some zeros;
  - hurdle (or zero-altered) models, with one submodel for the zero counts and a separate submodel for the positive counts.

- Data analysis and visualization of count data therefore requires flexible tools and graphical methods. Some useful exploratory methods include jittered scatterplots and boxplots of $\log(y)$ against predictors enhanced by smoothed curves and trend lines, spine plots and conditional density plots. Rootograms are quite helpful in visualizing the goodness-of-fit of count data models.

- Effect plots provide a convenient visual display of the high-order terms in a possibly complex GLM. They show the fitted values of the linear predictor $\widehat{\eta}^\star = \boldsymbol{X}^\star\widehat{\beta}$, using a score matrix $\boldsymbol{X}^\star$ that varies the predictors in a given term over their range while holding all other predictors constant. It is important to recognize, however, that like any model summary these show only the fitted effects under a given model, not the data.

- Model diagnostic measures (leverage, residuals, Cook's distance, etc.) and plots of these provide important ancillary information about the adequacy of a given model as a summary of relationships in the data. These help to detect problems of violations of assumptions, unusual or influential observations or patterns that suggest that an important feature has not been accounted for.

- For multivariate response count data, there is no fully general theory as there is for the MLM with multivariate normality assumed for the errors. Nevertheless, there is a lot one can do to analyse such data combining the ideas of estimation for the separate responses with analysis of dependencies among the responses, conditioned by the explanatory variables.

11.9 Lab exercises

Exercise 11.1 Poole (1989) studied the mating behavior of elephants over 8 years in Amboseli National Park, Kenya. A focal aspect of the study concerned the mating success of males in relation to age, since larger males tend to be more successful in mating. Her data were used by Ramsey and Schafer (2002, Chapter 22) as a case study, and are contained in the Sleuth2 (Ramsey et al., 2012) package (Ramsey et al., 2012) as `case2201`.

For convenience, rename this to `elephants`, and study the relation between `Age` (at the beginning of the study) and number of successful `Matings` for the 41 adult male elephants observed over the course of this study, ranging in age from 27–52.

```
> data("case2201", package="Sleuth2")
> elephants <- case2201
> str(elephants)

'data.frame': 41 obs. of  2 variables:
 $ Age    : num  27 28 28 28 28 29 29 29 29 29 ...
 $ Matings: num  0 1 1 1 3 0 0 0 2 2 ...
```

(a) Create some exploratory plots of `Matings` against `Age` in the styles illustrated in this chapter. To do this successfully, you will have to account for the fact that `Matings` has a range of only 0–9, and use some smoothing methods to show the trend.
(b) Repeat (a) above, but now plotting `log(Matings+1)` against `Age` to approximate a Poisson regression with a log link and avoid problems with the zero counts.
(c) Fit a linear Poisson regression model for `Matings` against `Age`. Interpret the fitted model *verbally* from a graph of predicted number of matings and/or from the model coefficients. (*Hint*: Using `Age-27` will make the intercept directly interpretable.)
(d) Check for nonlinearity in the relationship by using the term `poly(Age,2)` in a new model. What do you conclude?
(e) Assess whether there is any evidence of overdispersion in these data by fitting analogous quasi-Poisson and negative-binomial models.

Exercise 11.2 The data set `quine` in MASS gives data on absenteeism from schools in rural New South Wales, Australia. 146 children were classified by ethnic background (`Eth`), age (`Age`, a factor), `Sex`, and Learner status (`Lrn`), and the number of days absent (`Days`) from school in a particular school year was recorded.

(a) Fit the all main-effects model in the Poisson family and examine the tests of these effects using `summary()` and car::`Anova()`. Are there any terms that should be dropped according to these tests?
(b) Re-fit this model as a quasi-Poisson model. Is there evidence of overdispersion? Test for overdispersion formally, using `dispersiontest()` from AER.
(c) Carry out the same significance tests and explain why the results differ from those for the Poisson model.

Exercise 11.3 The data set *AirCrash* in vcdExtra was analyzed in Exercise 5.2 and Exercise 6.3 in relation to the Phase of the flight and Cause of the crash. Additional variables include the number of Fatalities and Year. How does Fatalities depend on the other variables?

(a) Use the methods of this chapter to make some exploratory plots relating fatalities to each of the predictors.
(b) Fit a main effects poisson regression model for Fatalities, and make effects plots to visualize the model. Which phases and causes result in the largest number of fatalities?
(c) A linear effect of Year might not be appropriate for these data. Try using a natural spline term, ns(Year, df) to achieve a better, more adequate model.
(d) Use a model-building tool like add1() or MASS::stepAIC() to investigate whether there are important two-way interactions among the factors and your chosen effect for Year.
(e) Visualize and interpret your final model and write a brief summary to answer the question posed.

Exercise 11.4 Male double-crested cormorants use advertising behavior to attract females for breeding. The *Cormorants* data set in vcdExtra gives some results from a study by Meagan Mc Rae (2015) on counts of advertising males observed two or three times a week at six stations in a tree-nesting colony for an entire breeding season. The number of advertising birds was counted and these observations were classified by characteristics of the trees and nests. The goal was to determine how this behavior varies temporally over the season and spatially over observation stations, as well as with characteristics of nesting sites. The response variable is count and other predictors are shown below. See help(Cormorants, package="vcdExtra") for further details.

```
> data("Cormorants", package = "vcdExtra")
> car::some(Cormorants)

      category week station    nest height  density tree_health count
61         Pre    2      C2      no    low      few        dead     1
87         Pre    2      C4 partial   high      few        dead     1
100        Pre    2      B1    full   high moderate        dead     1
129        Pre    3      C4      no    mid      few        dead     3
133        Pre    3      B1      no    mid      few        dead     2
205 Incubation    5      C2      no   high      few        dead     4
270 Incubation    7      C3      no   high      few     healthy     3
299 Incubation    8      B2      no    mid      few        dead     1
315 Incubation    9      B2      no    mid moderate        dead     1
322       <NA>   10      C3      no    mid      few        dead     1
```

(a) Using the methods illustrated in this chapter, make some exploratory plots of the number of advertising birds against week in the breeding season, perhaps stratified by another predictor, like tree height, nest condition, or observation station. To see anything reasonable, you should plot count on a log (or square root) scale, jitter the points, and add smoothed curves. The variable category breaks the weeks into portions of the breeding season, so adding vertical lines separating those will be helpful for interpretation.
(b) Fit a main-effects Poisson GLM to these data and test the terms using Anova() from the car package.
(c) Interpret this model using an effects plot.
(d) Investigate whether the effect of week should be treated as linear in the model. You could try using a polynomial term like poly(week, degree) or perhaps better, using a natural spline term like ns(week, df) from the splines package.
(e) Test this model for overdispersion, using either a quasipoisson family or dispersiontest() in AER.

Exercise 11.5 For the *CodParasites* data, recode the area variable as an ordered factor as

Colophon

This book was produced using R version 3.2.1 (2015-06-18) and knitr (1.11). Hence, we can be assured that the code examples produced the output in the text.

At the time of writing, the principal R package versions used in examples and illustrations are listed below. Most of these were current on CRAN repositories (e.g., `http://cran.us.r-project.org/`) but some development versions are indicated in the "source" column. "R-Forge" refers to the development platform (`https://r-forge.r-project.org`) used by many package authors to prepare and test new versions. By the time you read this, most of these should be current on CRAN.

package	version	date	source
AER	1.2-4	2015-06-06	CRAN
ca	0.58	2014-12-31	CRAN
car	2.1-0	2015-09-03	CRAN
colorspace	1.2-6	2015-03-11	CRAN
corrplot	0.73	2013-10-15	CRAN
countreg	0.1-3	2015-04-18	R-Forge
directlabels	2013.6.15	2013-07-23	CRAN
effects	3.0-5	2015-08-28	local
ggparallel	0.1.2	2015-08-21	CRAN
ggplot2	1.0.1	2015-03-17	CRAN
ggtern	1.0.6.0	2015-08-03	CRAN
gmodels	2.16.2	2015-07-22	CRAN
gnm	1.0-8	2015-04-22	CRAN
gpairs	1.2	2014-03-09	CRAN
heplots	1.0-16	2015-07-13	CRAN
Lahman	3.0-1	2014-09-13	CRAN
lattice	0.20-33	2015-07-14	CRAN
lmtest	0.9-34	2015-06-06	CRAN
logmult	0.6.2	2015-04-22	CRAN
MASS	7.3-44	2015-08-30	CRAN
mgcv	1.8-7	2015-07-23	CRAN
nnet	7.3-11	2015-08-30	CRAN
plyr	1.8.3	2015-06-12	CRAN
pscl	1.4.9	2015-03-29	CRAN
RColorBrewer	1.1-2	2014-12-07	CRAN
reshape2	1.4.1	2014-12-06	CRAN
rms	4.3-1	2015-05-01	CRAN
rsm	2.7-3	2015-09-03	CRAN
sandwich	2.3-3	2015-03-26	CRAN
vcd	1.4-2	2015-08-12	local
vcdExtra	0.6-10	2015-08-05	local
VGAM	0.9-8	2015-05-11	CRAN
xtable	1.7-4	2014-09-12	CRAN

To prepare your R installation for running the examples in this book, you can use the following commands to install these packages.

```
> packages <- c("AER", "ca", "car", "colorspace", "corrplot", "countreg",
+     "directlabels", "effects", "ggparallel", "ggplot2", "ggtern",
+     "gmodels", "gnm", "gpairs", "heplots", "Lahman", "lattice", "lmtest",
+     "logmult", "MASS", "mgcv", "nnet", "plyr", "pscl", "RColorBrewer",
+     "reshape2", "rms", "rsm", "sandwich", "splines", "vcd", "vcdExtra",
+     "VGAM", "xtable")
> install.packages(packages)
```

Example Index

512 paths to the White House, 21–22

Arbuthnot data, 66–67
Arthritis treatment, 5, 38–40, 55, 119, 124–125, 128–130, 174–175, 202–203, 265–267, 273–274, 280–281
  conditional plots, 275–276
  full-model plots, 276–278
  logistic regression, 265
  Plotting logistic regression with base graphics, 268–269
  Plotting logistic regression with ggplot2, 269–270

Barley data, 19
Bartlett data on plum-root cuttings, 197–198, 204
Berkeley admissions, 4, 116, 123–124, 129–132, 143–145, 199–201, 210, 357–367
Breathlessness and wheeze in coal miners, 135–138, 404–412
British social mobility, 18
Butterfly species in Malaya, 72–73

Cod parasites, 455–465
Collapsing categories, 52–53
Corporal punishment data, 193–197, 213–214

Dayton survey, 51–52
Death by horse kick, 54–55, 69–70, 87–88, 90–91, 97, 101
Death in the ICU, 295–301, 308–311, 316–318
  Visualization, 301–303
Demand for medical care, 466–478, 489–497
Diagnosis of MS patients, 147–148, 152–153
Donner Party, 11–13, 23, 282–290, 306–308, 313–316

Employment status data, 190–193

Families in Saxony, 6, 7, 67–68, 89, 102–103, 105–106
Federalist Papers, 70–71, 91–92, 97, 103

General social survey, 40–41, 54

Hair color and eye color, 8–10, 41–42, 116–118, 139, 162–166, 225–228, 241–242
~~hair color and eye color, 166~~ included above
Hair color, eye color, and sex, 176, 178–179, 182–183
Hauser's occupational mobility table, 390–397
Health concerns of teenagers, 368–371

Interpolation options, 172
Iris data, 20

Job satisfaction, 42–43

Lifeboats on the *Titanic*, 155–156
London cycling deaths, 71–72, 79

Mammogram ratings, 151
Marital status and pre- and extramarital sex, 186–188, 198–199, 244–246
Mating of horseshoe crabs, 438–442, 444, 448–449, 453
Mental impairment and parents' SES, 118–119, 125, 228, 375–387

Plotting styles for discrete distributions, 80–81
Publications of PhD candidates, 107–108, 433–438, 443–446, 448, 480–483, 485–487