

Chapter 11

Generalized Linear Models

{ch:glm}

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

In one word, to draw the rule from experience, one must generalize; this is a necessity that imposes itself on the most circumspect observer.

Henri Poincaré, *The Value of Science: Essential Writings of Henri Poincare*

In the modern history of statistics, most developments occur incrementally, with small additions to existing models and theory that extend their range and applicability to new problems and data. Occasionally, there is a major synthesis that unites a wide class of existing methods in a general framework and provides opportunities for far greater growth.

A prime example is the theory of generalized linear models, introduced originally by Nelder and Wedderburn (1972), that extended the familiar (classical) linear models for regression and ANOVA to include related models, such as logistic regression and logit models (described in Chapter 7) and loglinear models (described in Chapter 9) and other variations as “families” within a single general system.

This approach has proved attractive because it: (a) integrates many familiar statistical models in a general theory where they are just special cases; (b) provides the basis for extending these and developing new models within the same or similar framework; (c) simplifies the implementation of these models in software, since the same algorithm can be used for estimation, inference and assessing model adequacy for all generalized linear models.

Section 11.1 gives a brief sketch of the GLM framework. The focus of this book is on visualization methods for categorical data, and the two important topics concern models and methods for binomial response data and for count data. The first of these, was described extensively in Chapter 7, with extensions to multinomial data (Section ??) and there is little to add here, except for changes in notation.

GLM models for count data, however, provide the opportunity to extend the scope of these methods beyond what was covered in Chapter 9, and this topic is introduced in Section 11.2. The GLM framework also provides the opportunity to deal with common problems of overdispersion (Section 11.3) and an overabundance of zero counts (Section 11.3), giving some new models and

visualization methods that help to understand such data in greater detail. Section 11.6 illustrates other graphical methods for diagnostic model checking, some of which were introduced in earlier chapters. Finally, Section 11.7 outlines some simple extensions of these models to handle multi-variate responses.

11.1 Components of Generalized Linear Models

{glm:components}

The motivation for the *generalized linear model* (GLM) and its structure are most easily seen by considering the classical linear model,

$$y_i = \mathbf{x}_i^\top \boldsymbol{\beta} + \epsilon_i$$

where y_i is the response variable for case i , $i = 1, \dots, n$, \mathbf{x}_i is the vector of explanatory variables or regressors, $\boldsymbol{\beta}$ is the vector of model parameters, and the ϵ_i are random errors. In the classical linear model, the ϵ_i are assumed to (a) have constant variance, σ_ϵ^2 , (b) follow a normal (Gaussian) distribution (conditional on \mathbf{x}_i), (c) be independent across observations.

Thus, Nelder and Wedderburn (1972) generalized this Gaussian linear model to consist of the following three components, by relaxing assumptions (a) and (b) above:¹

random component The conditional distribution of the $y_i | \mathbf{x}_i$, with mean $\mathcal{E}(y_i) = \mu_i$. Under classical assumptions, this is independent, normal with constant variance σ^2 , i.e., $y_i \stackrel{\text{iid}}{\sim} N(\mu_i, \sigma^2)$. In the GLM, the probability distribution of the y_i can be any member of the *exponential family*, including the normal, Poisson, binomial, gamma and others. Subsequent work has extended this framework to include multinomial distributions and some non-exponential families such as the negative binomial distribution.

systematic component The idea that the predicted value of y_i itself is a linear combination of the regressors is replaced by that of a *linear predictor*, η , that captures this aspect of linear models,

$$\eta_i = \mathbf{x}_i^\top \boldsymbol{\beta}$$

link function The connection between the mean of the response, μ_i , and the linear predictor, η_i , is specified by the *link function*, $g(\bullet)$, giving

$$g(\mu_i) = \eta_i = \mathbf{x}_i^\top \boldsymbol{\beta}$$

The link function $g(\bullet)$ must be both *smooth* and *monotonic*, meaning that it is one-to-one, so an inverse transformation, $g^{-1}(\bullet)$ exists,

$$\mu_i = g^{-1}(\eta_i) = g^{-1}(\mathbf{x}_i^\top \boldsymbol{\beta})$$

which allows us to obtain and plot the predicted values on their original scale. The link function captures the familiar idea that linear models are often estimated with a transformation of the response, such as $\log(y_i)$ for a frequency variable or $\text{logit}(y_i)$ for a binomial variable. The inverse function $g^{-1}(\bullet)$ is also called the *mean function*.

Some commonly used link functions are shown in Table 11.1. Some of these link functions have restrictions on the range of y_i to which they can be applied. For example, the square-root and log links apply only to non-negative and positive values respectively. The last four link functions in this table are for binomial data, where y_i represents the observed proportion of successes in n_i independent trials, and thus the mean μ_i represents the probability of success (symbolized by π_i in Chapter 7). Binary data are the special case where $n_i = 1$.

¹The remaining assumption of independent observations is relaxed in *generalized linear mixed models* (GLMMs), in which random effects to account for non-independence are added to the linear predictor. This allows the modeling of correlated (responses of family members), clustered (residents in different communities) or hierarchical data (patients within hospitals within regions). See: McCulloch and Neuhaus (2005) ... **TODO: other references?**

Table 11.1: Common link functions and their inverses used in generalized linear models

Link name	Function: $\eta_i = g(\mu_i)$	Inverse: $\mu_i = g^{-1}(\eta_i)$
identity	μ_i	η_i
square-root	$\sqrt{\mu_i}$	η_i^2
log	$\log_e(\mu_i)$	$\exp(\eta_i)$
inverse	μ_i^{-1}	η_i^{-1}
inverse-square	μ_i^{-2}	$\eta_i^{-1/2}$
logit	$\log_e \frac{\mu_i}{1-\mu_i}$	$\frac{1}{1+\exp(-\eta_i)}$
probit	$\Phi^{-1}(\mu_i)$	$\Phi^{-1}(\eta_i)$
log-log	$-\log_e[-\log_e(\mu_i)]$	$\exp[-\exp(-\eta_i)]$
comp. log-log	$\log_e[-\log_e(1-\mu_i)]$	$1 - \exp[-\exp(\eta_i)]$

11.1.1 Variance functions

The GLM has the additional property that, for distributions in the exponential family, the conditional variance of $y_i | \eta_i$ is a known function, $\mathcal{V}(\mu_i)$ of the mean and possibly one other parameter called the *scale parameter* or *dispersion parameter*, ϕ . Some commonly used distributions in the exponential family and their variance functions are shown in Table 11.2.

Table 11.2: Common distributions in the exponential family used with generalized linear models and their canonical link and variance functions

Family	Notation	Canonical link	Range of y	Variance function, $\mathcal{V}(\mu \eta)$
Gaussian	$N(\mu, \sigma^2)$	identity: μ	$(-\infty, +\infty)$	ϕ
Poisson	$\text{Pois}(\mu)$	$\log_e(\mu)$	$0, 1, \dots, \infty$	μ
Negative-Binomial	$\text{NBin}(\mu, \theta)$	$\log_e(\mu)$	$0, 1, \dots, \infty$	$\mu + \mu^2/\theta$
Binomial	$\text{Bin}(n, \mu)/n$	logit(μ)	$\{0, 1, \dots, n\}/n$	$\mu(1-\mu)/n$
Gamma	$G(\mu, \nu)$	μ^{-1}	$(0, +\infty)$	$\phi\mu^2$
Inverse-Gaussian	$IG(\mu, \nu)$	μ^2	$(0, +\infty)$	$\phi\mu^3$

- In the classical Gaussian linear model, the conditional variance is constant, $\phi = \sigma_\epsilon^2$.
- In the Poisson family, $\mathcal{V}(\mu_i) = \mu_i$ and the dispersion parameter is fixed at $\phi = 1$. In practice, it is common for count data to exhibit **overdispersion**, meaning that $\mathcal{V}(\mu_i) > \mu_i$. One way to correct for this is to extend the GLM to allow the dispersion parameter to be estimated from the data, giving what is called the **quasi-Poisson** family, with $\mathcal{V}(\mu_i) = \hat{\phi}\mu_i$.
- Similarly, for binomial data, the variance function is $\mathcal{V}(\mu_i) = \mu_i(1-\mu_i)/n_i$, with ϕ fixed at 1. Overdispersion often results from failures of the assumptions of the binomial model: supposedly independent observations may be correlated or clustered and the probability of success may not be constant, or vary with unmeasured or unmodeled variables.
- The gamma and inverse-Gaussian families are distributions useful for modeling a continuous and positive response variable with no upper bound (e.g., reaction time). They both have the

property that conditional variance increases with the mean, and for the inverse-Gaussian, variance increases at a faster rate. Their dispersion parameters ϕ are simple functions of their intrinsic “shape” parameters, indicated as ν in the table.

The important points from this discussion are that the GLM together with the exponential family of distributions:

- provide for simple, linear relations between the response and the predictors via the link function and the linear predictor.
- allows a very flexible relationship between the mean and conditional variance to be specified in terms of a set of known families.
- incorporates a dispersion parameter ϕ that in some cases can be estimated or tested for departure from that entailed in a given family.
- has allowed further extensions of this framework outside the exponential family, ranging from simple adjustments for statistical inference (“quasi” families, adjusted “sandwich” covariances) to separate modeling of the variance relation to the predictors.

Further details of generalized linear models are beyond the scope of this book, but the interested reader should consult Fox (2008, §15.3) and Agresti (2013, Ch. 4) for a comprehensive treatment.

11.1.2 Hypothesis tests for coefficients

GLMs are fit using maximum likelihood estimation, and implemented in software using an iterative algorithm known as *iteratively weighted least squares* that generalizes the least squares method for classical linear models. This provides estimates $\hat{\beta}$ of the model coefficients for the predictors in \mathbf{x} , as well as an estimated asymptotic (large sample) variance matrix of $\hat{\beta}$, given by

$$\mathcal{V}(\hat{\beta}) = \phi(\mathbf{X}^T \mathbf{W} \mathbf{X}) \quad (11.1)$$

where \mathbf{W} is a diagonal matrix of weights computed in the final iteration. In the standard Poisson GLM, the weight matrix is $\mathbf{W} = \text{diag}(\hat{\mu})$ and $\phi = 1$ is assumed.

Asymptotic standard errors, $\text{se}(\hat{\beta}_j)$, for the coefficients are then the square roots of the diagonal elements of $\mathcal{V}(\hat{\beta})$, and tests of hypotheses regarding an individual coefficient, e.g., $H_0 : \beta_j = 0$, can be carried out using the Wald test statistic, $z_j = \hat{\beta}_j / \text{se}(\hat{\beta}_j)$. 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 $H_0 : \mathbf{L}\beta = \mathbf{c}$, where \mathbf{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 \mathbf{c} , typically taken as $\mathbf{c} = \mathbf{0}$. The test statistic is the Wald chi-square,

$$Z^2 = (\mathbf{L}\hat{\beta} - \mathbf{c})^T [\mathbf{L}\mathcal{V}(\hat{\beta})\mathbf{L}^T]^{-1} (\mathbf{L}\hat{\beta} - \mathbf{c}) \quad (11.2)$$

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

$$\mathbf{L} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = [\mathbf{0} \quad \mathbf{I}] \quad , \quad \mathbf{c} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

²Wald tests are sometimes carried out using z^2 , which has an equivalent χ^2_1 distribution with 1 degree of freedom.

³When 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.

Similarly, to test the hypothesis that $\beta_1 = \beta_2$ in the same model, you can use $\mathbf{L} = [0, 1, -1, 0]$ and $\mathbf{c} = [0]$.⁴

In R, such tests are most conveniently carried out using `linearHypothesis()` in the `car` package. The hypothesis matrix \mathbf{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 \mathbf{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

{sec:glm-goodfit}

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 $y_i, i = 1, 2, \dots, n$ be the response and $\hat{\mu}_i = g^{-1}(\mathbf{x}_i^\top \hat{\boldsymbol{\beta}})$ the fitted mean using the estimated coefficients, having estimated variance $\hat{\omega}_i = \mathcal{V}(\hat{\mu}_i | \eta_i)$ as in Table 11.2. Then the normalized squared residual for observation i is $(y_i - \hat{\mu}_i)^2 / \hat{\omega}_i$, and the Pearson statistic is

$$X_P^2 = \sum_{i=1}^n \frac{(y_i - \hat{\mu}_i)^2}{\hat{\omega}_i}. \quad (11.3) \quad \{\text{eq:pearson}\}$$

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

$$D(\mathbf{y}, \hat{\boldsymbol{\mu}}) \equiv 2[\log_e \mathcal{L}(\mathbf{y}; \mathbf{y}) - \log_e \mathcal{L}(\mathbf{y}; \hat{\boldsymbol{\mu}})]$$

For classical linear models under normality, the deviance is simply the residual sum of squares, $\sum_i^n (y_i - \hat{\mu}_i)^2$. This has led to the deviance being taken in the GLM framework as a generalization of the sum of squares used in ANOVA, and hence, an analogous **analysis of deviance** to carry out tests for individual terms in GLMs, or to compare nested models.

In R, `anova(mod)` for the "glm" object `mod` gives *sequential* ("Type I") tests of successive terms in a model, while `Anova()` in the `car` package gives the more generally useful "Type II" (and "Type III") *partial* tests, that assess the additional contribution of each term above all others, taking marginality into account.

For Poisson models with a log link giving $\boldsymbol{\mu} = \exp(\mathbf{x}^\top \boldsymbol{\beta})$, the deviance takes the form⁵

$$D(\mathbf{y}, \hat{\boldsymbol{\mu}}) = 2 \sum_{i=1}^n \left[y_i \log_e \left(\frac{y_i}{\hat{\mu}_i} \right) - (y_i - \hat{\mu}_i) \right] \quad (11.4) \quad \{\text{eq:pois-deviance}\}$$

For a GLM with p parameters, both the Pearson and residual deviance statistics follow approximate χ_{n-p}^2 distributions with $n - p$ degrees of freedom.

⁴Such a test is only sensible if the predictors \mathbf{x}_1 and \mathbf{x}_2 are on the same scale, so their coefficients are commensurable.

⁵In the context of the loglinear models discussed in Section 9.3.2, this is also referred to as the likelihood-ratio G^2 statistic.

11.1.4 Comparing non-nested models

{sec:glm-nonnest}

The flexibility of the GLM and its extensions allows us to fit models to the same data using different families and different link functions, and to fit models that allow for overdispersion (Section 11.3) or that make special provisions for zero counts (Section 11.4). One price paid for this additional versatility is that standard LR tests and F tests (such as provided by `anova()` and `linearHypothesis()` in the `car` package) do not apply to models that are not nested, that is, where one model cannot be represented as a restricted, special case of another.

For models estimated by maximum likelihood, one general route to comparing non-nested models is through the AIC information criterion proposed initially by Akaike (1973) and the related BIC criterion (Schwartz, 1978) based on the fitted log-likelihood function.

$$\text{AIC} = -2 \log_e \mathcal{L} + 2k \quad (11.5)$$

$$\text{BIC} = -2 \log_e \mathcal{L} + \log_e(n)k \quad (11.6)$$

As noted in Section 9.3.2, these both penalize models with larger k , the number of parameters in the model, with BIC adding a greater penalty with larger sample size. However, because they are based only on the maximized log-likelihood, they are agnostic as to whether models are nested or not, and give comparable results (lower is better) provided the same observations have been used in all models.

In R, these results are given for a collection of models by the generic functions `AIC()` and `BIC()`; these can be calculated for any model for which `logLik()` and (for BIC) `nobs()` methods exist. The `vcdExtra` function `LRstats()` is a convenient wrapper for these methods.

AIC and BIC do not give significance tests for assessing whether one model is significantly “better” than another. One test that *does* this was proposed by Vuong (1989), unsurprisingly called **Vuong’s test**. The test is based on comparing the predicted probabilities or the pointwise log-likelihoods of the two models, and tests the null hypothesis that each is equally close to the saturated model, against the alternative that one model is closer.

For two such models, let $f_1(y_i | \mathbf{x}_i, \boldsymbol{\theta}_1)$ be the density function under model 1, with parameters $\boldsymbol{\theta}_1$ and similarly $f_2(y_i | \mathbf{x}_i, \boldsymbol{\theta}_2)$ under model 2 with parameters $\boldsymbol{\theta}_2$, where $f_1(\bullet)$ and $f_2(\bullet)$ need not be the same. Vuong’s test compares these based on the observation-wise log-likelihood ratios,

$$\ell_i = \log_e \left(\frac{f_1(y_i | \mathbf{x}_i, \hat{\boldsymbol{\theta}}_1)}{f_2(y_i | \mathbf{x}_i, \hat{\boldsymbol{\theta}}_2)} \right)$$

The test statistic is

$$V = \frac{\bar{\ell} - \text{penalty}}{\sqrt{ns_\ell}}$$

where $\bar{\ell}$ is the mean of the ℓ_i , s_ℓ is their variance, and penalty is an adjustment for model parsimony, typically taken as $\log(n)(k_1 - k_2)/2$ when model 1 has k_1 parameters in $\boldsymbol{\theta}_1$ and model 2 has k_2 parameters in $\boldsymbol{\theta}_2$.

The test statistic V has an asymptotic normal $N(0, 1)$ distribution, and is directional, with large positive values favoring model 1, and large negative values favoring model 2. This test is implemented as the `vuong()` function in the `pscl` package.

11.2 GLMs for count data

{sec:glm-count}

The prototypical GLM for count data, where the response y_i takes on non-negative values $0, 1, 2, \dots$, uses the Poisson family with the log link. We used this model extensively throughout all of Chapter 9. There the focus was on the special case of the loglinear model applied largely to contingency

tables, where the loglinear model could be seen as a fairly direct extension of ANOVA models for a quantitative response applied to the log of cell frequency.

The advantage there was that models for two-way, three-way and by implication n -way tables could be discussed and illustrated using notation and graphs that separated the parameters and effects for one-way terms (“main effects”), two-way terms (“simple associations”) and higher-way terms (“conditional associations”).

The disadvantage is that these models as formulated there do not easily accommodate general quantitative predictors and were limited to the log link and the Poisson family. For example, the models discussed in Section 10.1 for ordinal variables allow one or more table factors to be assigned quantitative scores or have such scores estimated from the data, as in `RC()` models (Section 10.1.2). Yet, the contingency table approach for loglinear models breaks down if there are continuous predictors, and count data often exhibits features that make the equivalent Poisson regression model unsuitable or incomplete. We consider some extended models here.

{ex:phdpubs1}

EXAMPLE 11.1: Publications of PhD candidates

In Example 3.24 we considered the distribution of the number of publications by PhD candidates in their last three years of study, but without taking any available predictors into account. For these data, a simple calculation shows why the Poisson distribution is unsuitable (for the marginal distribution), because the variance is 2.19 times the mean.

```
> data("PhdPubs", package="vcdExtra")
> with(PhdPubs, c(mean=mean(articles), var=var(articles),
+               ratio=var(articles)/mean(articles)))

    mean    var  ratio
1.6929 3.7097 2.1914
```

The earlier example showed rootograms (in Figure 3.25) of the number of articles, but here it is useful to consider some more basic exploratory displays. A basic barplot of the frequency distribution of number of articles published is shown in the left panel of Figure 11.1. A quick look indicates that the distribution is highly skewed and there is a large number of counts of zero.

Another problem is that the frequencies of 0–2 articles account for over 75% of the total, so that the frequencies of the larger counts get lost in the display. The rootogram corrects for this by plotting frequency on the square-root scale. However, because we are contemplating a model with a log link, the same goal can be achieved by plotting log of frequency, as shown in the right panel of Figure 11.1. To accommodate the zero frequencies, the plot shows $\log(\text{Frequency}+1)$, avoiding errors from $\log(0)$. It can be seen that log frequency decreases steadily up to 7 articles and then levels off approximately.

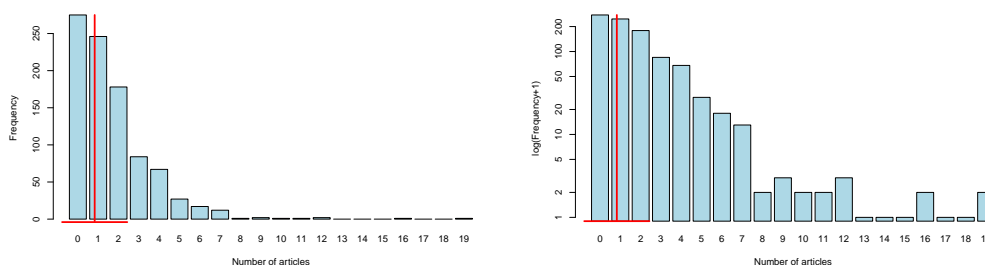


Figure 11.1: Barplots showing the frequency distribution of number of publications by PhD candidates. Left: raw scale; Right: a log scale makes the smaller counts more visible. The vertical red lines show the mean and horizontal lines show mean ± 1 standard deviation.

{fig:phdpubs-barplot}

These plots are produced as shown below. The frequency distribution of articles can be tabulated by `table()`, but there is a subtle wrinkle here: By default, `table()` excludes the values of articles that do not occur in the data (zero frequencies). To include all values in the entire range, it is necessary to treat articles as a factor with levels `0:19`.

```
> art.fac <- factor(PhdPubs$articles, levels=0:19) # include zero frequencies
> art.tab <- table(art.fac)
> art.tab
```

art.fac	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
	275	246	178	84	67	27	17	12	1	2	1	1	2	0	0	0	1	0	0	1

Then, the basic plot on the frequency scale is created using `barplot()`, and some annotations showing the mean and a one standard deviation interval can be added using standard plotting tools.

```
> barplot(art.tab, xlab="Number of articles", ylab="Frequency",
+         col="lightblue")
> abline(v=mean(PhdPubs$articles), col="red", lwd=3)
> ci <- mean(PhdPubs$articles)+c(-1,1) * sqrt(var(PhdPubs$articles))
> lines(x=ci, y=c(-4, -4), col="red", lwd=3, xpd=TRUE)
```

Similarly, the plot on the log scale in the right panel of Figure 11.1 is produced with `barplot()`, but using `art.tab+1` to start frequency at one and `log="y"` to scale the vertical axis to log.

```
> barplot(art.tab+1, ylab="log(Frequency+1)", xlab="Number of articles",
+         col="lightblue", log="y")
```

Other useful exploratory plots for count data include boxplots of the response (on a log scale) and scatterplots against continuous predictors, where jittering the response is often necessary to avoid overplotting and a smooth nonparametric curve can show possible non-linearity. The `log="y"` option is again handy, and the formula method allows adding a start value to the response. Figure 11.2 illustrates these ideas, for the factor `married` and the covariate `mentor`.

```
> boxplot(articles+1 ~ married, data=PhdPubs, log="y", varwidth=TRUE,
+         ylab="log(articles+1)", xlab="married", cex.lab=1.25)
> plot(jitter(articles+1) ~ mentor, data=PhdPubs, log="y",
+       ylab="log(articles+1)", cex.lab=1.25)
> lines(lowess(PhdPubs$mentor, PhdPubs$articles+1), col="blue", lwd=3)
```

It can be seen that the distribution of articles for married and non-married are quite similar, except that for the married students there are quite a few observations with a large number of publications. The relationship between `log(articles)` and `mentor` publications seems largely linear except possibly at the very low end. The large number of zero counts at the lower left corner stands out; this would not be seen without jittering.

Plots similar to those in Figure 11.2 can also be produced using `ggplot2` with greater flexibility, but perhaps greater effort to get the details right. One key feature is the use of `scale_y_log10()` to plot the response, and all other features on a log scale. The following code gives a plot similar to the right panel of Figure 11.2, but also plots a confidence band around the smoothed curve, and adds a linear regression line of `log(articles)` on `mentor` publications. This plot is not shown here, but it is a good exercise to reproduce it for yourself.

```
> ggplot(PhdPubs, aes(mentor, articles+1)) +
+   geom_jitter(position=position_jitter(h=0.05)) +
+   stat_smooth(method="loess", size=2, fill="blue", alpha=0.25) +
+   stat_smooth(method="lm", color="red", size=1.25, se=FALSE) +
+   scale_y_log10(breaks=c(1,2,5,10,20)) +
+   labs(y = "log (articles+1)", x="Mentor publications")
```

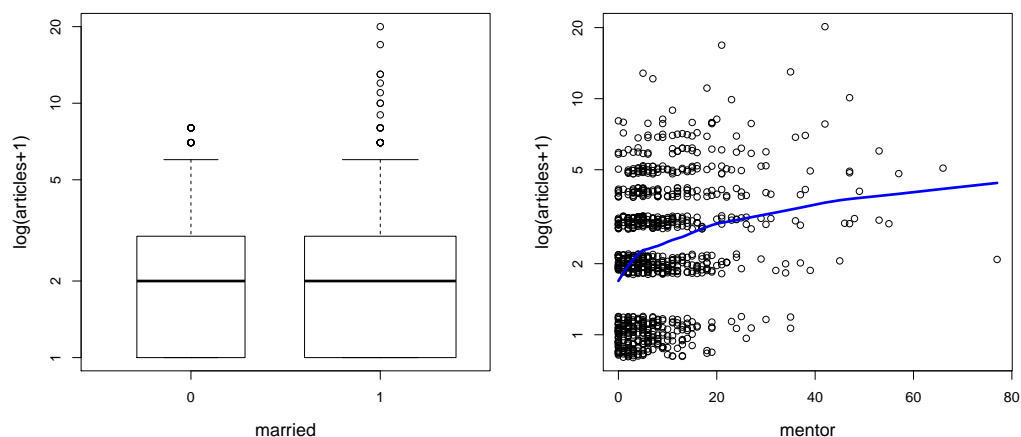


Figure 11.2: Exploratory plots for the number of articles in the PhdPubs data. Left: boxplots for married (1) vs. non-married (0); right: jittered scatterplot vs. mentor publications with a lowess smoothed curve.

g:phdpubs-logplots}

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()`, but with abbreviated output. **TODO:** `output.lines=9:24` works in `chapter.Rnw`, but not in `book.Rnw`. Why???

```
> 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.⁶

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 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="")
```

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 were truly prolific.

You should note that the panels for the predictors in Figure 11.3 are scaled individually for the range of the fitted main effects. This is often a sensible default and all predictors except `mentor` give a similar range here. To make all of these plots strictly comparable, provide a `ylim` argument, giving the range of the response on the log scale, as below (but not shown here).

```
> plot(allEffects(phd.pois), band.colors="blue", ylim=c(0, log(10)))
```

All of the above is useful, but still leaves aside the question of how well the Poisson model fits

⁶It 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.

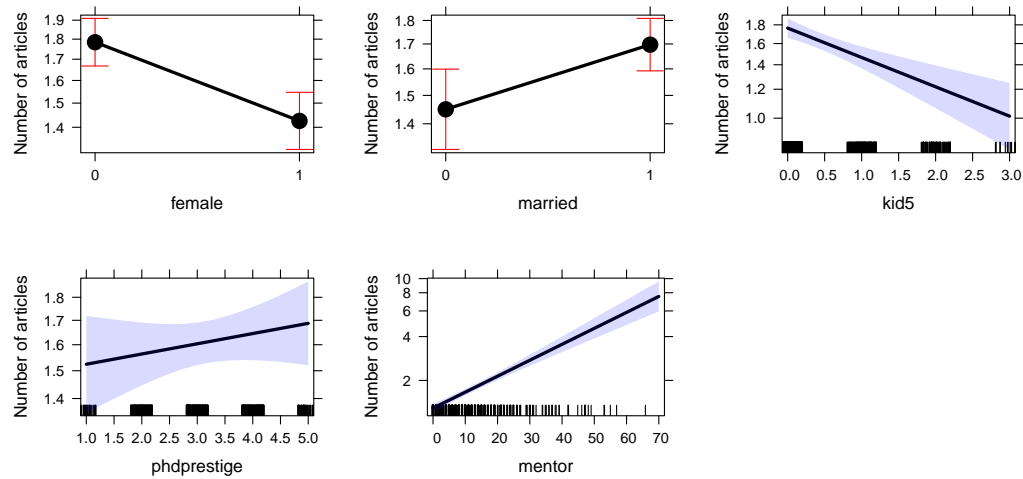


Figure 11.3: Effect plots for the predictors in the Poisson regression model for the PhdPubs data. Jittered values of the continuous predictors are shown at the bottom as rug-plots.

```
g:phdpubs1-effpois}
```

the data. The output from `summary(phd.pois)` above showed that the Poisson model fits quite badly. The residual deviance of 1633.6 with 909 degrees of freedom is highly significant.

△

```
{ex:crabs1}
```

EXAMPLE 11.2: Mating of horseshoe crabs

Brockmann (1996) studied the mating behavior of female horseshoe crabs in the Gulf of Mexico. In the mating season, crabs arrive on the beach in female/male pairs to lay and fertilize eggs. However, unattached males, called “satellites,” also come to the beach, crowd around the nesting couples and compete with attached males for fertilizations, contributing to reproductive success. Some females are ignored by satellite males, and some attract more satellites than others, and the question is: what factors contribute to the number of satellites for each female? Or, perhaps better, how do unattached males choose among available females? This is another example in which zero counts may require special treatment.

The data, given in *CrabSatellites* in the `countreg` package, give the response variable, satellites for 173 females. Possible predictors are the female’s color and spine condition, given as ordered factors, as well as her weight and carapace (shell) width.

```
> data("CrabSatellites", package = "countreg")
> str(CrabSatellites)

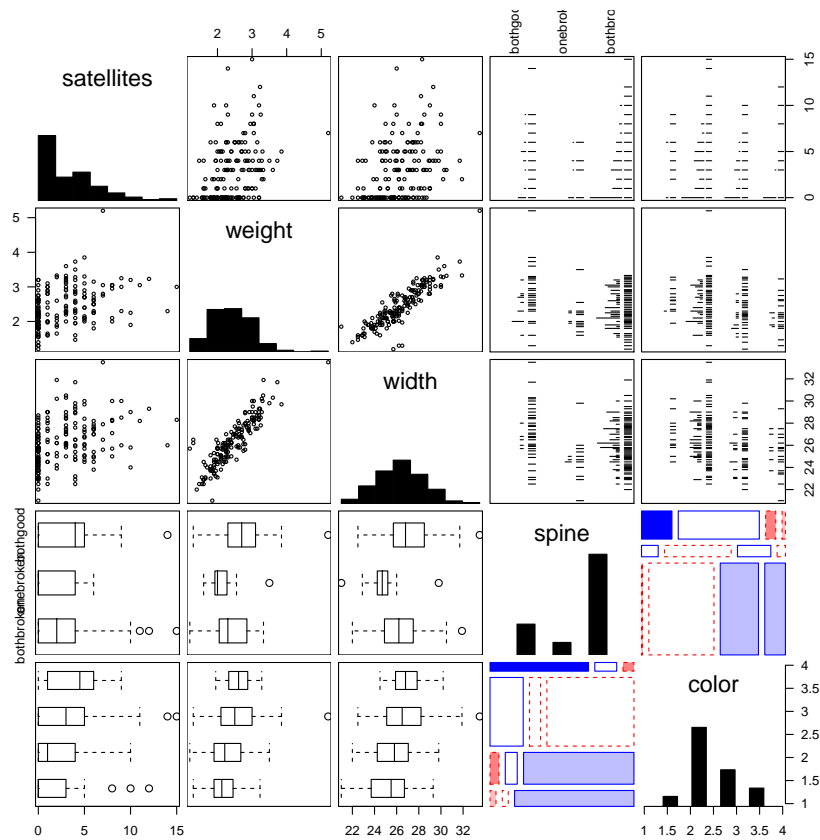
'data.frame': 173 obs. of  5 variables:
 $ color      : Ord.factor w/ 4 levels "lightmedium"<..: 2 3 3 4 2 1 4 2 2 2 ...
 $ spine      : Ord.factor w/ 3 levels "bothgood"<"onebroken"<..: 3 3 3 2 3 2 3 3 1 3 ...
 $ width      : num  28.3 26 25.6 21 29 25 26.2 24.9 25.7 27.5 ...
 $ weight     : num  3.05 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, §4.3) analyses 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),
+       mosaic.pars = list(gp=shading_Friendly, gp_args=list(interpolate=1:4)))
```



{fig:crabs1-gpairs} **Figure 11.4:** Generalized pairs plot for the CrabSatellites data.

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))
```

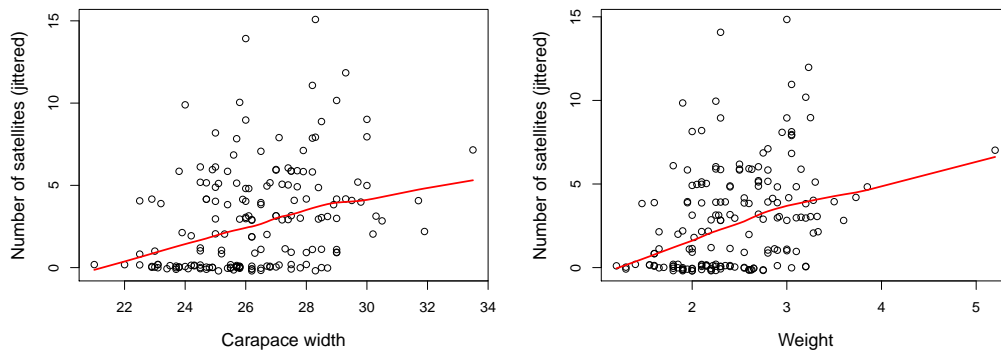


Figure 11.5: Scatterplots of number of satellites vs. width and weight, with lowess smooths.

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)")
```

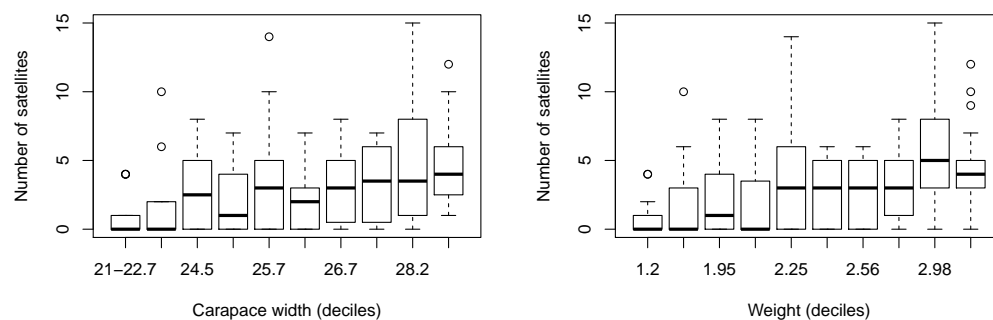


Figure 11.6: Boxplots of number of satellites vs. width and weight.

{fig:crabs1-boxplots}

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. **TODO:** `output.lines=9:26` works in `chapter.Rnw`, but not in `book.Rnw`. Why???

```
> crabs.pois <- glm(satellites ~ ., data=CrabSatellites, family=poisson)
> summary(crabs.pois)
```

Call:
glm(formula = satellites ~ ., family = poisson, data = CrabSatellites)

Deviance Residuals:

Min	1Q	Median	3Q	Max
-3.029	-1.863	-0.599	0.933	4.945

Coefficients:

	Estimate	Std. Error	z value	Pr(> z)
(Intercept)	-0.7057	0.9344	-0.76	0.4501
color.L	-0.4120	0.1567	-2.63	0.0085 **
color.Q	0.1237	0.1231	1.00	0.3150
color.C	0.0481	0.0914	0.53	0.5983
spine.L	0.0618	0.0848	0.73	0.4660
spine.Q	0.1585	0.1609	0.99	0.3244
width	0.0165	0.0489	0.34	0.7358
weight	0.4971	0.1663	2.99	0.0028 **

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

Null deviance: 632.79 on 172 degrees of freedom
Residual deviance: 549.56 on 165 degrees of freedom
AIC: 920.9

Number of Fisher Scoring iterations: 6

The Wald tests for the coefficients show that only the linear effect of color and the effect of width are significant. Effect plots, in Figure 11.7, show the nature of these effects—lighter colored females attract more satellites, as do wider and heavier females.

```
> plot(allEffects(crabs.pois), main="")
```

A simpler model can be constructed using color as a numeric variable, and either width or weight to represent female size. We choose weight here.⁷

```
> CrabSatellites1 <- transform(CrabSatellites,
+   color = as.numeric(color))
>
> crabs.pois1 <- glm(satellites ~ weight + color, data=CrabSatellites1,
+   family=poisson)
> summary(crabs.pois1)
```

Call:
glm(formula = satellites ~ weight + color, family = poisson,
data = CrabSatellites1)

Deviance Residuals:

Min	1Q	Median	3Q	Max
-2.978	-1.916	-0.547	0.918	4.834

⁷Agresti (2013, §4.3) and others who have analyzed this example uses carapace width as the main quantitative predictor, possibly because width might be more biologically salient to the single males than weight. This is a case where two highly correlated predictors are each strongly related to the outcome, yet partial tests (controlling for all others) may prefer one over the other.

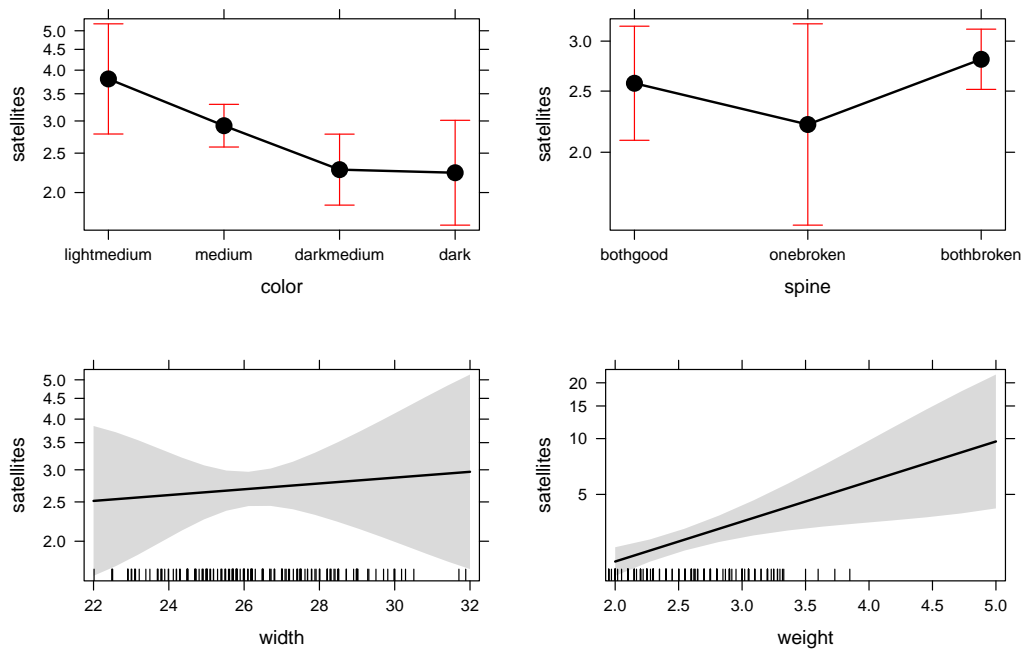


Figure 11.7: Effect plots for the predictors in the Poisson regression model for the CrabSatellites data.

```

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)   0.0888    0.2544   0.35   0.727
weight        0.5458    0.0675   8.09   6e-16 ***
color        -0.1728    0.0615  -2.81   0.005 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 632.79  on 172  degrees of freedom
Residual deviance: 552.77  on 170  degrees of freedom
AIC: 914.1

Number of Fisher Scoring iterations: 6

```

From the statistical and graphical analysis so far, the answer to the question posed in this example is clear: unattached male horseshoe crabs prefer light-colored big, fat mamas!

Yet, neither of these models fit well, as can be seen from their residual deviances and likelihood-ratio tests.

```

> vcdExtra::LRstats(crabs.pois, crabs.pois1)

Likelihood summary table:
            AIC BIC LR Chisq  Df Pr(>Chisq)
crabs.pois  921 946    550 165    <2e-16 ***
crabs.pois1  914 924    553 170    <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```


Perhaps there is something else to be learned here.

△

11.3 Models for overdispersed count data

In practice, the Poisson model is often very useful for describing the relationship between the mean μ_i and the linear predictors, but typically underestimates the variance in the data. The consequence is that the Poisson standard errors are too small, rendering the Wald tests of coefficients, $z_j = \hat{\beta}_j / \text{se}(\hat{\beta}_j)$ (and other hypothesis test statistics) too large, and thus overly liberal.

In applications of the GLM, overdispersion is usually assessed by the likelihood-ratio test of the deviance (or the Pearson statistic) given in Section 11.1.3, but there is a subtle problem here. Lack of fit in a GLM for count data can result either from a mis-specified model for the systematic component (omitted or unmeasured predictors, non-linear relations, etc.) or from failure of the Poisson mean = variance assumption. Thus, use of these methods requires some high degree of confidence that the systematic part of the model has been correctly specified, so that any lack of fit can be attributed to overdispersion.

One way of dealing with this is to base inference on so-called *sandwich* covariance estimators that are robust against some types of model mis-specification. In R, this is provided by the `sandwich()` function in the `sandwich` package, and can be used with `coefest(model, vcov=sandwich)` to give overdispersion-corrected hypothesis tests. 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 statistics, that gives a method-of-moments estimate with improved statistical properties.

$$\hat{\phi} = \frac{X_P^2}{n - p} = \sum_{i=1}^n \frac{(y_i - \hat{\mu}_i)^2}{\hat{\mu}_i} / (n - p)$$

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 $\hat{\phi}^{1/2}$ 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 $\hat{\phi}$ can be obtained as the dispersion value of the `summary()` method for a quasi-Poisson model.

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

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

{sec:glm-negbin}

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 $y_i | x_i$ where the mean μ_i for fixed 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 $\alpha \rightarrow 0$, $\mathcal{V}(y_i) \rightarrow \mu_i$ 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.

{ex:crabs-nbin}

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 $\hat{\theta}$ returned by `glm.nb()` is not very far from 1. Hence, we might also consider fixing $\theta = 1$, as illustrated below.

```
> crabs.nbin1 <- glm(satellites ~ weight + color, data=CrabSatellites1,
+ family=negative.binomial(1))
```

△

11.3.3 Visualizing the mean–variance relation

The quasi-Poisson and negative-binomial models have different variance functions, and one way to visualize which provides a better fit to the data is to group the data according to the fitted value of the linear predictor, calculate the mean and variance for each group, and then plot the variances against the means. A smoothed curve will then approximate the *empirical* mean–variance relationship. To this, we can add curves showing the mean–variance function implied by various models.⁸

{ex:phdpubs3}

EXAMPLE 11.5: Publications of PhD candidates

For the *PhdPubs* data, the fitted values are obtained with `fitted()` for the Poisson and negative binomial models. Either set can be used to categorize the observations into groups for the purpose of calculating means and variances of the response.

```
> fit.pois <- fitted(phd.pois, type="response")
> fit.nbin <- fitted(phd.nbin, type="response")
```

Here we use a simpler version of the `cutfac()` function to group a numeric variable into quantile-based groups. `cutq()` also uses deciles by default, and just uses simple integer values for the factor labels.

```
> cutq <- function(x, q = 10) {
+   quantile <- cut(x, breaks = quantile(x, probs = 0:q/q),
+   include.lowest = TRUE, labels = 1:q)
+   quantile
+ }
```

Using this, we create a variable `group` giving 20 quantile groups of the fitted values, and then use `aggregate()` to find the mean and variance of the number of articles in each group.

```
> group <- cutq(fit.nbin, q=20)
> qdat <- aggregate(PhdPubs$Articles,
+   list(group),
+   FUN = function(x) c(mean=mean(x), var=var(x)))
> qdat <- data.frame(qdat$x)
> qdat <- qdat[order(qdat$mean),]
```

We can then calculate the theoretical variances implied by the quasi-Poisson and negative-binomial models:

⁸This idea and the example that follows was suggested by Germán Rodrigues in a Stata example given at <http://data.princeton.edu/wws509/stata/overdispersion.html>.

```
> phi <- summary(phd.qpois)$dispersion
> qdat$qvar <- phi * qdat$mean
> qdat$nbvar <- qdat$mean + (qdat$mean^2) / phd.nbin$theta
> head(qdat)
```

	mean	var	qvar	nbvar
1	0.61224	0.78401	1.1206	0.7776
2	1.14894	1.78168	2.1030	1.7312
8	1.24444	2.46162	2.2778	1.9276
4	1.26087	1.70821	2.3079	1.9622
6	1.27273	1.83087	2.3296	1.9873
7	1.29787	4.34413	2.3756	2.0409

The plot, shown in Figure 11.8, then simply plots the points and uses `lines()` to plot the model-implied variances.

```
> with(qdat, {
+   plot(var ~ mean, xlab="Mean number of articles", ylab="Variance",
+         pch=16, cex=1.2, cex.lab=1.2)
+   abline(h=mean(PhdPubs$articles), col=gray(.40), lty="dotted")
+   lines(mean, qvar, col="red", lwd=2)
+   lines(mean, nbvar, col="blue", lwd=2)
+   lines(lowess(mean, var), lwd=2, lty="dashed")
+   text(3, mean(PhdPubs$articles), "Poisson", col=gray(.40))
+   text(3, 5, "quasi-Poisson", col="red")
+   text(3, 6.7, "negbin", col="blue")
+   text(3, 8.5, "lowess")
+ })
```

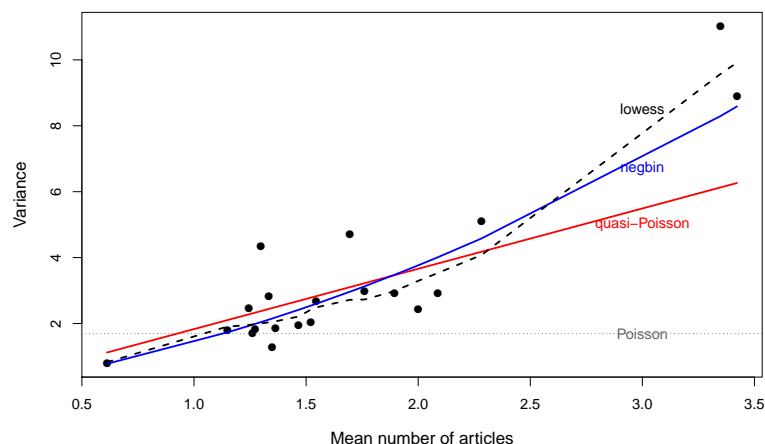


Figure 11.8: Mean–variance functions for the PhdPubs data. Points show the observed means and variances for 20 quantile groups based on the fitted values in the negative-binomial model. The labeled lines and curves show the variance functions implied by various models.

{fig:phd-mean-var-plot}

We can see from this plot that the variances implied by the quasi-Poisson and negative-binomial models are in reasonable accord with the data and with each other up to a mean of about 2.5. They diverge substantially at the upper end, for the 20–30% of the most productive candidates, where the quadratic variance function of the negative-binomial provides a better fit.

Finally, we can also compare the standard errors of coefficients for the various methods designed

to correct for overdispersion. These are extracted as the diagonal elements of the `vcov()` and `sandwich()` methods from the model objects.

```
> library(sandwich)
> phd.SE <- sqrt(cbind(
+   pois=diag(vcov(phd.pois)),
+   sand=diag(sandwich(phd.pois)),
+   qpois=diag(vcov(phd.qpois)),
+   nbin=diag(vcov(phd.nbin))))
> round(phd.SE, 4)
```

	pois	sand	qpois	nbin
(Intercept)	0.0996	0.1382	0.1348	0.1327
female1	0.0546	0.0714	0.0738	0.0726
married1	0.0613	0.0823	0.0829	0.0819
kid5	0.0401	0.0560	0.0543	0.0528
phdprestige	0.0253	0.0392	0.0342	0.0343
mentor	0.0020	0.0039	0.0027	0.0032

For this example, the sandwich, quasi-Poisson and negative-binomial methods give similar results, all about 40% larger on average than those from the Poisson model. \triangle

11.3.4 Testing overdispersion

The forms of overdispersion seen in these examples and in Figure 11.8 give rise to a statistical test (Cameron and Trivedi 1990; Cameron and Trivedi 1998, §3.4) for the null hypothesis of Poisson variation, $H_0 : \mathcal{V}(y) = \mu$ against an alternative that the variance has a particular form depending on the mean,

$$\mathcal{V}(y) = \mu + \alpha \times f(\mu) ,$$

where $f(\mu)$ is a given transformation function of the mean.

Overdispersion corresponds to $\alpha > 0$ and underdispersion to $\alpha < 0$. The coefficient α can be estimated by an auxiliary OLS regression (without an intercept, i.e., of the form

```
lm(var ~ -1 + f(mean))
```

and tested with the corresponding t (or z) statistic, which is asymptotically standard normal under the null hypothesis.

Common specifications of the transformation function are $f(\mu) = \mu$ and $f(\mu) = \mu^2$. The first corresponds to a NB model with a linear variance function (called NB1 by various authors) or a quasi-Poisson model with dispersion parameter ϕ , i.e.,

$$\mathcal{V}(y) = (1 + \alpha)\mu = \phi\mu .$$

The second is the more traditional form with quadratic variance function described in Section 11.3.2 (called NB2 by some authors).

These tests are carried out using the `dispersiontest()` function in the `AER` package. The first argument is a Poisson GLM model; the second specifies the alternative hypothesis, either as an integer power of μ or a function of the mean.

```
> library(AER)
> dispersiontest(phd.pois)
```

Overdispersion test

```
data:  phd.pois
z = 5.7347, p-value = 4.885e-09
```

```

alternative hypothesis: true dispersion is greater than 1
sample estimates:
dispersion
  1.8259

> dispersiontest(phd.pois, 2)

Overdispersion test

data:  phd.pois
z = 6.4579, p-value = 5.308e-11
alternative hypothesis: true alpha is greater than 0
sample estimates:
  alpha
0.50877

```

These tests use a specified alternative hypothesis, so there is no way to compare directly which of the NB1 or NB2 models is better or worse, except by using methods such as AIC or BIC described in Section 11.1.4.

11.3.5 Visualizing goodness-of-fit

{sec:glm-visfit}

Even with correction for overdispersion, goodness-of-fit tests provide only an overall summary of model fit. Some specialized tests for particular forms of overdispersion are also available (e.g., see Cameron and Trivedi (1998, Chapter 5)), but these only identify general problems and cannot provide detailed indications of the possible source of these problems.

In Chapter 3, we illustrated the use of rootograms for visualizing goodness-of-fit to a wide variety discrete distributions using the `plot()` method for class "goodfit" objects with the `vcd` package. However, those methods were developed for one-way discrete distributions without explanatory variables.

Kleiber and Zeileis (2014) have generalized this idea to the wider class of GLM-related count regression models considered here. The `countreg` package provides a new implementation of `rootogram()` with methods for all of these models (and others not mentioned). We illustrate these plots for the models considered to this point, and then extend this use for models allowing for excess zero counts in Section 11.4.

{ex:phdpubs4}

EXAMPLE 11.6: Publications of PhD candidates

For the *PhdPubs* data, Figure 11.9 shows hanging rootograms for the Poisson and negative-binomial models produced using `countreg::rootogram`⁹ on the fitted model objects. We are looking both for general patterns of under/over fit, as well as counts that stand out as poorly fitted against the background.

```

> library(countreg)
> countreg::rootogram(phd.pois, max=12, main="PhDPubs: Poisson")
> countreg::rootogram(phd.nbin, max=12, main="PhDPubs: Negative-Binomial")

```

The Poisson model shows a systematic, wave-like pattern with excess zeros, too few observed frequencies for counts of 1–3, but generally greater frequencies for counts of 4 or more. The negative-binomial model clearly fits much better, though there is a peculiar tendency among the smaller frequencies for 8 or more articles. \triangle

{ex:crabs2}

⁹At the time of this writing, `rootogram` in `countreg` conflicts with the version in `vcd`, so we qualify the use here with the package name.

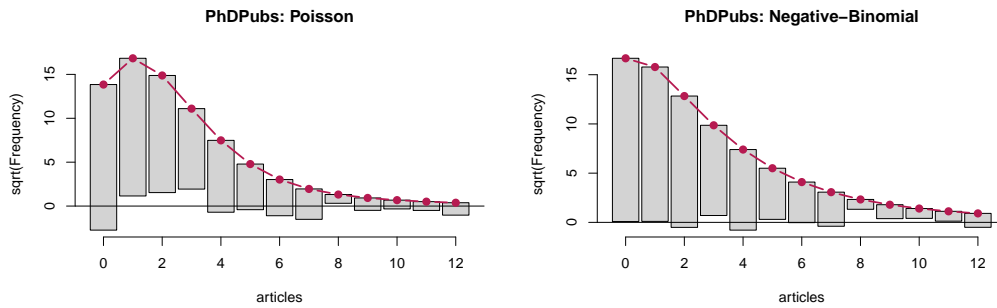


Figure 11.9: Hanging rootograms for the PhdPubs data.

{fig:phdpubs4-rootogram}

EXAMPLE 11.7: Mating of horseshoe crabs

Figure 11.10 shows similar plots for the same two models fit to the number of crab satellites. The fit of the Poisson model clearly reveals the excess of zero male satellites. For the negative-binomial, the rootogram no longer exhibits same wave-like pattern, however, the underfitting of the count for 0 and overfitting for counts 1–2 is characteristic of data with excess zeros.

```
> countreg::rootogram(crabs.pois, max=15, main="CrabSatellites: Poisson")
> countreg::rootogram(crabs.nbin, max=15, main="CrabSatellites: Negative-Binomial")
```

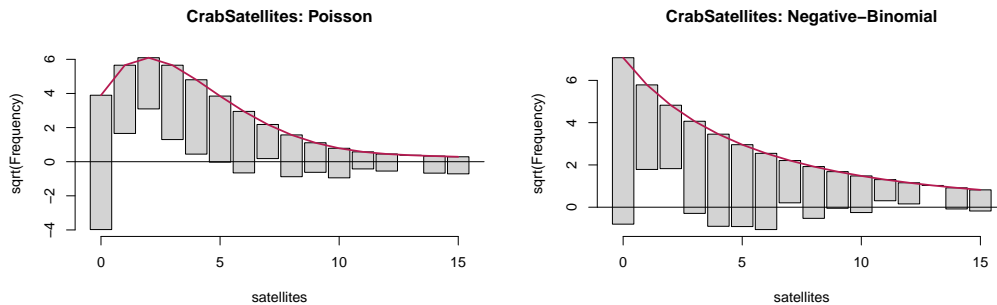


Figure 11.10: Hanging rootograms for the CrabSatellites data.

g:crabs2-rootogram}

△

11.4 Models for excess zero counts

{sec:glm-zeros}

In addition to overdispersion, many sets of empirical data exhibit a greater prevalence of zero counts than can be accommodated by the Poisson or negative-binomial models. We saw this in the *PhdPubs* data set, where there were many candidates who had not published at all, and in the *CrabSatellites* data where a large number of females attracted no unattached males. Other examples abound in many different fields: studies of the use of health care services often find that many people never visit a hospital in some time frame; similarly, the distribution of insurance claims often shows large numbers who make no claims (Yip and Yau, 2005) because of under-reporting of small claims, policy deductible provisions and desire to avoid premium increases.

Beyond simply identifying this as a problem of lack-of-fit, understanding the reasons for excess

zero counts can make a contribution to a more complete explanation of the phenomenon of interest, and this requires both new statistical models and visualization techniques illustrated in this section.

In the first example, Long (1997) argued that the PhD candidates might fall into two distinct groups: “publishers” (perhaps striving for an academic career) and “non-publishers” (seeking other career paths). Of the 275 observations having `articles==0`, some might not have published due to chance or unmeasured factors. One reasonable form of explanation is that the observed zero counts reflect a mixture of the two latent classes—those who simply have not yet published and those who will likely never publish. A statistical formulation of this idea leads to the class of *zero-inflated* models described below.

A different form of explanation is that there may be some special circumstance or “hurdle” required to achieve a positive count, like publishing the master’s thesis (such as being driven internally by a personality trait or externally by pressure from a mentor). This idea leads to the class of *hurdle* models that entertain and fit (simultaneously) two separate models: one for the occurrence of the zero counts, and one for the positive counts. These two approaches are illustrated in Figure 11.11

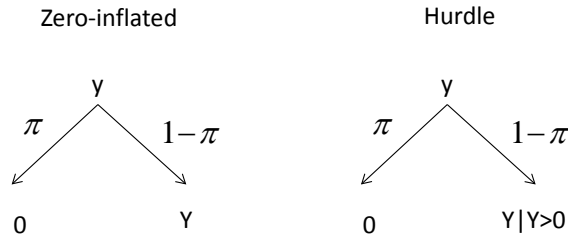


Figure 11.11: Models for excess zeros. The observed response y is derived from a latent or parent distribution for Y yielding zero counts with probability π .

{fig:ExcessZeros}

11.4.1 Zero-inflated models

{sec:glm-zip}

Zero-inflated models, introduced by Lambert (1992) as the **zero-inflated Poisson (ZIP)** model, provide an attractive solution to the problem of dealing with an overabundance of zero counts. It postulates that the observed counts arise from a mixture of two latent classes of observations: some structural zeros for whom y_i will always be 0, and the rest, sometimes giving random zeros. The ZIP model is comprised of two components:

- A model for the binary event of membership in the unobserved (latent) class of those for whom the count is necessarily zero (e.g., “non-publishers”). This is typically taken as a logistic regression for the probability π_i that observation i is in this class, with predictors z_1, z_2, \dots, z_q , giving

$$\text{logit}(\pi_i) = \mathbf{z}_i^T \boldsymbol{\gamma} = \gamma_0 + \gamma_1 z_{i1} + \gamma_2 z_{i2} + \dots + \gamma_q z_{iq} . \quad (11.7) \quad \{\text{eq:zip-logit}\}$$

- A Poisson model for the other class (e.g., “publishers”), for whom the observed count may 0 or positive. This model typically uses the usual log link to predict the mean, using predictors $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_p$, so

$$\log_e \mu(\mathbf{x}_i) = \mathbf{x}_i^T \boldsymbol{\beta} = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \dots + \beta_p x_{ip} . \quad (11.8) \quad \{\text{eq:zip-pois}\}$$

In application, it is permissible and not uncommon to use the same set of predictors $\mathbf{x} = \mathbf{z}$ in both submodels, but the notation indicates that this is not required. Some simple special cases arise when the model for the always zero latent class is an intercept-only model, $\text{logit}(\pi_i) = \gamma_0$, implying

the same probability for all individuals, and (less commonly) when the Poisson mean model is intercept-only with no predictors but there might be excess zero counts.

With this setup, one can show that the probability of observing counts of $y_i = 0$ and $y_i > 0$ are

$$\begin{aligned}\Pr(y_i = 0 | \mathbf{x}, \mathbf{z}) &= \pi_i + (1 - \pi_i)e^{-\mu_i} \\ \Pr(y_i | \mathbf{x}, \mathbf{z}) &= (1 - \pi_i) \times \left[\frac{\mu_i^{y_i} e^{-\mu_i}}{y_i!} \right], \quad y_i \geq 0\end{aligned}\tag{11.9} \quad \{\text{eq:zip-probs}\}$$

where the term in brackets in the second equation is the Poisson probability $\Pr(y = y_i)$ with rate parameter $\text{Pois}(\mu_i)$. In these equations, $\pi_i = \text{logit}^{-1}(\mathbf{z}_i^T \boldsymbol{\gamma})$ depends on the \mathbf{z} through Eqn. (11.7), and $\mu_i = \exp(\mathbf{x}_i^T \boldsymbol{\beta})$ depends on the \mathbf{x} through Eqn. (11.8).

The conditional expectation and variance of y_i then have the forms

$$\begin{aligned}\mathcal{E}(y_i) &= (1 - \pi_i) \mu_i \\ \mathcal{V}(y_i) &= (1 - \pi_i) \mu_i (1 + \mu_i \pi_i) .\end{aligned}$$

Thus, when $\pi_i > 0$, the mean of y is always less than μ_i , and the variance of y is greater than its mean by a dispersion factor of $(1 + \mu_i \pi_i)$.

There is nothing special about the use of the Poisson distribution here. The model for the count variable could also be taken as the negative-binomial, giving a *zero-inflated negative-binomial* (ZINB) model using $\text{NBin}(\mu, \theta)$ or a *zero-inflated geometric* model using $\text{NBin}(\mu, \theta = 1)$.

`{ex:zipois}`

EXAMPLE 11.8: Simulating zero-inflated data

A simple way of understanding the effects of zero-inflation on count data is to simulate data from their distribution and plot it. For the standard Poisson and negative-binomial, random values can be generated using `rpois()` and `rnegbin()` (in **MASS**), respectively. Their zero-inflated counterparts are implemented in the **VGAM** package as `rzipois()` and `rzinegbin()`.

To illustrate this use, we generate two random data sets using `rzipois()` having constant mean $\mu = 3$. The first is a standard Poisson ($\pi = 0$), while the second has a constant probability $\pi = 0.3$ of an excess zero.

```
> library(VGAM)
> set.seed(1234)
> data1 <- rzipois(200, 3, 0)
> data2 <- rzipois(200, 3, .3)
```

Barplots of the frequencies in these data sets are shown in Figure 11.12. The sample mean in `data1` is 2.925, quite close to $\mu = 3$. In the zero-inflated `data2`, the mean is only 2.25 due to the excess zeros.

```
> tdata1 <- table(data1)
> barplot(tdata1, xlab="Count", ylab="Frequency",
+         main="Poisson(3) ")
> tdata2 <- table(data2)
> barplot(tdata2, xlab="Count", ylab="Frequency",
+         main=expression("ZI Poisson(3, " * pi * "= .3)"))
```

△

There are several packages in **R** capable of fitting zero-inflated models. The most mature and complete of these is `zeroinfl()` in the **countreg** package (a successor to the **pscl** package) The function `zeroinfl()` is modeled after `glm()`, but provides an extended syntax for the model formula.

If the `formula` argument is supplied in the form $y \sim x_1 + x_2 + \dots$, it not only describes the count regression of y on x_1, x_2, \dots , but also implies that the *same* set of regressors,

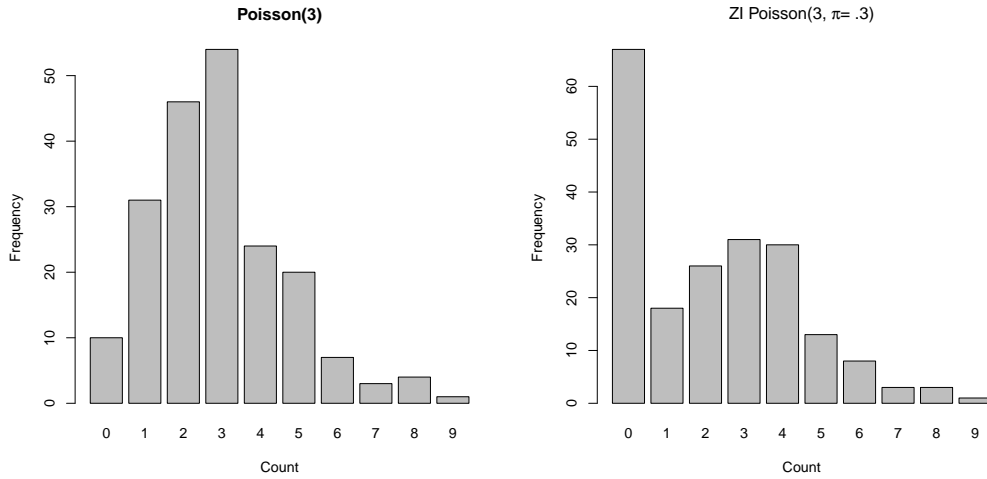


Figure 11.12: Bar plots of simulated data from Poisson and zero-inflated Poisson distributions

$z_j = x_j$, is used for the zero count binary submodel. The extended syntax uses the notation $y \sim x_1 + x_2 + \dots \mid z_1 + z_2 + \dots$ to specify the x variables separately, conditional on (\mid) the always-zero count model $y \sim z_1 + z_2 + \dots$. 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

{sec:glm-hurdle}

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:

$$\begin{aligned} \Pr(y_i = 0 \mid \mathbf{x}, \mathbf{z}) &= \pi_i \\ \Pr(y_i \mid \mathbf{x}, \mathbf{z}) &= \frac{(1 - \pi_i)}{1 - e^{-\mu_i}} \times \left[\frac{\mu_i^{y_i} e^{-\mu_i}}{y_i!} \right], \quad y_i \geq 0 \end{aligned} \quad (11.10)$$

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 $y \sim x_1 + x_2$ uses the same regressors for the zero and positive count submodels, while $y \sim x_1 + x_2 \mid z_1 + z_2$ uses $y \sim z_1 + z_2$ 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 sub-models, 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 plot or spinogram, the widths of the bars correspond to the relative frequencies of x and heights of the bars correspond to the conditional relative frequencies of $y = 0$ in every x group. In R, spine plots are implemented in the function `spineplot()`, however, this is what you get by default if you use `plot(y==0 ~ x)` to plot the binary factor against any regressor x .

A related graphical method is the *conditional density plot* (Hofmann and Theus, 2005). The conditional probabilities $\Pr(y = 0|x)$ are derived using a smoothing approach (via `density()`) over x rather than by making x discrete. These plots are provided by `cdplot()` in the `graphics` package and a similar `cd_plot()` in `vcd`. The smoothing method for the density estimate is controlled by a `bw` (bandwidth) method and other arguments.

EXAMPLE 11.9: Mating of horseshoe crabs

For the *CrabSatellites* data, we can examine the relationship of the zero counts (females who attract no unattached male satellites) to the predictors using spinograms or conditional density plots. Here, we consider `weight` and `color` (treated numerically) as predictors. **TODO: Fixup use of color in Example 11.2 so it doesn't cause a problem here**

Spinograms for the occurrence of zero satellites against `weight` and `color` are shown in Figure 11.13, where we have used quantiles of those distributions to define the breaks on the horizontal axis. Using `ylevels=2:1` reverses the order of the vertical categories. You can easily see that the zeros decrease steadily with weight and increase with darkness.

```
> op <- par(cex.lab=1.2, mfrow = c(1, 2))
> plot(factor(satellites == 0) ~ weight, data = CrabSatellites,
+       breaks = quantile(weight, probs=seq(0,1,.2)), ylevels=2:1,
+       ylab="No satellites")
> plot(factor(satellites == 0) ~ color, data = CrabSatellites,
+       breaks = quantile(color, probs=seq(0,1,.33)), ylevels=2:1,
+       ylab="No satellites")
> par(op)
```

Similar plots in the form of conditional density plots are shown in Figure 11.14, with a similar interpretation.

```
> op <- par(cex.lab=1.2, mfrow = c(1, 2))
> cdplot(factor(satellites == 0) ~ weight, data = CrabSatellites,
+        ylevels=2:1, ylab="No satellites")
> cdplot(factor(satellites == 0) ~ color, data = CrabSatellites,
+        ylevels=2:1, , ylab="No satellites")
> par(op)
```

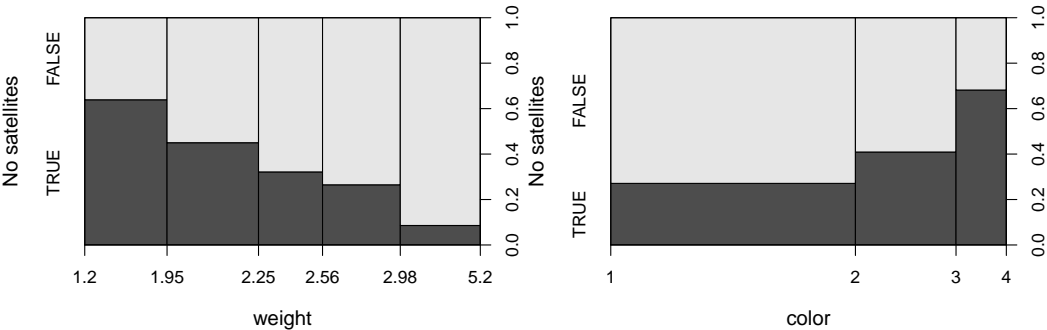


Figure 11.13: Spinograms for the CrabSatellites data. The variables weight (left) and color(right) have been made discrete using quantiles of their distributions.

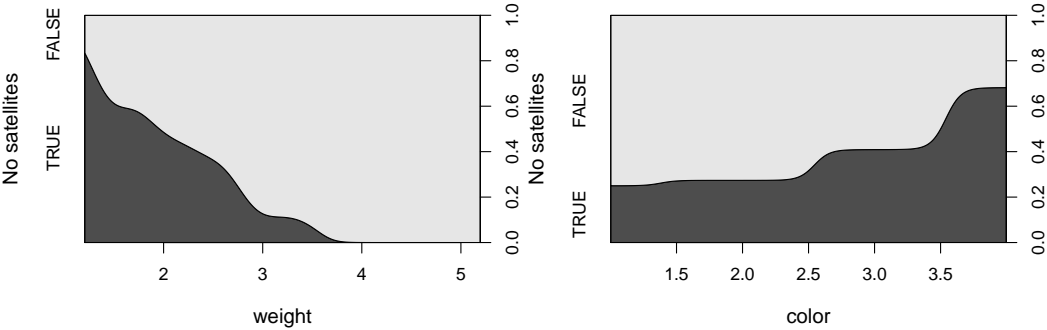


Figure 11.14: Conditional density plots for the CrabSatellites data. The region shaded below shows the conditional probability density estimate for a count of zero.

11.5 Case studies

{ec:glm-casestudies}

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

{sec:glm-case-cod}

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.”¹⁰

{ex:cod1}

EXAMPLE 11.10: Cod parasites

The data from Hemmingsen *et al.* (2005) is contained in *CodParasites* in the *countreg* package. It gives the results for 1254 cod caught by one ship in annual autumn cruises from 1999–2001. The main response variable, *intensity*, records the counted number of *Trypanosoma* parasites found in blood samples from these fish. To distinguish between infected vs. non-infected fish, a secondary response, *prevalence* is also recorded, corresponding to the expression

```
> CodParasites$prevalence <- ifelse(CodParasites$intensity == 0, "no", "yes")
```

Thus, *intensity* is the basic count response variable, and *prevalence* reflects the zero count that would be assessed in zero-inflated and hurdle models. In substantive terms, in a hurdle model, *prevalence* corresponds to whether a fish is infected or not; once infected, *intensity* gives the degree of infection. In a zero-inflated model, infected could be considered a latent variable; there are extra zeros from non-infected fish, but some infected fish are measured as “normal” zeros.

Hemmingsen *et al.* (2005) consider only three explanatory predictors: *area*, *year* (both factors) and *length* of the fish.¹¹ A quick numerical summary of the univariate properties of these variables is shown below. The intensity values are indeed extremely skewed, with a median of 0 and

¹⁰ 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.

¹¹ Other potential predictors include weight, sex, age, and developmental stage, as well as the depth at which the fish were caught.

{fn:russian}

a maximum of 257. However, there are some missing values (NAs) among the response variables and a few in the length variable.

```
> data("CodParasites", package = "countreg")
> summary(CodParasites[, c(1:4,7)])
```

intensity	prevalence	area	year	length
Min. : 0.00	no : 654	soroya : 272	1999: 567	Min. : 17.0
1st Qu.: 0.00	yes : 543	mageroya : 255	2000: 230	1st Qu.: 44.0
Median : 0.00	NA's : 57	tanafjord : 415	2001: 457	Median : 54.0
Mean : 6.18		varangerfjord : 312		Mean : 53.4
3rd Qu.: 4.00				3rd Qu.: 62.0
Max. : 257.00				Max. : 101.0
NA's : 57				NA's : 6

Even better, a quick univariate and bivariate summary of these variables can be shown in a generalized pairs plot (Figure 11.15).

```
> library(vcd)
> library(gpairs)
> gpairs(CodParasites[, c(1:4,7)],
+        diag.pars=list(fontsize=16),
+        mosaic.pars=list(gp=shading_Friendly))
```

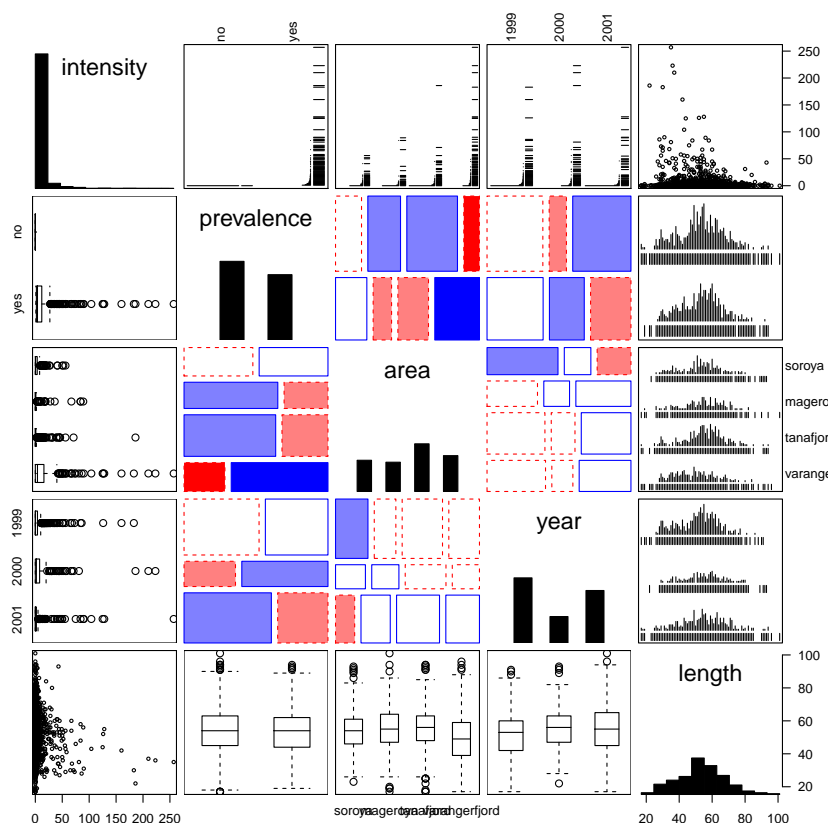


Figure 11.15: Generalized pairs plot for the CodParasites data.

{fig:cod1-gpairs}

In this plot, among the categorical variables, prevalence is strongly associated with area, but also with year. As well there seems to be an association between area and year, meaning the number

of cod samples collected in different areas varied over time. In the univariate plots on the diagonal, intensity stands out as extremely skewed, and the distribution of length appears reasonably symmetric.

Before fitting any models, some more detailed exploratory plots are helpful for understanding the relationship of both prevalence and intensity to the predictors. The general idea is to make separate plots of prevalence and intensity and to try to show both the data and some simple summaries. In their Table 1, Hemmingsen *et al.* (2005) counted the missing observations as infected and we do the same to get a similar contingency table.

```
> cp.tab <- xtabs(~ area + year + factor(is.na(prevalence) |
+                                     prevalence == "yes"),
+               data = CodParasites)
> dimnames(cp.tab)[3] <- list(c("No", "Yes"))
> names(dimnames(cp.tab))[3] <- "prevalence"
```

For the factors *area* and *year*, we can visualize prevalence as before (Example 11.9) using spineplots, but, for two (or more) factors, doubledecker and mosaic plots are better because they are more flexible and keep the factors distinct. The doubledecker plot (Figure 11.16) highlights the infected fish, and shows that prevalence is indeed highest in all years in Varangerfjord.

```
> doubledecker(prevalence ~ area + year, data=cp.tab)
```

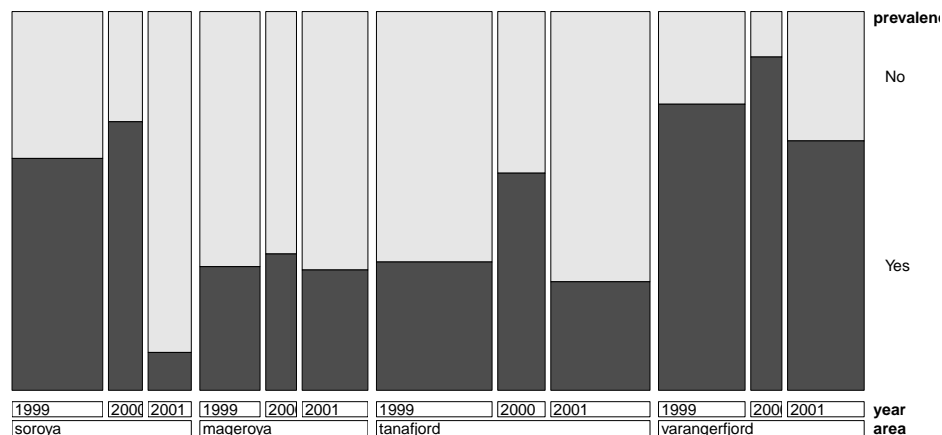


Figure 11.16: Doubledecker plot for prevalence against area and year in the CodParasites data. The cases of infected fish are highlighted

A similar plot, in the doubledecker format, can be drawn as a mosaic plot, but now shading the tiles according to a model for the expected counts. It makes sense here to consider the null loglinear model for prevalence as a response, independent of the combinations of area and year. This plot (Figure 11.17) shows further that prevalence differs substantially over the area-year combinations, so we should expect an interaction in the model for zero counts. As well, Varangerfjord stands out as having consistently greater prevalence in all years than expected under this model.

```
> mosaic(~area + year + prevalence, data=cp.tab,
+        split_vertical=c(TRUE, TRUE, FALSE),
+        labeling=labeling_doubleddecker, spacing=spacing_highlighting,
+        expected = ~year:area + prevalence)
```

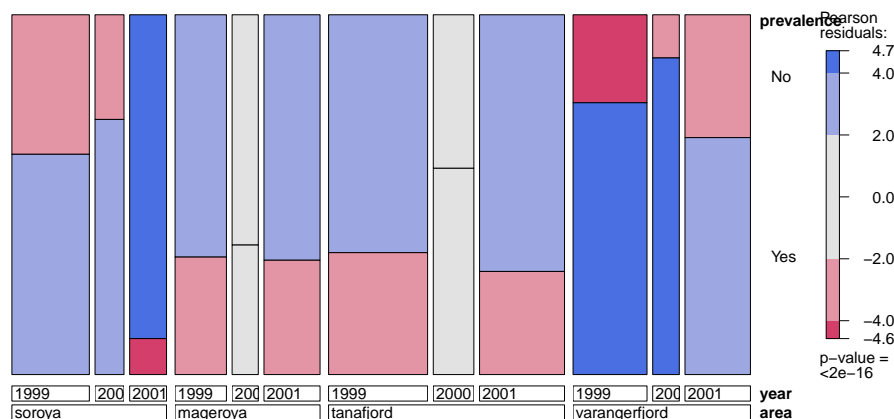


Figure 11.17: Mosaic plot for prevalence against area and year in the CodParasites data, in the doubledecker format. Shading reflects departure from a model in which prevalence is independent of area and year jointly.

{fig:cod1-mosaic}

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) +
+   theme_bw() + labs(y='prevalence')
```

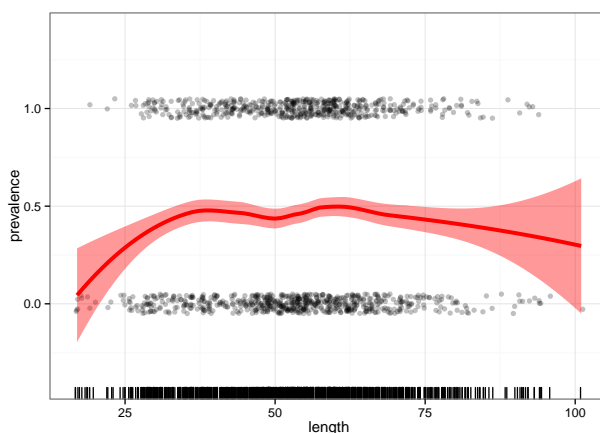


Figure 11.18: Jittered scatterplot of prevalence against length of fish, with loess smooth.

{fig:cod1-length-prevalence}

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)) +
+   geom_boxplot(outlier.size=3, notch=TRUE, aes(fill=year), alpha=0.2) +
+   geom_jitter(position=position_jitter(width=0.1), alpha=0.25) +
+   facet_grid(.~area) +
+   scale_y_log10(breaks=c(1,2,5,10,20,50,100, 200)) +
+   theme_bw() + theme(legend.position="none") +
+   labs(y='intensity (log scale)')
```

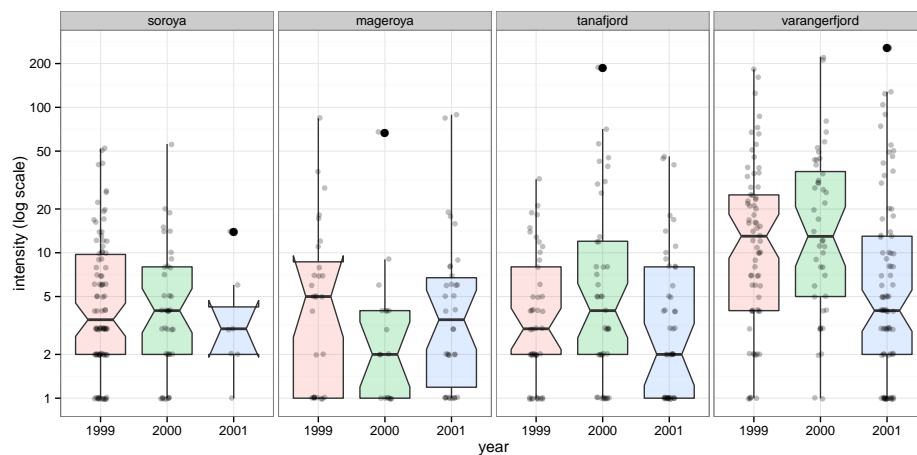


Figure 11.19: Notched boxplots for log (intensity) of parasites by area and year in the CodParasites data. Significant differences in the medians are signaled when the notches of two groups do not overlap.

{fig:cod1-boxplot}

Most of these distributions are positively skewed and there are a few high outliers, but probably not more than would be expected in a sample of this size. The positive counts (degree of infection) are also higher in all years in Varangerfjord than other areas. You can also see that the intensity values were generally lower in 2001 than other years.

For the effect of length of fish, we want to know if log (intensity) is reasonably linear on length. A jittered scatterplot produced with `ggplot2` is shown in Figure 11.20. The smoothed loess curve together with the linear regression line show no indication of non-linearity.

```
> ggplot(CPpos, aes(x=length, y=intensity)) +
+   geom_jitter(position=position_jitter(height=.1), alpha=0.25) +
+   geom_rug(position='jitter', sides='b') +
+   scale_y_log10(breaks=c(1,2,5,10,20,50,100, 200)) +
+   stat_smooth(method="loess", color="red", fill="red", size=2) +
+   stat_smooth(method="lm", size=1.5) + theme_bw()
```

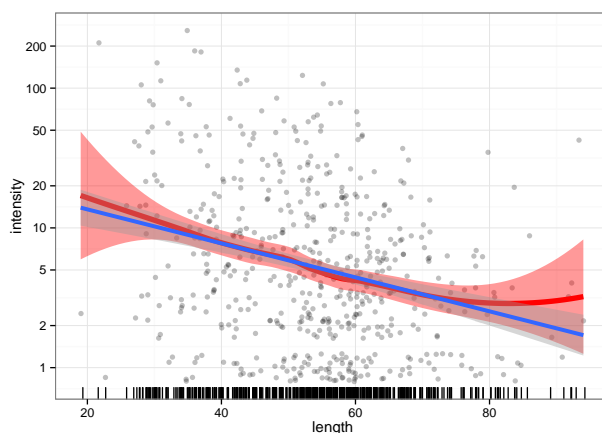


Figure 11.20: Jittered scatterplot of log (intensity) for the positive counts against length of fish, with loess smooth and linear regression line.

```
g:cod1-length-scat}
```

11.5.1.1 Fitting models

The simple summary of these exploratory analyses is that both the zero component (prevalence) and non-zero component (intensity) involve an interaction of area and year and at least intensity depends on length. We proceed to fit some count data models.

```
{ex:cod2}
```

EXAMPLE 11.11: Cod parasites

For a baseline reference, we first fit the standard Poisson and negative-binomial models, not allowing for excess zeros.

```
> library(MASS); library(countreg)
> cp_p <- glm(intensity ~ length + area * year,
+             data = CodParasites, family = poisson)
> cp_nb <- glm.nb(intensity ~ length + area * year,
+                 data = CodParasites)
```

Next, we fit analogous hurdle and zero-inflated models, in each case allowing the non-zero count component to be either Poisson or negative-binomial. The zero components are fit as logistic regressions with the same predictors and the logit link.

```
> cp_hp <- hurdle(intensity ~ length + area * year,
+                 data = CodParasites, dist = "poisson")
> cp_hnb <- hurdle(intensity ~ length + area * year,
+                  data = CodParasites, dist = "negbin")
> cp_zip <- zeroinfl(intensity ~ length + area * year,
+                    data = CodParasites, dist = "poisson")
> cp_znb <- zeroinfl(intensity ~ length + area * year,
+                     data = CodParasites, dist = "negbin")
```

Following Section 11.3.5, we can compare the fit of these models using rootograms. The details of fit of these six models are shown in Figure 11.21.

```
> op <- par(mfrow = c(3, 2))
> countreg::rootogram(cp_p, max = 50, main = "Poisson")
> countreg::rootogram(cp_nb, max = 50, main = "Negative Binomial")
> countreg::rootogram(cp_hp, max = 50, main = "Hurdle Poisson")
> countreg::rootogram(cp_hnb, max = 50, main = "Hurdle Negative Binomial")
> countreg::rootogram(cp_zip, max = 50, main = "Zero-inflated Poisson")
```

```
> countreg::rootogram(cp_znb, max = 50, main = "Zero-inflated Negative Binomial")
> par(op)
```

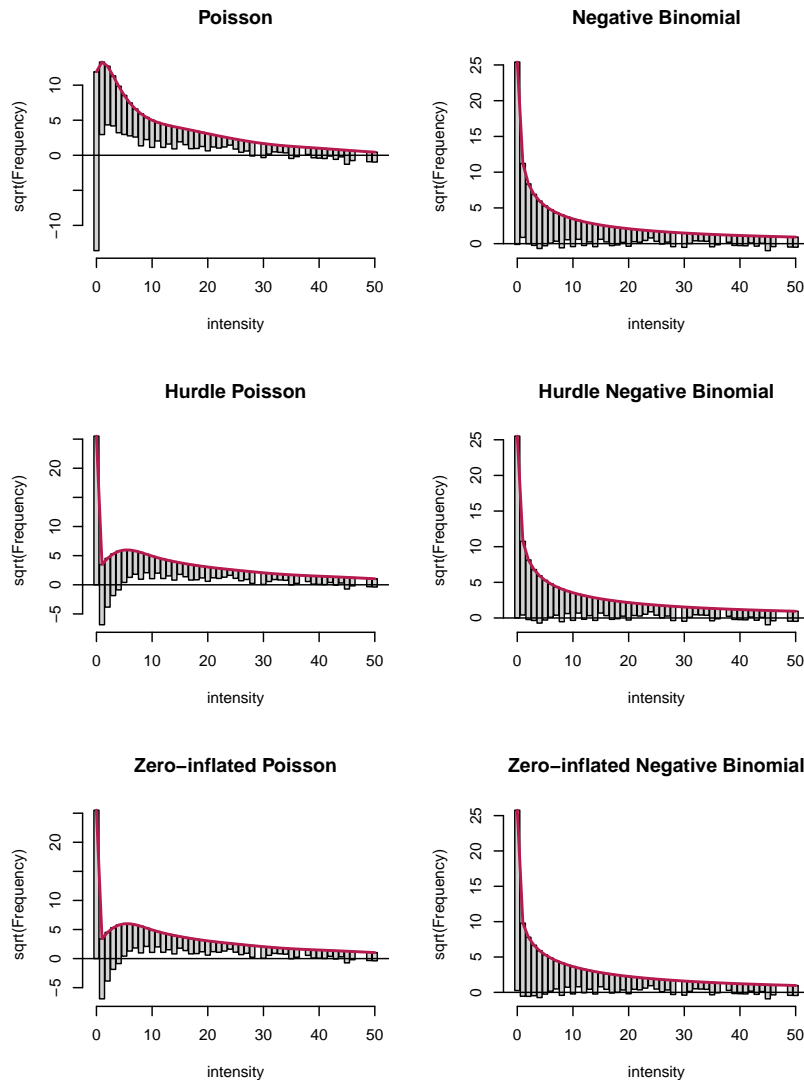


Figure 11.21: Rootograms for six models fit to the CodParasites data

The basic Poisson model of course fits terribly due to the excess zero counts. The hurdle Poisson and zero-inflated Poisson fit the zero counts perfectly, but at the expense of underfitting the counts for low intensity values. All of the negative binomial models show a reasonable fit (at the scale shown in this plot), and none show a systematic pattern of under/overfitting.

These models are all in different GLM and extended-GLM families, and there are no `anova()` methods for hurdle and zero-inflated models. Each pair of Poisson and negative-binomial models are a nested set, because the Poisson is a special case of the negative-binomial where $\theta \rightarrow \infty$, and so can be compared using likelihood-ratio tests available with `lrtest()` from `lmtest`. However, this cannot be used to compare models of different class, such as a hurdle model vs. a zero-inflated

model. (In Figure 11.21, each pair in the same row are nested models, while all other pairs are non-nested.) Yet, they all have `logLik()` methods to calculate their log likelihood, and so `AIC()` and `BIC()` can be used.

```
> vcdExtra::LRstats(cp_p, cp_nb, cp_hp, cp_hnb, cp_zip, cp_znb, sortBy="BIC")

Likelihood summary table:
      AIC    BIC LR Chisq  Df Pr(>Chisq)
cp_p   20378 20444      20352 1178    <2e-16 ***
cp_hp   13688 13820      13636 1165    <2e-16 ***
cp_zip  13687 13819      13635 1165    <2e-16 ***
cp_nb    5031  5102      5003 1178    <2e-16 ***
cp_znb   4955  5092      4901 1164    <2e-16 ***
cp_hnb   4937  5074      4883 1164    <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

These show that all the Poisson models fit quite badly, and among the negative-binomial models, the hurdle version, `cp_hnb`, is preferred by both AIC and BIC. If you want to carry out formal tests, `lrtest()` can be used to compare a given Poisson model to its negative-binomial counterpart, which are nested. For example, the test below compares the hurdle Poisson to the hurdle negative-binomial and confirms that the latter is a significant improvement.

```
> library(lmtest)
> lrtest(cp_hp, cp_hnb)

Likelihood ratio test

Model 1: intensity ~ length + area * year
Model 2: intensity ~ length + area * year
#Df LogLik Df Chisq Pr(>Chisq)
1  26 -6818
2  27 -2442  1  8752    <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Of greater interest is the difference among the negative-binomial models, that are not nested. As described in Section 11.1.4, these can be compared using Vuong's test.

```
> library(pscl)
> vuong(cp_nb, cp_hnb)      # nb vs. hurdle nb

[1] -46.037
      Raw AIC-corrected BIC-corrected
      -5.4873      -5.4863      -1.2625
[1] 0.31575
[1] -59.794 -59.783 -13.758
Vuong Non-Nested Hypothesis Test-Statistic:
(test-statistic is asymptotically distributed N(0,1) under the
null that the models are indistinguishable)
-----
      Vuong z-statistic      H_A p-value
Raw      -5.4873 model2 > model1 2.04e-08
AIC-corrected -5.4863 model2 > model1 2.05e-08
BIC-corrected -1.2625 model2 > model1 0.103

> vuong(cp_hnb, cp_znb)      # hurdle nb vs znb

[1] 0
      Raw AIC-corrected BIC-corrected
      1.7941      1.7941      1.7941
```

```
[1] 0.14383
[1] 8.9056 8.9056 8.9056
Vuong Non-Nested Hypothesis Test-Statistic:
(test-statistic is asymptotically distributed N(0,1) under the
null that the models are indistinguishable)
-----
              Vuong z-statistic              H_A p-value
Raw              1.7941 model1 > model2  0.0364
AIC-corrected    1.7941 model1 > model2  0.0364
BIC-corrected    1.7941 model1 > model2  0.0364
```

The negative-binomial model is considered to be a closer fit than the hurdle version (because it is more parsimonious), while the hurdle NB model has a significantly better fit than the zero-inflated NB model. For this example, we continue to work with the hurdle NB model. The tests for individual coefficients in this model are shown below.

```
> summary(cp_hnb)

Call:
hurdle(formula = intensity ~ length + area * year, data = CodParasites,
  dist = "negbin")

Pearson residuals:
      Min       1Q   Median       3Q      Max
-0.696 -0.407 -0.336 -0.108  11.114

Count model coefficients (truncated negbin with log link):
              Estimate Std. Error z value Pr(>|z|)
(Intercept)    3.37580    0.39947   8.45 < 2e-16 ***
length        -0.03748    0.00587  -6.38 1.7e-10 ***
areamageroya     0.37898    0.38105   0.99  0.3199
areatanafjord  -0.50480    0.31238  -1.62  0.1061
areavarangerfjord 0.89159    0.29161   3.06  0.0022 **
year2000       -0.03957    0.32857  -0.12  0.9041
year2001       -0.75388    0.68925  -1.09  0.2741
areamageroya:year2000 -0.63981    0.61667  -1.04  0.2995
areatanafjord:year2000 1.19387    0.49479   2.41  0.0158 *
areavarangerfjord:year2000 0.51074    0.47719   1.07  0.2845
areamageroya:year2001 0.70444    0.82036   0.86  0.3905
areatanafjord:year2001 0.90824    0.77685   1.17  0.2424
areavarangerfjord:year2001 0.59838    0.74738   0.80  0.4233
Log(theta)    -1.49866    0.23904  -6.27 3.6e-10 ***
Zero hurdle model coefficients (binomial with logit link):
              Estimate Std. Error z value Pr(>|z|)
(Intercept)    0.08526    0.29505   0.29  0.773
length         0.00693    0.00465   1.49  0.136
areamageroya   -1.32137    0.28526  -4.63 3.6e-06 ***
areatanafjord  -1.44918    0.24388  -5.94 2.8e-09 ***
areavarangerfjord 0.30073    0.27111   1.11  0.267
year2000       0.39507    0.34382   1.15  0.251
year2001      -2.65201    0.43340  -6.12 9.4e-10 ***
areamageroya:year2000 -0.08034    0.50797  -0.16  0.874
areatanafjord:year2000 0.87058    0.45027   1.93  0.053 .
areavarangerfjord:year2000 0.86462    0.59239   1.46  0.144
areamageroya:year2001 2.73749    0.53291   5.14 2.8e-07 ***
areatanafjord:year2001 2.71899    0.49949   5.44 5.2e-08 ***
areavarangerfjord:year2001 2.54144    0.51825   4.90 9.4e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Theta: count = 0.223
Number of iterations in BFGS optimization: 25
Log-likelihood: -2.44e+03 on 27 Df
```

From the above and from Figure 11.18, it appears that length is not important as a linear effect in the submodel for prevalence. A revised model excludes this from the zero formula.

```
> cp_hnb1 <- hurdle(intensity ~ length + area * year | area*year,
+                   data = CodParasites, dist = "negbin")
```

A likelihood-ratio test shows no advantage for the smaller model, however Vuong's test leads to the conclusion that this reduced model is preferable:

```
> lrtest(cp_hnb, cp_hnb1)

Likelihood ratio test

Model 1: intensity ~ length + area * year
Model 2: intensity ~ length + area * year | area * year
#Df LogLik Df Chisq Pr(>Chisq)
1 27 -2442
2 26 -2443 -1 2.23 0.14

> vuong(cp_hnb, cp_hnb1)

[1] 3.5413
      Raw AIC-corrected BIC-corrected
      0.74180      0.74124      -1.61277
[1] 0.04358
[1] 1.1157 1.1148 -2.4256
Vuong Non-Nested Hypothesis Test-Statistic:
(test-statistic is asymptotically distributed N(0,1) under the
null that the models are indistinguishable)
-----
              Vuong z-statistic              H_A p-value
Raw              0.74180 model1 > model2 0.2291
AIC-corrected    0.74124 model1 > model2 0.2293
BIC-corrected   -1.61277 model2 > model1 0.0534
```

△

11.5.1.2 Model interpretation: Effect plots

Interpreting these models from their coefficients is very difficult because an interaction is present and there are separate submodels for the zero and count components. This task is much easier with effects plots. The `effects` package has methods for any GLM, but cannot handle the extended forms of the zero-inflated and hurdle models.

When the same predictors are used in both submodels, and a standard GLM such as the negative-binomial provides a reasonable fit, you can use the standard `effects` functions to visualize the (total) expected count, which for the zeros would include both the extra zeros and those that derive from the count submodel. For visual interpretation, these will be sufficiently similar, even though the hurdle and zero-inflated models differ with respect to explaining overdispersion and/or excess zeros.

Alternatively, if you want to visualize and interpret the zero and nonzero components separately, perhaps with different predictors, you can fit the implied submodels separately, and then use `effects` functions for the effects in each. These ideas are illustrated in the next example.

{ex:cod3}

EXAMPLE 11.12: Cod parasites

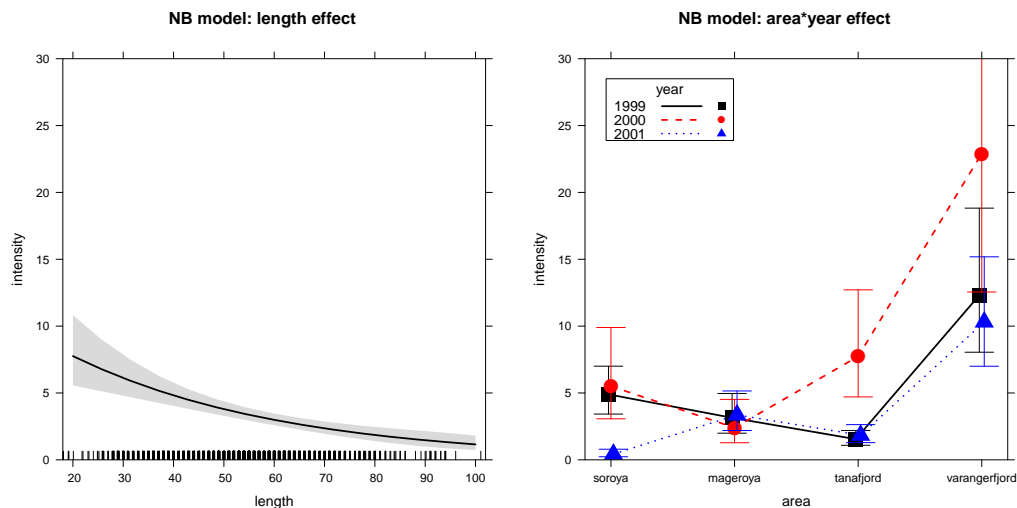
The expected counts for `intensity`, including both zero and positive counts can be plotted using `effects` for the `cp_nb` NB model. Figure 11.21 gives some confidence that the fitted values are similar to those in the hurdle and zero-inflated versions.

We use `allEffects()` to calculate the effects for the high-order terms—the main effect of `length` and the interaction of `area` and `year`. These could be plotted together by plotting the resulting `eff.nb` object, but we plot them separately to control the plot details. In these plots, the argument `rescale=FALSE` gives plots on the response scale, and we use `ylim` to equate the ranges to make the plots directly comparable. The code below produces Figure 11.22.

```

> library(effects)
> eff.nb <- allEffects(cp_nb)
> plot(eff.nb[1], rescale=FALSE, ylim=c(0,30),
+      main="NB model: length effect")
>
> plot(eff.nb[2], rescale=FALSE, ylim=c(0,30),
+      multiline=TRUE, ci.style='bars',
+      key.args=list(x=.05, y=.95, columns=1),
+      colors=c("black", "red", "blue"),
+      symbols=15:17, cex=2,
+      main="NB model: area*year effect")

```



{fig:cod3-eff1}

Figure 11.22: Effect plots for total intensity of parasites from the negative-binomial model

This helps to interpret the nature of the area by year effect. The pattern of mean expected intensity of cod parasites is similar in 1999 and 2001, except for the Sørøya area. The results in year 2000 differ mainly in greater intensity in Tanafjord and Varangerfjord. Varangerfjord shows larger infection counts overall, but particularly in year 2000. The effect plot for length on this scale is roughly comparable to the variation in areas and years.

In this example, the submodels for zero and positive counts have substantively different interpretations. To visualize the fitted effects in these submodels using `effects`, first fit the equivalent submodels separately using GLM methods. The following models for prevalence, using the binomial family, and the positive counts for intensity, using `glm.nb()`, give similar fitted results to those obtained from the hurdle negative-binomial model, `cp_hnb` discussed earlier.

```

> cp_zero <- glm(prevalence ~ length + area * year,
+               data = CodParasites, family=binomial)
> cp_nzero <- glm.nb(intensity ~ length + area * year,
+                   data = CodParasites, subset=intensity>0)

```

We could construct effect plots for each of these submodels, but interest here is largely on the binomial model for the zero counts, `cp_zero`. Effect plots for the terms in this model are shown in Figure 11.23. Again, we set the `ylim` values to equate the vertical ranges to make the plots comparable.

```

> eff.zero <- allEffects(cp_zero)
> plot(eff.zero[1], ylim=c(-2.5, 2.5),
+      main="Hurdle zero model: length effect")
>
> plot(eff.zero[2], ylim=c(-2.5, 2.5),
+      multiline=TRUE,
+      key.args=list(x=.05, y=.95, columns=1),
+      colors=c("black", "red", "blue"),
+      symbols=15:17, cex=2,
+      main="Hurdle zero model: area*year effect")

```

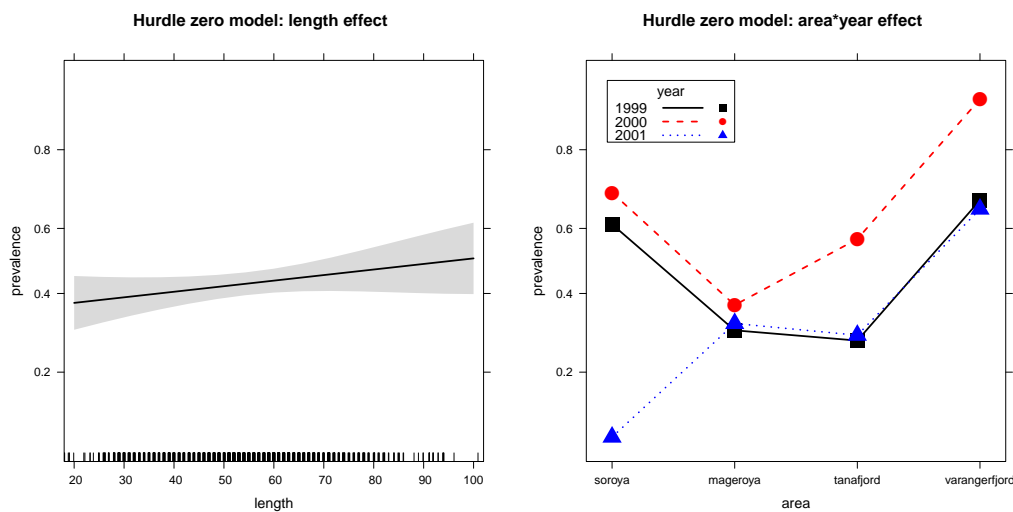


Figure 11.23: Effect plots for prevalence of parasites analogous to the hurdle negative-binomial model, fitted using a binomial GLM model.

{fig:cod3-eff2}

The effect of `length` on prevalence is slightly increasing, but we saw earlier that this is not significant. For the area-year interaction, the three curves have similar shapes, except for the aberrant value for Sørøya in 2001 and the closeness of the values at Magerøya in all years. Overall, prevalence was highest in 2000, and also in the Varangerfjord samples.

△

11.5.2 Demand for medical care by the elderly

{sec:glm-case-nmes}

A large cross-sectional study was carried out by the U.S. National Medical Expenditure Survey (NMES) in 1987–1988 to assess the demand for medical care, as measured by the number of physician/non-physician office visits and the number of hospital outpatient visits to a physician/non-physician. The survey was based upon a representative, national probability sample of the civilian non-institutionalized population and individuals admitted to long-term care facilities during 1987. A subsample of 4,406 individuals ages 66 and over, all of whom are covered by Medicare is contained in the *NMES1988* data set in the *AER* package. These data were previously analyzed by Deb and Trivedi (1997) and Zeileis *et al.* (2008), from which this account borrows. The objective of the study and these analyses is to create a descriptive, and hopefully predictive, model for the demand for medical care in this elderly population.

{ex:nmes1}

EXAMPLE 11.13: Demand for medical care

The potential response variables in the *NMES1988* data set form a 2×2 set of the combinations

of *place of visit* (office vs. hospital) and (physician vs. non-physician) *practitioner*. Here, we focus on the highest total frequency variable `visits`, recording office visits to a physician. There are quite a few potential predictors, but here we consider only the following:

- `hospital`: number of hospital stays¹²
- `health`: a factor indicating self-perceived health status, with categories "poor", "average" (reference category), "excellent"
- `chronic`: number of chronic conditions
- `gender`
- `school`: number of years of education
- `insurance`: a factor. Is the individual covered by private insurance?

For convenience, these variables are extracted to a reduced data set, `nmes`.

```
> data("NMES1988", package="AER")
> nmes <- NMES1988[, c(1, 6:8, 13, 15, 18)]
```

A quick overview of the response variable, `visits` is shown as simple (unbinned) histograms on the frequency and $\log(\text{frequency})$ scales in Figure 11.24. The zero counts are not as extreme as we have seen in other examples. On the log scale, there is a small, but noticeable spike at 0, followed by a progressive, nearly linear decline, up to about 30 visits.

```
> plot(table(nmes$visits),
+       xlab="Physician office visits", ylab="Frequency")
> plot(log(table(nmes$visits)),
+       xlab="Physician office visits", ylab="log(Frequency)")
```

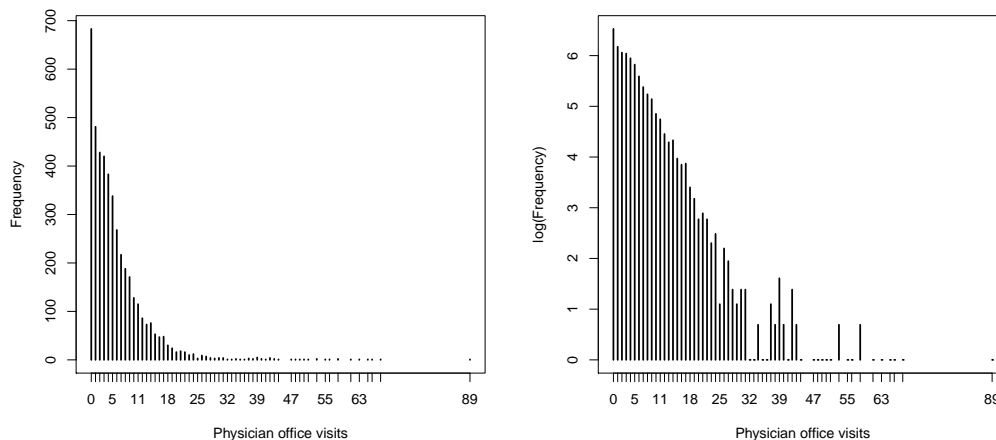


Figure 11.24: Frequency distributions of the number of physician office visits

However as a benchmark, without taking any predictors into account, there is very substantial overdispersion relative to a Poisson distribution, the variance being nearly 8 times the mean.

¹²It is arguable that number of hospitalizations should be regarded as a dependent variable, reflecting another aspect of demand for medical care, rather than as a predictor. We include it here as a predictor to control for its relationship to the outcome `visits`.

```
> with(nmes, c(mean=mean(visits),
+             var=var(visits),
+             ratio=var(visits)/mean(visits)))
```

mean	var	ratio
5.7744	45.6871	7.9120

As before, it is useful to precede formal analysis with a variety of exploratory plots. Figure 11.25 shows a few of these as boxplots, using `cutfac()` to make predictors discrete, and plotting `visits` on a log scale, started at 1. All of these show the expected relationships, e.g., number of office visits increases with numbers of chronic conditions and hospital stays, but decreases with better perceived health status.

```
> op <- par(mfrow=c(1, 3), cex.lab=1.4)
> plot(log(visits+1) ~ cutfac(chronic), data = nmes,
+      ylab = "Physician office visits (log scale)",
+      xlab = "Number of chronic conditions", main = "chronic")
> plot(log(visits+1) ~ health, data = nmes, varwidth = TRUE,
+      ylab = "Physician office visits (log scale)",
+      xlab = "Self-perceived health status", main = "health")
> plot(log(visits+1) ~ cutfac(hospital, c(0:2, 8)), data = nmes,
+      ylab = "Physician office visits (log scale)",
+      xlab = "Number of hospital stays", main = "hospital")
> par(op)
```

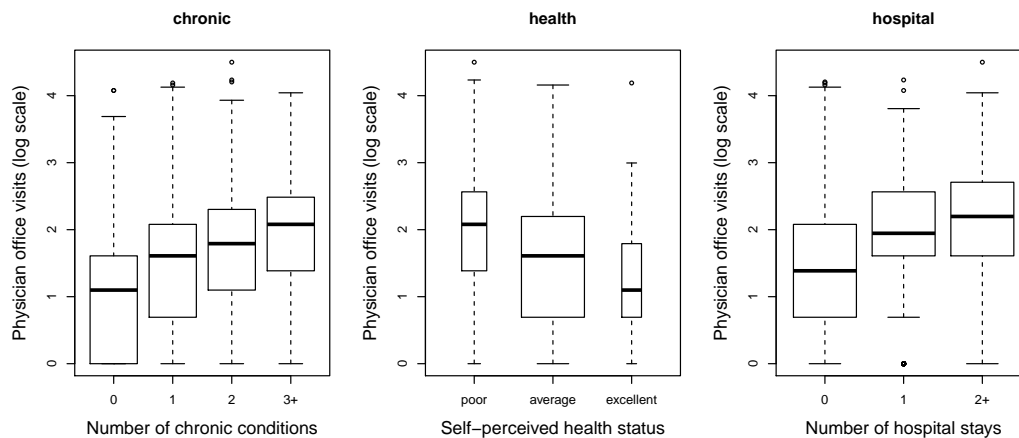


Figure 11.25: Number of physician office visits plotted against some of the predictors

{fig:nmes-boxplots}

Similar plots for insurance and gender show that those with private insurance have more office visits and women slightly more than men.

The relationship with number of years of education could be shown in boxplots by the use of `cutfac(school)`, or with `spineplot()` by making both variables discrete. However, it is more informative (shows the data) to depict this in a smoothed and jittered scatterplot, as in Figure 11.26.

```
> library(ggplot2)
> ggplot(nmes, aes(x=school, y=visits+1)) +
+   geom_jitter(alpha=0.25) +
+   stat_smooth(method="loess", color="red", fill="red", size=1.5, alpha=0.3) +
+   labs(x="Number of years of education", y="log(Physician office visits+1)") +
+   scale_y_log10(breaks=c(1, 2, 5, 10, 20, 50, 100)) + theme_bw()
```

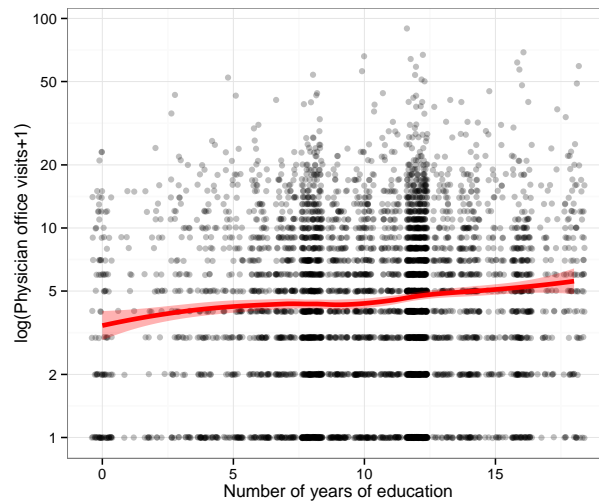


Figure 11.26: Jittered scatterplot of physician office visits against number of years of education, with nonparametric (loess) smooth.

{fig:nmes-school}

As you might expect, there is a small but steady increase in mean office visits with years of education. It is somewhat surprising that there are quite a few individuals with 0 years of education; jittering also shows the greater density of observations at 8 and 12 years.

As in previous examples, a variety of other exploratory plots would be helpful in understanding the relationships among these variables *jointly*, particularly how office visits depends on combinations of two (or more) predictors. Some natural candidates would include mosaic and doubledecker plots (using `cutfac(visits)`), e.g., as in Figure 11.17, and conditional or faceted versions of the boxplots shown in Figure 11.25, each stratified by one (or more) additional predictors. These activities are left as exercises for the reader.

△

11.5.2.1 Fitting models

Most previous analyses of these data have focused on exploring and comparing different types of count data regression models. Deb and Trivedi (1997) compared the adequacy of fit of the negative-binomial, a hurdle NB, and models using finite mixtures of NB models. Zeileis *et al.* (2008) used this data to illustrate hurdle and zero-inflated models using the `countreg` package, while Cameron and Trivedi (1998, 2013) explored a variety of competing models, including 1- and 2-parameter NB models and C -component finite mixture models that can be thought of as generalizations of the 2-component models described in Section 11.4.

In most cases, the full set of available predictors was used, and models were compared using the standard methods for model selection: likelihood-ratio tests for nested models, AIC, BIC and so forth. An exception is Kleiber and Zeileis (2014), who used a reduced set of predictors similar to those employed here, and illustrated the use of rootograms and plots of predicted values for visualizing and comparing fitted models.

This is where model comparison and selection for count data models (and other GLMs) adds another layer of complexity beyond what needs to be considered for classical (Gaussian) linear models, standard logistic regression models and the special case of loglinear models treated earlier. Thus, when we consider and compare different distribution types or link functions, we have to be reasonably confident that the systematic part of the model has been correctly specified (as we noted

in Section 11.3), and is the *same* in all competing models, so that any differences can be attributed to the distribution type. However, lack-of-fit may still arise because the systematic part of the model is incorrect.

In short, we cannot easily compare apples to oranges (different distributions with different regressors), but we also have to make sure we have a good apple to begin with. The important questions are:

- Have all important predictors and control variables have been included in the model?
- Are quantitative predictors represented on the correct scale (via transformations or non-linear terms) so their effects are reasonably additive for the linear predictor?
- Are there important interactions among the explanatory variables?

{ex:nmes2}

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 are all significant. Moreover, the signs of the coefficients are all as we would expect from our exploratory plots. For example, $\log(\text{visits})$ increases with number of hospital stays, chronic conditions and education, and is greater for females and those with private health insurance. So, what's not to like?

```
> summary(nmes.nbin)

Call:
glm.nb(formula = visits ~ ., data = nmes, init.theta = 1.206603534,
link = log)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-3.047  -0.995  -0.295   0.296   5.818

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  0.92926    0.05459  17.02 < 2e-16 ***
hospital     0.21777    0.02018  10.79 < 2e-16 ***
healthpoor   0.30501    0.04851   6.29 3.2e-10 ***
healthexcellent -0.34181  0.06092  -5.61 2.0e-08 ***
```

```

chronic      0.17492    0.01209   14.47 < 2e-16 ***
gendermale  -0.12649    0.03122   -4.05 5.1e-05 ***
school       0.02682    0.00439    6.10 1.0e-09 ***
insuranceyes 0.22440    0.03946    5.69 1.3e-08 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for Negative Binomial(1.2066) family taken to be 1)

    Null deviance: 5743.7  on 4405  degrees of freedom
Residual deviance: 5044.5  on 4398  degrees of freedom
AIC: 24359

Number of Fisher Scoring iterations: 1

            Theta:  1.2066
         Std. Err.:  0.0336

2 x log-likelihood:  -24341.1070

```

This all-main-effects model is relatively simple to interpret, but a more important question is whether it adequately explains the relations of the predictors to the outcome, `visits`.

Significant interactions among the predictors could substantially change the interpretation of the model, and in the end, could affect policy recommendations based on this analysis. This question turns out to be far more interesting and important than the subtle differences among models for handling overdispersion and zero counts.

One simple way to consider whether there are important interactions among the predictors that better explain patient visits is to get simple tests of the additional contribution of each two-way (or higher-way) interaction using the `add1()` function. The formula argument in the call below specifies to test the addition of all two-way terms.

```

> add1(nmes.nbin, . ~ .^2, test="Chisq")

Single term additions

Model:
visits ~ hospital + health + chronic + gender + school + insurance
      Df Deviance   AIC   LRT Pr(>Chi)
<none>                5045 24357
hospital:health      2    5025 24341 19.9  4.7e-05 ***
hospital:chronic     1    5009 24324 35.2  3.0e-09 ***
hospital:gender      1    5044 24358  0.8  0.3650
hospital:school      1    5041 24355  4.0  0.0453 *
hospital:insurance   1    5036 24351  8.0  0.0046 **
health:chronic       2    5005 24322 39.5  2.6e-09 ***
health:gender        2    5040 24357  4.3  0.1172
health:school        2    5030 24347 14.3  0.0008 ***
health:insurance     2    5032 24348 12.9  0.0016 **
chronic:gender       1    5045 24359  0.0  0.9008
chronic:school       1    5043 24357  1.9  0.1705
chronic:insurance    1    5039 24354  5.1  0.0246 *
gender:school        1    5040 24354  4.8  0.0290 *
gender:insurance     1    5042 24357  2.5  0.1169
school:insurance     1    5037 24352  7.2  0.0072 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

From this, we decide to add all two-way interactions among `health`, `hosp` and `numchron`, and also the two-way interaction `health:school`. Other significant interactions could also be explored, but we don't do this here.

```
> nmes.nbin2 <- update(nmes.nbin,
+                       . ~ . + (health+chronic+hospital)^2
+                       + health:school)
```

This model clearly fits much better than the main effects model, as shown by a likelihood ratio test. The same conclusion would result from `anova()`.

```
> lrtest(nmes.nbin, nmes.nbin2)

Likelihood ratio test

Model 1: visits ~ hospital + health + chronic + gender + school + insurance
Model 2: visits ~ hospital + health + chronic + gender + school + insurance +
  health:chronic + hospital:health + hospital:chronic + health:school
#Df LogLik Df Chisq Pr(>Chisq)
1    9 -12171
2   16 -12133   7  74.3      2e-13 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

△

11.5.2.2 Model interpretation: Effect plots

Complex models with more than a few predictors are difficult to understand and explain, even more so when there are interactions among the predictors. As we have noted previously, effect plots (Fox, 1987, Fox and Andersen, 2006) provide a ready solution.

They have the advantage that each plot shows the correct *partial* relation between the response and the variables in the term shown, controlling (adjusting) for all other variables in the model, as opposed to *marginal* plots that ignore all other variables. From these, it is possible to read an interpretation of a given model effect directly from the effect plot graphs, knowing that all variables not shown in a given graph have been controlled (adjusted for) by setting them equal to average or typical values.

A disadvantage is that these plots show only the predicted (fitted) effects under the *given model* (and not the *data*). If relationships of the response to predictors are nonlinear, or important interactions are not included in the model, you won't see this in an effect plot. We illustrate this point using the results of the main effect NB model, `nmes.nbin`, as shown in Figure 11.27.

{ex:nmes2a}

EXAMPLE 11.15: Demand for medical care

```
> library(effects)
> plot(allEffects(nmes.nbin), ylab = "Office visits")
```

All of these panels show the expected relations of the predictors to the `visits` response, and the confidence bands and error bars provide visual tests of the sizes of differences. But they don't tell the full story, because the presence of an important interaction (such as `health:chronic`) means that the effect of one predictor (`health`) differs over the values of the other (`chronic`).

We can see this clearly in effect plots for the model `nmes.nbin2` with interactions. For display purposes, it is convenient here to calculate the fitted effects for model terms over a smaller but representative subset of the levels of the integer-valued predictors, using the `xlevels=` argument to `allEffects()`.

```
> eff_nbin2 <- allEffects(nmes.nbin2,
+ xlevels=list(hospital=c(0:3, 6, 8), chronic=c(0:3, 6, 8), school=seq(0,20,5)))
```

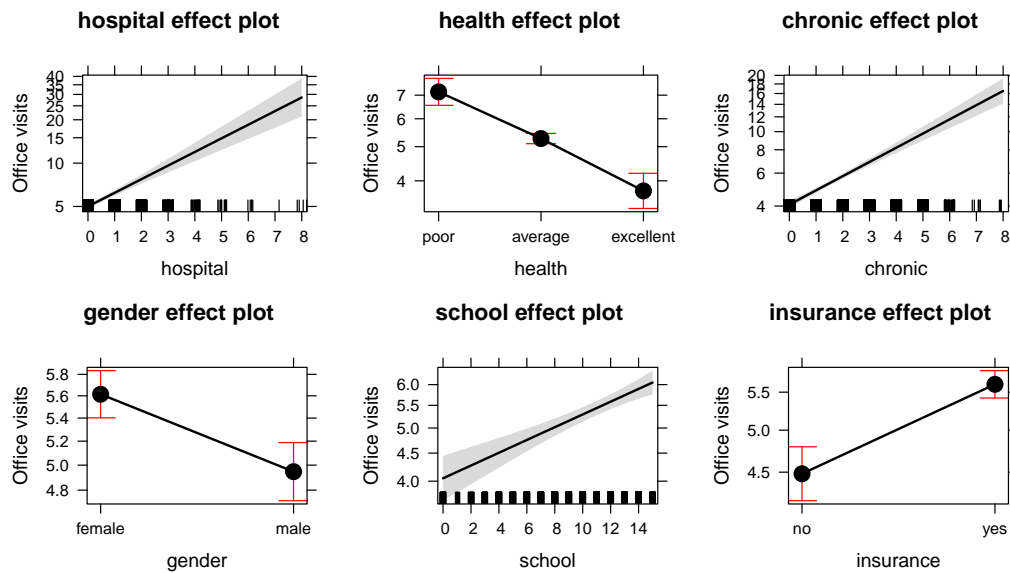


Figure 11.27: Effects plots for the main effects of each predictor in the negative binomial model `nmes.nbin`

{fig:nmes2-eff1}

The result of `allEffects()`, `eff_nbin2`, is a "efflist" object, a list of effects for each *high-order term* in the model. Note that only the terms `gender` and `insurance`, not involved in any interaction, appear as main effects here.

```
> names(eff_nbin2)

[1] "gender"          "insurance"        "health:chronic"
[4] "hospital:health" "hospital:chronic" "health:school"
```

Plotting the entire "efflist" object gives a collection of plots, one for each high-order term, as in Figure 11.27, and is handy for a first look. However, the `plot()` methods for effects objects offer greater flexibility when you plot terms individually using additional options. For example, Figure 11.28 plots the effect for the interaction of health and number of chronic conditions with a few optional arguments. See `help(plot.eff, package="effects")` for the available options.

```
> plot(eff_nbin2, "health:chronic", layout=c(3,1),
+       ylab = "Office visits", colors="blue")
```

The default style shown in Figure 11.28 is a conditional or faceted plot, graphing the response against the X variable with the greatest number of levels, with separate panels for the levels of the other predictor. Alternatively, the effects for a given term can be shown overlaid in a single plot, using the `multiline=TRUE` argument, as shown in Figure 11.29 for the two interactions involving health status. Not only is this style more compact, but it also makes direct comparison of the trends for the other variable easier.

```
> plot(eff_nbin2, "health:chronic", multiline=TRUE, ci.style="bands",
+       ylab = "Office visits", xlab="# Chronic conditions",
+       key.args = list(x = 0.05, y = .80, corner = c(0, 0), columns = 1))
>
> plot(eff_nbin2, "hospital:health", multiline=TRUE, ci.style="bands",
```

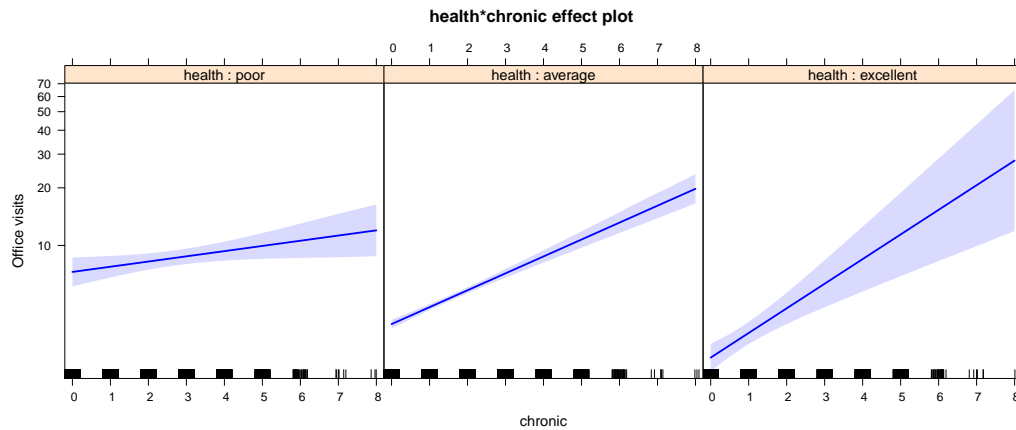


Figure 11.28: Effect plot for the interaction of health and number of chronic conditions in the model `nmes.nbin2`

```
+ ylab = "Office visits", xlab="Hospital stays",
+ key.args = list(x = 0.05, y = .80, corner = c(0, 0), columns = 1))
```

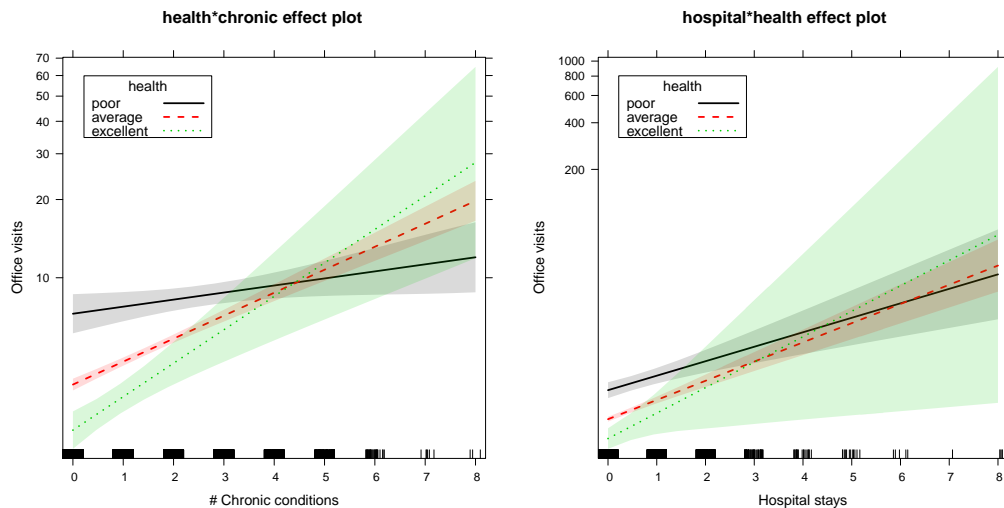


Figure 11.29: Effect plots for the interactions of chronic conditions and hospital stays with perceived health status in the model `nmes.nbin2`

{fig:nmes2-eff3}

From both Figure 11.28 and the left panel of and Figure 11.29, it can be seen that for people with poor health status, the relationship of chronic conditions to office visits is relatively flat. For those who view their health status as excellent, their use of office visits is much more strongly related to their number of chronic conditions.

The interaction of perceived health status with number of hospital stays (right panel of Figure 11.29) shows that the difference in office visits according to health status is mainly important only for those with 0 or 1 hospital stays.

The remaining two interaction effects are plotted in Figure 11.30. The interaction of hospital stays and number of chronic conditions (left panel of Figure 11.30) has a clearly interpretable pattern: for those with few chronic conditions, there is a strong positive relationship between hospital

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))
```

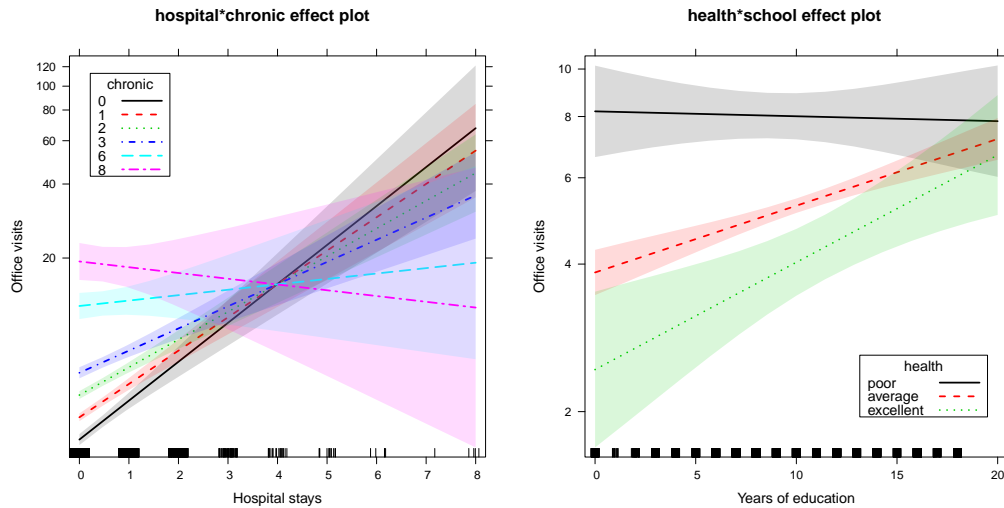


Figure 11.30: Effect plots for the interactions of chronic conditions and hospital stays and for health status with years of education in the model `nmes.nbin2`

{fig:nmes2-eff4}

Finally, the interaction of `health:school` is shown in the right panel of Figure 11.30. It can be readily seen that for those of poor health, office visits are uniformly high, and have no relation to years of education. Among those of average or excellent health, office visits increase with years of education in roughly similar ways. \triangle

11.5.2.3 More model wrinkles: Nonlinear terms

Effect plots such as those above are much easier to interpret than tables of fitted coefficients. However, we emphasize that these only reflect the *fitted model*. It might be that the effects of both `hospital` and `chronic` are nonlinear (on the scale of $\log(\text{visits})$). In assessing this question, we increase the complexity of model and try to balance parsimony against goodness-of-fit, but also assure that the model retains a sensible interpretation.

{ex:nmes3}

EXAMPLE 11.16: Demand for medical care

The simplest approach is to use `poly(hosp, 2)` and/or `poly(numchron, 2)` to add possible quadratic (or higher power) relations to the model `nmes.nbin2` containing interactions studied above. A slightly more complex model could use `poly(hosp, numchron, degree=2)` for a response-surface model in these variables. A significantly improved fit of such a model is evidence for nonlinearity of the effects of these predictors. This is easily done using `update()`:

```
> nmes.nbin3 <- update(nmes.nbin2, . ~ . + I(chronic^2) + I(hospital^2))
```

This model is equivalent to the long-form version below:

```
> nmes.nbin3 <- glm.nb(visits ~ poly(hospital, 2) + poly(chronic, 2) +
+ insurance + school + gender +
+ (health+chronic+hospital)^2 + health:school, data = nmes)
```

Comparing these models using `anova()`, we see that there is a substantial improvement in the model fit by including these nonlinear terms. The quadratic model also fits best by AIC and BIC.

```
> anova(nmes.nbin, nmes.nbin2, nmes.nbin3)

Likelihood ratio tests of Negative Binomial Models

Response: visits

1
2
3 hospital + health + chronic + gender + school + insurance + he
  theta Resid. df    2 x log-lik.    Test    df LR stat.
1 1.2066      4398      -24341
2 1.2354      4391      -24267 1 vs 2    7    74.307
3 1.2446      4389      -24245 2 vs 3    2    22.278
  Pr(Chi)
1
2 1.9829e-13
3 1.4537e-05

> vcdExtra::LRstats(nmes.nbin, nmes.nbin2, nmes.nbin3)

Likelihood summary table:
      AIC   BIC LR Chisq   Df Pr(>Chisq)
nmes.nbin 24359 24417    5045 4398  2.2e-11 ***
nmes.nbin2 24299 24401    5047 4391  1.1e-11 ***
nmes.nbin3 24281 24396    5049 4389  8.5e-12 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

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))
```

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 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,

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

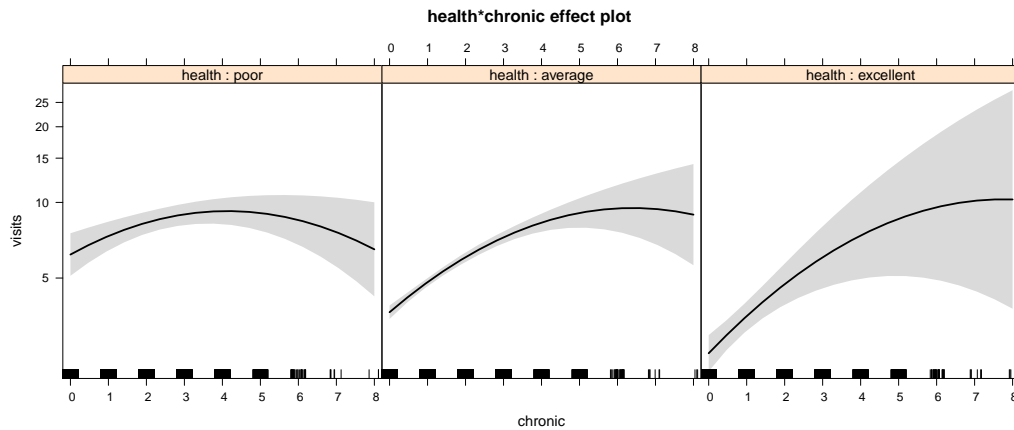


Figure 11.31: Effect plot for the interaction of health and number of chronic conditions in the quadratic model `nmes.nbin3`

{fig:nmes3-eff1}

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` 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 `glm.nb()`), or `family=negbin(theta)` when θ is taken as fixed, for example using the value `theta=1.24` available from models `nmes.nbin2`, and `nmes.nbin3`.

```
> library(mgcv)
> nmes.gamnb <- gam(visits ~ s(hospital, k=3) + s(chronic, k=3) +
+                     insurance + school + gender +
+                     (health+chronic+hospital)^2 + health:school,
+                     family=nb(), data = nmes)
```

The key feature here is the specification of the smooth terms for `s(hospital, k=3)` and `s(chronic, k=3)`, where `k=3` specifies the dimension of the basis used to represent the smooth term. There are many other possibilities with `gam()`, but these are beyond the scope of this example.

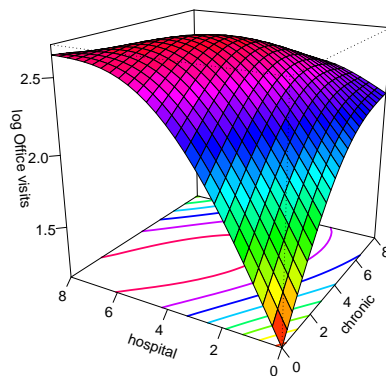
We could again visualize the predicted values from this model using effect plots. However a different approach is to visualize the *fitted surface* in 3D, using a range of values for two of the predictors, and controlling for the others.

The `rsm` package provides extensions of the standard `contour()`, `image()` and `persp()` functions for this purpose. The package provides S3 methods (e.g., `persp.lm()`) for “lm” objects, or classes (such as “`negbin`” and “`glm`”) that inherit methods from `lm`. The calculation of fitted values in these plots use the applicable `predict()` method for the model object. As in effect plots, the remaining predictors are controlled at their average values (or other values specified in the `at` argument).

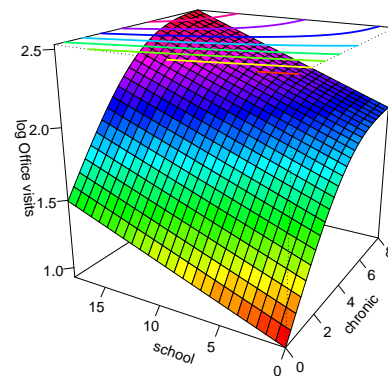
Two such plots are shown in Figure 11.32. The left panel shows the interaction of hospital stays and chronic conditions, included in the model with smoothed terms for their main effects. The right

panel shows the joint effects of years of education and chronic conditions on office visits, but there is no interaction of these variables in the GAM model `nmes.gamnb`. These plots use `rainbow()` colors to depict the predicted values of office visits. Contours of these values are projected into the bottom or top plane with corresponding color coding.¹³

```
> library(rsm)
> persp(nmes.gamnb, hospital ~ chronic, zlab="log Office visits",
+       col=rainbow(30), contour=list(col="colors", lwd=2),
+       at=list(school=10, health='average'), theta=-60)
>
> persp(nmes.gamnb, school ~ chronic, zlab="log Office visits",
+       col=rainbow(30), contour=list(col="colors", lwd=2, z="top"),
+       at=list(hospital=0.3, health='average'), theta=-60)
```



Slice at school = 10, health = average



Slice at hospital = 0.3, health = average

Figure 11.32: Fitted response surfaces for the relationships among chronic conditions, number of hospital stays and years of education to office visits in the generalized additive model, `nmes.gamnb`

{fig:nmes3-rsm}

A simple, credible interpretation of the plot in the left panel is that office visits rise steeply initially with both hospital stays and number of chronic conditions, and then levels off. For those with no chronic conditions, the effect of hospital stays rises to a higher level compared with the effect of chronic conditions among those who have had no hospital stays. However, as we have seen before, the data is quite thin at the upper end of these predictors, and this plot does not show model uncertainty.

The right panel of Figure 11.32 illustrates the form of model predictions for a term where one variable (`chronic`) is treated as possibly nonlinear using a smooth `s()` effect, the other is treated as linear (`school`), and no interaction between these is included in the model. At each fixed value of `chronic`, increasing education results in greater office visits. At each fixed value of `school`, the number of chronic conditions shows a steep increase in office visits initially, leveling off toward higher levels, but these all have the same predicted shape.

△

¹³The vignette `vignette("rsm-plots", package="rsm")` illustrates some of these options.

11.6 Diagnostic plots for model checking

{sec:glm-diag}

Models, of course, are never true, but fortunately it is only necessary that they be useful.

G. E. P. Box, *Some Problems of Statistics of Everyday Life*, 1979, p. 2

Most of the model diagnostic methods for classical linear models extend in a relatively direct way to GLMs. These include (a) plots of residuals of various types, (b) diagnostic measures and plots of leverage and influence, as well as some (c) more specialized plots (component-plus-residual plots, added-variable plots) designed to show the specific contribution of a given predictor among others in a linear model. These methods were described in Section 7.5 in the context of logistic regression, and most of that discussion is applicable here in wider GLM class.

One additional complication here is that in any GLM we are specifying: (a) the distribution of the random component, which for count data models may also involve a dispersion parameter or other additional parameters; (b) the form of the linear predictor, $\eta = \mathbf{x}^T \boldsymbol{\beta} = \beta_0 + \beta_1 x_1 + \dots$, where all important regressors have been included, and on the right scale; (c) the correct link function, $g(\mu) = \eta$ transforming the conditional mean of the response y to the predictor variables where they have linear relationships.

Thus, there are a lot of things that could go wrong, but the famous quote from George Box should remind us that all models are approximate, and the goal for model diagnosis should be an adequate model, useful for description, estimation or prediction as the case may be. What is most important is that our models cannot be misleadingly wrong, that is they should not affect substantive conclusions or interpretation.

11.6.1 Diagnostic measures and residuals for GLMs

Estimation of GLMs by maximum likelihood uses an iterative weighted least squares (IWLS) algorithm, and many of the diagnostic measures for these models are close counterparts of their forms for classical linear models. Roughly speaking, these follow from replacing \mathbf{y} and $\hat{\mathbf{y}}$ in least squares diagnostics by a “working response” and $\hat{\boldsymbol{\eta}}$, replacing the residual variance $\hat{\sigma}^2$ by $\hat{\phi}$, and using a weighted form of the Hat matrix.

11.6.1.1 Leverage

Hat values, h_i , measuring leverage or the potential of an observation to affect the fitted model are defined as the diagonal elements of the hat matrix \mathbf{H} , using the weight matrix \mathbf{W} from the final IWLS iteration. This has the same form as in a weighted least squares regression using a fixed \mathbf{W} matrix:

$$\mathbf{H} = \mathbf{W}^{1/2} \mathbf{X} (\mathbf{X}^T \mathbf{W} \mathbf{X})^{-1} \mathbf{X}^T \mathbf{W}^{1/2}.$$

In contrast to OLS, the weights depend on the \mathbf{y} values as well as the \mathbf{X} values, so high leverage observations do not necessarily reflect only unusualness in the space of the predictors.

11.6.1.2 Residuals

Several types of residuals can be defined starting from the goodness-of-fit measures discussed in Section 11.1.3. The **raw residual** or **response residual** is simply the difference $y_i - \hat{\mu}_i$ between the observed response y_i and the estimated mean, $\hat{\mu} = g^{-1}(\hat{\boldsymbol{\eta}}) = g^{-1}(\mathbf{x}_i^T \hat{\boldsymbol{\beta}})$.

From this, the **Pearson residual** is defined as

{eq:res-pearson}

$$r_i^P = \frac{y_i - \hat{\mu}_i}{\sqrt{\hat{\mathcal{V}}(y_i)}} \quad (11.11)$$

and the **deviance residual** is defined as the signed square root of the contribution of observation i to the deviance in Eqn. (11.4).

$$r_i^D = \text{sign}(y_i - \hat{\mu}_i) \sqrt{d_i} \quad (11.12) \quad \{\text{eq:res-deviance}\}$$

The Pearson and deviance residuals do not account for dispersion or for differential leverage (which makes their variance smaller), so **standardized residuals** (sometimes called *scaled* residuals) can be calculated as

$$\tilde{r}_i^P = \frac{r_i^P}{\sqrt{\hat{\phi}(1 - h_i)}} \quad (11.13) \quad \{\text{eq:res-pearson-s}\}$$

$$\tilde{r}_i^D = \frac{r_i^D}{\sqrt{\hat{\phi}(1 - h_i)}} \quad (11.14) \quad \{\text{eq:res-deviance-s}\}$$

These have approximate standard normal $\mathcal{N}(0, 1)$ distributions, and will generally have quite similar values (except for small values in $\hat{\mu}$). Consequently, convenient thresholds like $|\tilde{r}_i| > 2$ or $|\tilde{r}_i| > 4$ are useful for identifying unusually large residuals.

Finally, the **studentized residual** (or *deletion* residual) gives the standardized residual that would result omitting each observation in turn and calculating the change in the deviance. Calculating these exactly would require refitting the model n times, but an approximation is

$$\tilde{r}_i^S = \text{sign}(y_i - \hat{\mu}_i) \sqrt{(\tilde{r}_i^D)^2 + (\tilde{r}_i^P)^2 h_i / (1 - h_i)} . \quad (11.15)$$

From the theory of classical linear models, these provide formal outlier tests for individual observations (Fox, 2008, §11.3) as a *mean-shift* outlier model that dedicates an additional parameter to fit observation i exactly. To correct for multiple testing and a focus on the largest absolute residuals, it is common to apply a Bonferroni adjustment to the p -values of these tests, multiplying them by n .

For a class "glm" object, the function `residuals(object, type)` returns the unstandardized residuals for `type="pearson"` or `type="deviance"`.¹⁴ 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 $\hat{\beta} - \hat{\beta}_{(-i)}$ in all p coefficients in the model. The approximation used in `cooks.distance()` is

$$C_i = \frac{\tilde{r}_i h_i}{\hat{\phi} p (1 - h_i)} .$$

This follows Williams (1987), but scales the result by the estimated dispersion $\hat{\phi}$ as an approximate $F_{p, n-p}$ statistic rather than χ_p^2 .

¹⁴Other types include raw response residuals (`type="response"`), working residuals (`type="working"`) and partial residuals (`type="partial"`).

- DFFITS, the standardized signed measure of the difference of the fitted value $\hat{\mu}_i$ using all the data and the value $\hat{\mu}_{(-i)}$ omitting observation i .

{ex:phdpubs5}

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.

```
> op <- par(mfrow=c(2,2), mar=c(4,4,2,1)+.1, cex.lab=1.2)
> plot(phd.nbin)
> par(op)
```

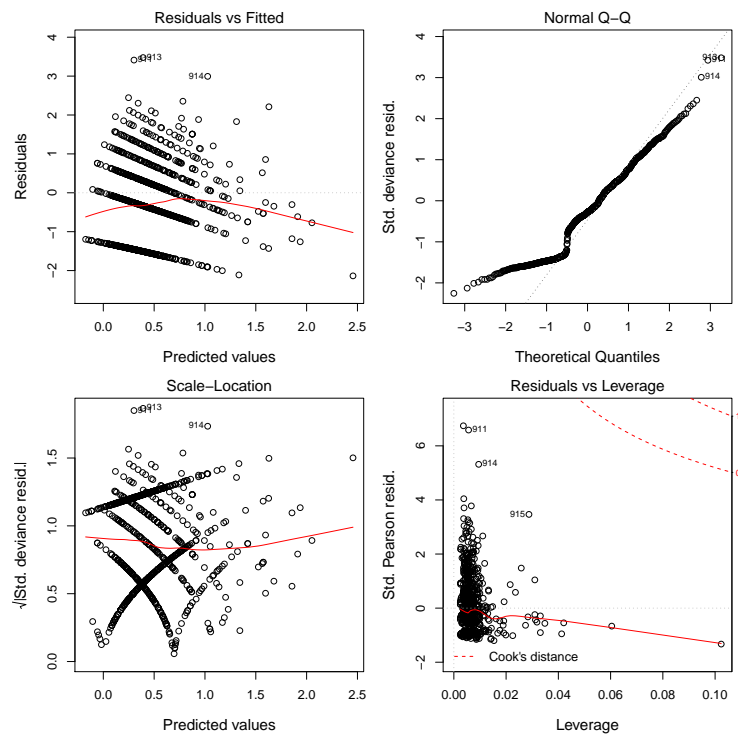


Figure 11.33: Default diagnostic plots for the negative-binomial model fit to the PhdPubs data.

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)
```

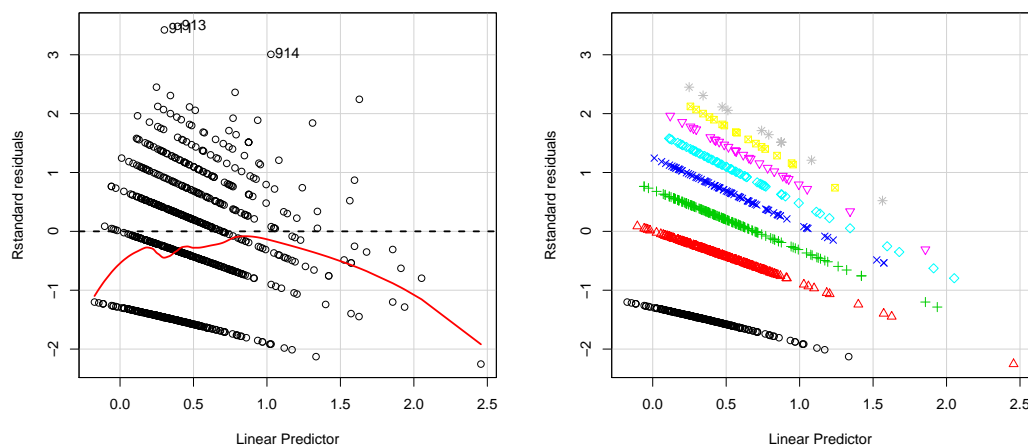


Figure 11.34: Plots of residuals against the linear predictor using `residualPlot()`. The right panel shows that the diagonal bands correspond to different values of the discrete response.

```
:phdpubs5-resplot1}
```

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.

```
> residualPlot(phd.nbin, "mentor", type="rstudent",
+             quadratic=TRUE, col.smooth="red", col.quad="blue", id.n=3)
> residualPlot(phd.nbin, "phdprestige", type="rstudent",
+             quadratic=TRUE, col.smooth="red", col.quad="blue", id.n=3)
```

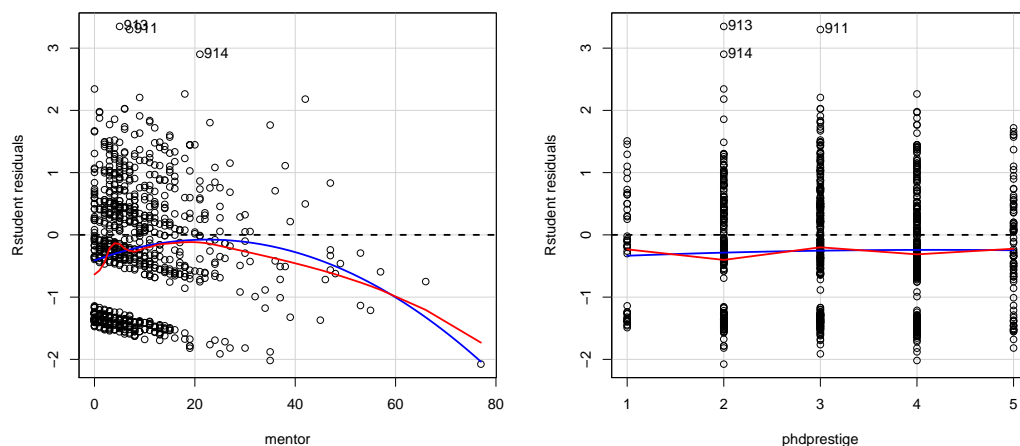


Figure 11.35: Plots of residuals against two predictors in the `phd.nbin` model. Such plots should show no evidence of a systematic trend for a good-fitting model.

```
{fig:phdpubs5-resplot2}
```


In the plot at the left for number of articles by the student's mentor, the curvature is quite pronounced: at high values of `mentor`, nearly all of the residuals are negative, these students publishing fewer articles than would be expected. This would indicate a problem in the scale for `mentor` if there were more observations at the high end; but only about 1.5% points occur for `mentor > 45`, so this can be discounted.

Figure 11.36 gives a better version of the influence plot shown in the lower right panel of Figure 11.33. This plots studentized (deletion) residuals against leverage, showing the value of Cook's distance by the area of the bubble symbol.

```
> influencePlot(phd.nbin)
```

	StudRes	Hat	CookD
328	-2.0762	0.1023449	0.18325
913	3.3488	0.0036473	0.16652
915	2.1810	0.0287496	0.24345

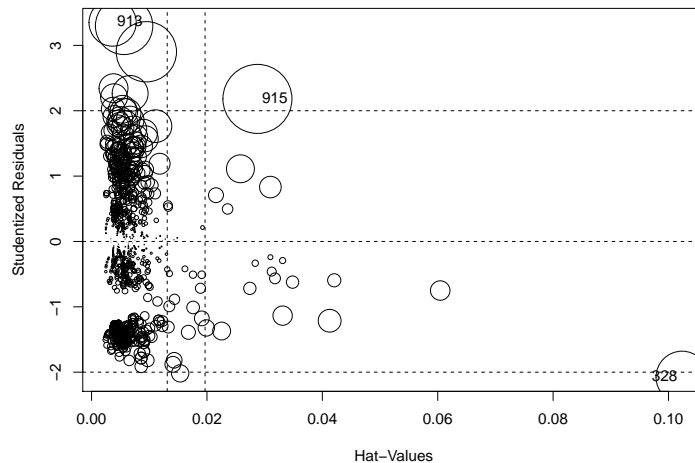


Figure 11.36: Influence plot showing leverage, studentized residuals and Cook's distances for the negative-binomial model fit to the PhdPubs data. Conventional cutoffs for studentized residuals are shown by dashed horizontal lines at ± 2 ; vertical lines show 2 and 3 times the average hat-value.

Several observations are considered noteworthy, because of one or more of large absolute residual, large leverage or large Cook's distance. `influencePlot()` uses different default rules for point labeling than does the `plot()` method, but provides many options to control the details. Observation 328 stands out as having the largest leverage and a large negative residual; case 913 has the largest absolute residual, but is less influential than case 915.¹⁵

The `outlierTest()` function in `car` gives a formal test of significance of the largest absolute studentized residuals, with a Bonferroni-adjusted p -value accounting for choosing the largest values among n such tests. Individually, case 913 is extreme, but it is not at all extreme among $n = 915$ such tests, each using $\alpha = .05$.

¹⁵The higher case numbers appear in these plots and diagnostics because the data set *PhdPubs* had been sorted by the response, articles.

```
> outlierTest(phd.nbin)

No Studentized residuals with Bonferonni p < 0.05
Largest |rstudent|:
      rstudent unadjusted p-value Bonferonni p
913      3.3488      0.00084491      0.77309
```

This example started with the negative-binomial model, the best-fitting from the previous examples. It highlighted a few features of the data not seen previously and worth considering, but doesn't seriously challenge the substantive interpretation of the model. This is what we hope for from model diagnostic plots.

△

11.6.2 Quantile-quantile and half-normal plots

As we noted above, in theory the standardized and studentized Pearson and deviance residuals have approximate standard normal $\mathcal{N}(0, 1)$ distributions (in large samples) when the fitted model is correct. This suggests a plot of the sorted residuals, $r_{(i)}$, against the corresponding expected values, $z_{(i)}$ an equal-sized sample of size n would have in a normal distribution.¹⁶

If the distribution of the residuals is approximately normal, the points $(r_{(i)}, z_{(i)})$ should lie along a line with unit slope through the origin; systematic or individual departure from this line signals a potential violation of assumptions. The expected values are typically calculated as $z_{(i)} = \Phi^{-1}\{(i - \frac{3}{8})/(n + \frac{1}{4})\}$, where $\Phi^{-1}(\bullet)$ is the inverse normal, or normal quantile function, `qnorm()` in R.

Such plots, called *normal quantile plots* or *normal QQ plots*, are commonly used for GLMs with a quantitative response variable. The upper right panel of Figure 11.33 illustrates the form of such plots produced by `plot()` for a "glm" object.

One difficulty with the default plots is that it is hard to tell to what extent the points deviate from the unit line because there is no visual reference for the line or envelope to indicate expected variability about that line. This problem is easily remedied using `qqPlot()` from `car`.

Figure 11.37 shows the result for the model `phd.nbin`. The envelope lines used here are at the quartiles of the expected normal distribution. They suggest a terrible fit, but, surprisingly, the largest three residuals are within the envelope.

```
> qqPlot(rstudent(phd.nbin), id.n=3,
+        xlab="Normal quantiles", ylab="Studentized residuals")

913 911 914
915 914 913
```

For GLMs with discrete responses, such plots are often disappointing, even with a reasonably good-fitting model, because: (a) possible outliers can appear at both the lower and upper ends of the distribution of residuals; (b) the theoretical normal distribution used to derive the envelope may not be well approximated in a given model.

Atkinson (1981, 1987) suggested a more robust and useful version of these QQ plots: half normal plots, with simulated confidence envelopes. The essential ideas are:

- Model departures and outliers are often easier to see for discrete data when the *absolute values* of residuals are plotted, because large positive and negative values are sorted together. This gives the *half-normal plot*, in which the absolute values of residuals, arranged in increasing

¹⁶The subscripted notation $r_{(i)}$, and $z_{(i)}$ here denotes an *order statistic*, the i^{th} largest value in a set arranged in increasing order.

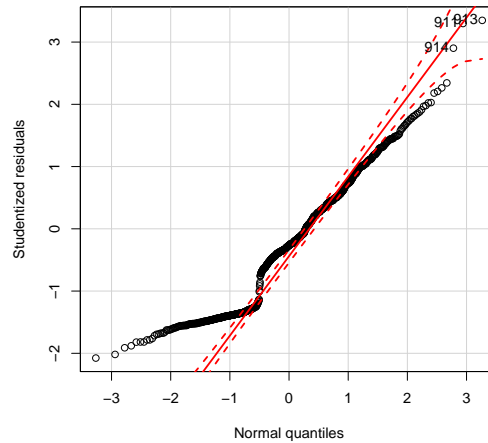


Figure 11.37: Normal QQ plot of the studentized residuals from the NB model for the PhdPubs data. The normal-theory reference line and confidence envelope are misleading here.

{fig:phdpubs6-qqplot}

order, $|r|_{(i)}$, are plotted against $|z|_{(i)} = \Phi^{-1}\{(n + i - \frac{1}{8})/(2n + \frac{1}{2})\}$. All outliers will then appear in the upper right of such a plot, as points separated from the trend of the remaining cells.

- The normal-theory reference line, $|r|_{(i)} = |z|_{(i)}$ and the normal-theory confidence envelope can be replaced by simulating residuals from the assumed distribution, that need not be normal. The reference line is taken as the mean of S simulations and the envelope with $1 - \alpha$ coverage is taken as the $(\alpha/2, 1 - \alpha/2)$ quantiles of their values.
- Specifically, for a GLM, S sets of random observations $\mathbf{y}_j, j = 1, 2, \dots, S$ are generated from the fitted model, each with mean $\hat{\boldsymbol{\mu}}$, the fitted values under from the model and with the *same* distribution. In R, this is readily accomplished using the generic `simulate()` function; the random variation around $\hat{\boldsymbol{\mu}}$ uses `rnorm()`, `rpois()`, `rnegbin()`, etc., as appropriate for the family of the model.
- The same model is then fit to each simulated \mathbf{y}_j , giving a new set of residuals for each simulation. Sorting their absolute values then gives the simulation distribution used as reference for the observed residuals.

At the time of writing there is no fully general implementation of these plots in R, but the technique is not too difficult and is sufficiently useful to illustrate here.

{ex:phdpubs6}

EXAMPLE 11.18: Publications of PhD candidates

First, calculate the sorted absolute values of the residuals $|r|_{(i)}$ and their expected normal values, $|z|_{(i)}$. The basic plot will be `plot(expected, observed)`.

```
> observed <- sort(abs(rstudent(phd.nbin)))
> n <- length(observed)
> expected <- qnorm((1:n + n - 1/8)/(2*n + 1/2))
```

Then, use `simulate()` to generate $S = 100$ simulated response vectors around the fitted values in the model. Here this uses the negative-binomial random number generator (`rnegbin()`) with the same dispersion value ($\hat{\theta} = 2.267$) estimated in the model. The result, called `sims` here, is a data frame of $n = 915$ rows and $S = 100$ columns, named `sim_1`, `sim_2`, `...`

```
> S <- 100
> sims <- simulate(phd.nbin, nsim=S)
> simdat <- cbind(PhdPubs, sims)
```

The next step is computationally intensive, because we have to fit the NB model $S = 100$ times and a little bit tricky, because we need to use the same model formula as the original, but with the simulated y . We first define a function `resids` to do this for a given y , and then use a loop to calculate them all. To save computing time, the coefficients from the `phd.nbin` model are used as starting values.

```
> # calculate residuals for one simulated data set
> resids <- function(y)
+   rstudent(glm.nb(y ~ female + married + kid5 + phdprestige + mentor,
+                   data=simdat, start=coef(phd.nbin)))
> # do them all ...
> simres <- matrix(0, nrow(simdat), S)
> for(i in 1:S) {
+   simres[,i] <- sort(abs(resids(dat[,paste("sim", i, sep="_")]))))
+ }
```

We can then use `apply()` to compute the summary measures defining the center and limits for the simulated confidence interval.

```
> envelope <- 0.95
> mean <- apply(simres, 1, mean)
> lower <- apply(simres, 1, quantile, prob=(1 - envelope)/2)
> upper <- apply(simres, 1, quantile, prob=(1 + envelope)/2)
```

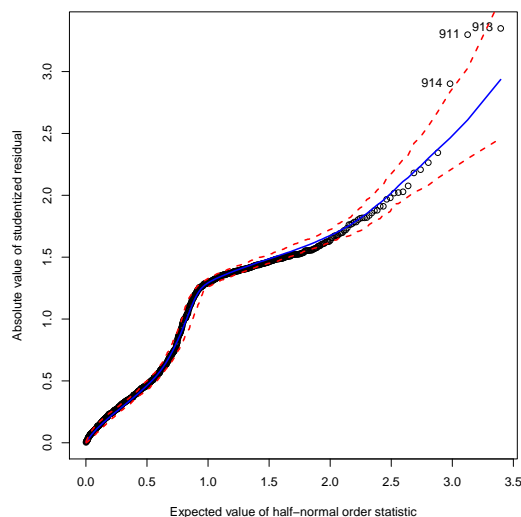


Figure 11.38: Half-normal QQ plot of studentized residuals for the NB model fit to the PhdPubs data. The reference line and confidence envelope reflect the mean and (2.5%, 97.5%) quantiles of the simulation distribution under the negative-binomial model for the same data.

{fig:phd-halfnorm}

Finally, plot the observed against expected absolute residuals as points, and add the lines for the confidence envelope, producing Figure 11.38.

```

> plot(expected, observed,
+       xlab='Expected value of half-normal order statistic',
+       ylab='Absolute value of studentized residual')
> lines(expected, mean, lty=1, lwd=2, col="blue")
> lines(expected, lower, lty=2, lwd=2, col="red")
> lines(expected, upper, lty=2, lwd=2, col="red")
> identify(expected, observed, labels=names(observed), n=3)

```

The shape of the QQ plot in Figure 11.37 shows a peculiar bend at low values and the half-normal version in Figure 11.38 has a peculiar hump in the middle. What could be the cause?

Figure 11.39 shows two additional plots of the studentized residuals that give a clear answer. The density plot at the left shows a strongly bimodal distribution of the residuals. An additional plot at the right of residuals against the log(response) confirms the guess that the lower mode corresponds to those students who published no articles— excess zeros again!

```

> # examine distribution of residuals
> res <- rstudent(phd.nbin)
> plot(density(res), lwd=2, col="blue",
+       main="Density of studentized residuals")
> rug(res)
> # why the bimodality?
> plot(jitter(log(PhdPubs$articles+1), factor=1.5), res,
+       xlab="log (articles+1)", ylab="Studentized residual")

```

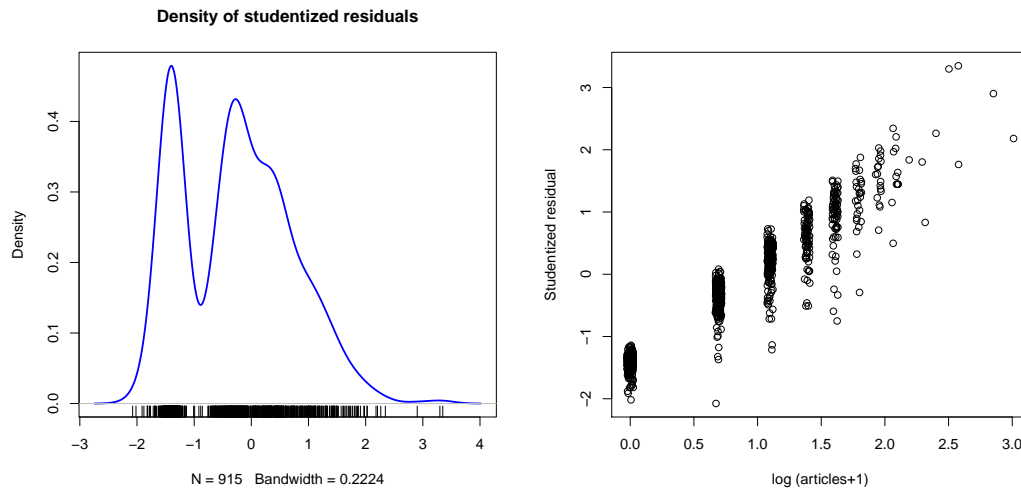


Figure 11.39: Further plots of studentized residuals. Left: density plot; right: residuals against $\log(\text{articles}+1)$

Now we have something to worry about that *could* affect substantive interpretation or conclusions from this analysis using the NB model, but not accounting for excess zeros. If we believe, following Long (1997), that there is a separate latent class of students who don't publish, it would be sensible to fit a zero-inflated NB model, perhaps with a different subset of predictors for the zero component. The alternative theory of a “hurdle” to a first publication suggests fitting a hurdle model. We leave these as exercises for the reader.

△

11.7 Multivariate response GLM models

{sec:glm-multiv}

Far better an approximate answer to the right question, which is often vague, than an exact answer to the wrong question, which can always be made precise.

John W. Tukey (1962), *The future of data analysis*

As noted in Section 10.4, in many studies, there may be several response variables along with one or more explanatory variables, and it is useful to try to model some properties of their joint distribution as well as their separate dependence on the predictors. In the current chapter, the case study (Section 11.5.2) of demand for medical care by the elderly provides a relevant example. There are actually four indicators of medical care, a 2×2 set of (office vs. hospital) place and (physician vs. non-physician) practitioner. That case study analysed only the office visits by physicians.

This section describes a few steps in this direction. To provide some context, we begin with a capsule overview of classical multivariate response models.

In the case of classical linear models with Gaussian error distributions, the model for a univariate response, $\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}$, with $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$ extends quite readily to the **multivariate linear model** (MLM) for q response variables, $\mathbf{Y} = \{\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_q\}$. The MLM has the form

$$\underset{(n \times q)}{\mathbf{Y}} = \underset{(n \times p)}{\mathbf{X}} \underset{(p \times q)}{\mathbf{B}} + \underset{(n \times q)}{\mathbf{E}} \quad (11.16) \quad \{\text{eq:mlm}\}$$

where \mathbf{Y} is a matrix of n observations on q response variables; \mathbf{X} is a model matrix with columns for p regressors, typically including an initial column of 1s for the regression constant; \mathbf{B} is a matrix of regression coefficients, one column for each response variable; and \mathbf{E} is a matrix of errors.

It is important to note that:

- The maximum likelihood estimator of \mathbf{B} in the MLM is equivalent to the result of fitting q separate univariate models for the individual responses and joining the coefficients columnwise, giving

$$\hat{\mathbf{B}} = \{\hat{\boldsymbol{\beta}}_1, \hat{\boldsymbol{\beta}}_2, \dots, \hat{\boldsymbol{\beta}}_q\} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{Y}$$

- Procedures for statistical inference (hypothesis tests, confidence intervals), however, take account of the correlations among the responses. Multivariate tests can therefore be more powerful than separate univariate tests under some conditions.
- A unique feature of the MLM stems from the assumption of multivariate normality of the errors, so that each row, $\boldsymbol{\epsilon}_i^\top$ of \mathbf{E} is assumed to be distributed independently, $\boldsymbol{\epsilon}_i^\top \sim \mathcal{N}_q(\mathbf{0}, \boldsymbol{\Sigma})$, where $\boldsymbol{\Sigma}_{q \times q}$ is the error covariance matrix, constant across observations, like σ^2 in univariate models. Then, the conditional distributions of $\mathbf{y}_j | \mathbf{X}$ are all univariate normal, all bivariate distributions, $\mathbf{y}_j, \mathbf{y}_k | \mathbf{X}$ are bivariate normal, and any linear combination of the conditional \mathbf{y}_j is univariate normal.
- Consequently, all relationships among the \mathbf{y}_j s can be summarized by correlations and relationships between the \mathbf{y}_j s and \mathbf{x} s by linear regressions. These can be visualized using **data ellipses** (Friendly *et al.*, 2013) and hypothesis tests in the MLM can be visualized by ellipses using **hypothesis-error plots** (Friendly, 2007, Fox *et al.*, 2009).

This generality of the MLM is lost, however, when we move to multivariate response models in the non-Gaussian case. For binomial responses, Section 10.4 described several approaches toward a multivariate logistic regression model that attempt to separate the marginal dependence of each \mathbf{y} on the \mathbf{x} s from the relationship of the association among the \mathbf{y}_j s on the \mathbf{x} s. The bivariate logistic model for $(\mathbf{y}_1, \mathbf{y}_2)$ for example, was parameterized (see Eqn. (10.15)) in terms of submodels for a logit for each response, $\eta_1 = \mathbf{x}^\top \boldsymbol{\beta}_1$, $\eta_2 = \mathbf{x}^\top \boldsymbol{\beta}_2$ and a submodel for the odds ratio, $\theta_{12} = \mathbf{x}^\top \boldsymbol{\beta}_{12}$.

The situation becomes more difficult for multivariate count data responses, because parametric

approaches to their joint distribution (e.g., a multivariate Poisson distribution) given a set of explanatory variables are computationally and analytically intractable. Cameron and Trivedi (2013, Chapter 8) provide a detailed description of the problems and some solutions for the bivariate case, including bivariate Poisson, negative-binomial and hurdle models.

Consequently, only a few special cases have been worked out theoretically, and mostly for the bivariate case. For example, King (1989) described a seemingly unrelated bivariate Poisson model for two correlated count variables. This models the separate linear predictors for y_1 and y_2 as

$$\begin{aligned} g(\mu_1) &= x_1^\top \beta_1 \\ g(\mu_2) &= x_2^\top \beta_2, \end{aligned}$$

with the covariance between y_1 and y_2 represented as ξ . As in the MLM, the coefficients have the same point estimates as in equation-by-equation Poisson models. However, there is a gain in efficiency (reduced standard errors) resulting from a bivariate full-information maximum likelihood solution, and efficiency increases with the covariance ξ between the two count variables.

As a result, for lack of a fully general model for multivariate count data, one simple approach is to employ a method for simultaneous estimation of the equation-by-equation coefficients, accepting some loss of efficiency. This allows for hypothesis tests that may not be most powerful, but provide approximate answers to more interesting questions. We can supplement this with separate analysis of the dependencies among the responses, and how these vary with the explanatory variables.

In R, the VGAM package is the most general available package for analysis of multivariate response GLMs. For multivariate count data, it provides for both Poisson and negative-binomial models. For NB models, the dispersion parameters $\theta_j = \alpha_j^{-1}$ can be allowed to vary with the predictors via a GLM of the form $\log \theta_j = x^\top \gamma_j$ or can be constrained to be “intercept-only,” $\log \theta_j = \gamma_{0j}$, giving separate global dispersion estimates for each response. In the latter case, the resulting coefficients are the same as fitting a separate model for each response using `glm.nb()`.

{ex:nmes4}

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"
```

11.7.0.1 Analyzing correlations: HE plots

For purely descriptive purposes, a useful starting point is often an analysis of the log (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 a 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.¹⁷ 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` 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.

```
> 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)
```

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 overview of the relationships among the y and x variables.

11.7.0.2 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",
```

¹⁷When the errors, 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.

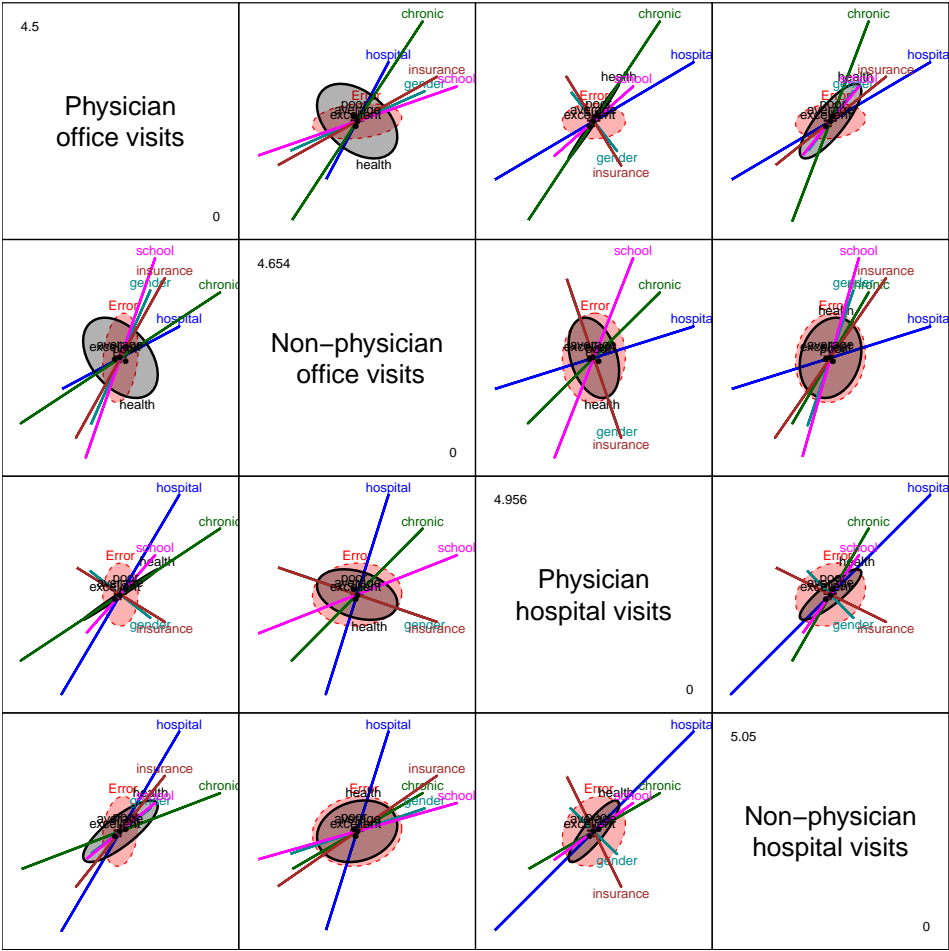


Figure 11.40: Pairwise HE plots for all responses in the nmes2 data.

{fig:nmes4-hepairs}

```
+ 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 the 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"),
+ hosptf = 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 hospital 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, mfc=col=c(4,4))
```

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)` to a data frame with factors for these variables and variables `LOR` and `ASE` containing the estimated log odds ratio ($\hat{\vartheta}$) and its asymptotic standard error ($ASE(\hat{\vartheta})$). Figure 11.43 shows the plot of these values as line graphs with associated 95% error bars produced using `ggplot2`.

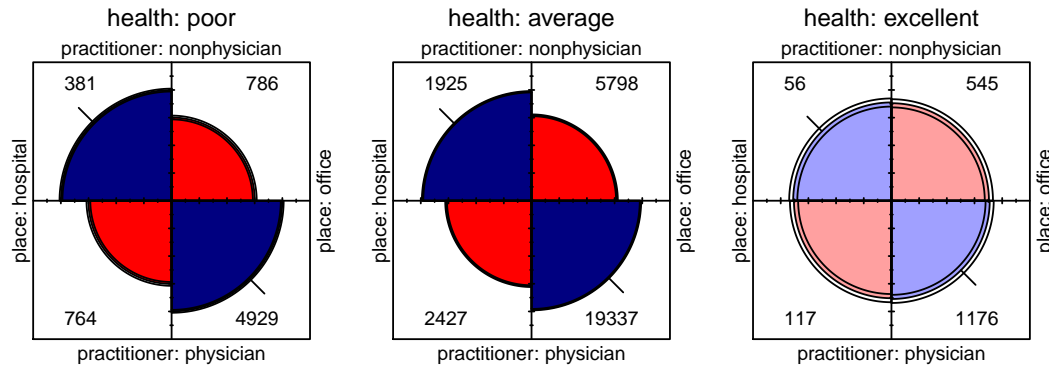


Figure 11.41: Fourfold displays for the association between practitioner and place in the `nmes.long` data, conditioned on health status.

{fig:nmes4-fourfold1}

```
> 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) +
+   facet_grid(. ~ gender, labeller=label_both) +
+   labs(x="Number of chronic conditions",
+        y="log odds ratio (physician|place)") +
+   theme_bw()
```

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, $H_0 : \vartheta_{ijk\dots} = 0$, the $\hat{\vartheta}$ are each distributed normally, $\mathcal{N}(0, \text{ASE}(\hat{\vartheta}))$, 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)
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

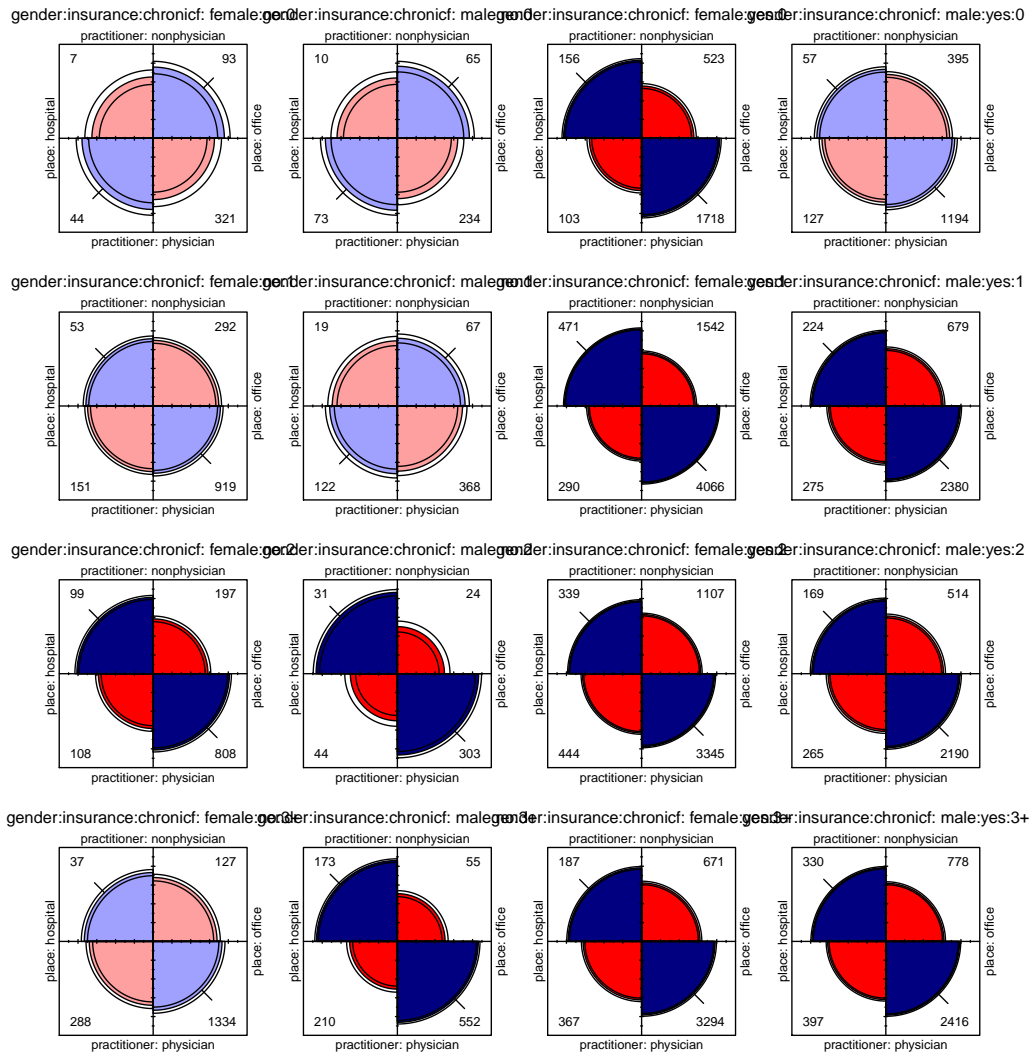


Figure 11.42: Fourfold displays for the association between practitioner and place in the nmes.long data, conditioned on gender, insurance and number of chronic conditions. Rows are levels of chronic; columns are the combinations of gender and insurance.

{fig:nmes4-fourfold2}

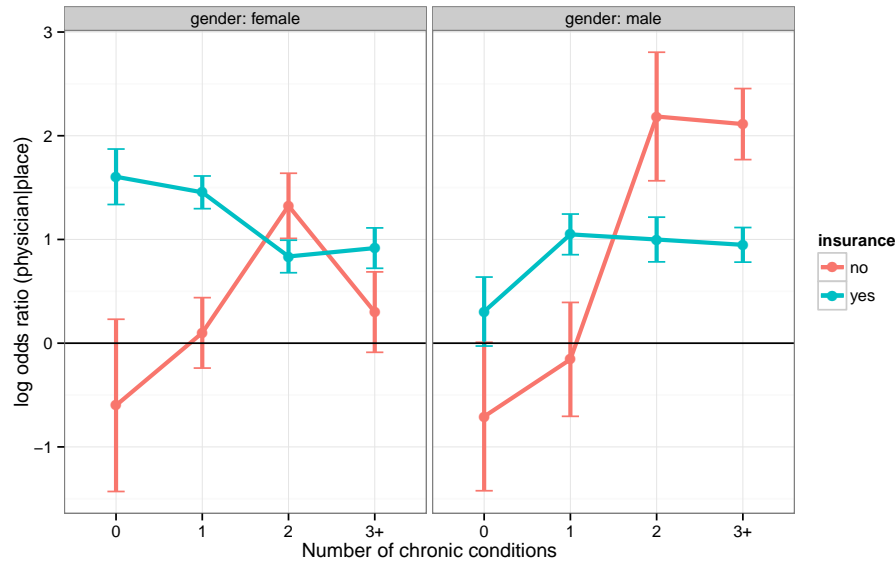


Figure 11.43: Plot of log odds ratios with standard error bars for the association between practitioner and place, conditioned on gender, insurance and number of chronic conditions.

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.

△

11.7.0.3 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 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,

$$\eta(\mathbf{x}) = \sum_{k=1}^p \mathbf{H}_k \beta_k \mathbf{x}_k$$

where $\mathbf{H}_1, \dots, \mathbf{H}_p$ are *known* full-rank constraint matrices. With no constraints the \mathbf{H}_k are identity matrices \mathbf{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 $\mathbf{H}_k = \mathbf{1}_q$, 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 non-linear 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 \mathbf{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.

```
> nmes2.nbin <- vglm(cbind(visits, nvisits, ovisits, novisits) ~ .,
+                   data = nmes2, family = negbinomial)
```

The estimated parameters from this model are returned by the `coef()` method as pairs of columns labeled `log(mu)`, `logsize` for each response. For example, the parameters for the visits response are in the first two columns, and are the same as those estimated for the model `nmes.nbin` using `glm.nb()`.

```
> # coefficients for visits
> coef(nmes2.nbin, matrix=TRUE)[, c(1, 2)]
```

	loge(mu1)	loge(size1)
(Intercept)	0.929257	0.18781
hospital	0.217772	0.00000
healthpoor	0.305013	0.00000
healthexcellent	-0.341807	0.00000
chronic	0.174916	0.00000
gendermale	-0.126488	0.00000
school	0.026815	0.00000
insuranceeyes	0.224402	0.00000

```
> # theta for visits
> exp(coef(nmes2.nbin, matrix=TRUE)[1, 2])
```

```
[1] 1.2066
```

The `log(mu)` coefficients for all four response variables are shown below.

```
> coef(nmes2.nbin, matrix=TRUE)[, c(1, 3, 5, 7)]
```

	loge(mu1)	loge(mu2)	loge(mu3)	loge(mu4)
(Intercept)	0.929257	-0.747798	-1.11284	-1.341783
hospital	0.217772	0.144645	0.41506	0.483881
healthpoor	0.305013	-0.179822	0.16491	0.033495
healthexcellent	-0.341807	-0.038121	-0.42449	-1.006528
chronic	0.174916	0.093430	0.27664	0.243572
gendermale	-0.126488	-0.255508	0.33456	0.052051
school	0.026815	0.068269	0.03559	-0.027476
insuranceeyes	0.224402	0.492793	-0.53105	0.484061

We notice that the coefficients for `hospital` and `chronic` have values with the same signs for all four responses. If it is desired to test the hypothesis that their coefficients are all the same for each of these predictors, first extract the H matrices for the unconstrained model using `constraints()`.

```
> clist <- constraints(nmes2.nbin, type = "term")
> clist$hospital[c(1, 3, 5, 7),]
```

	[,1]	[,2]	[,3]	[,4]
[1,]	1	0	0	0
[2,]	0	1	0	0
[3,]	0	0	1	0
[4,]	0	0	0	1

Then, reset the constraints for these terms to be unit vectors, forcing them to be all equal.

```
> clist2 <- clist
> clist2$hospital <- cbind(rowSums(clist$hospital))
> clist2$chronic <- cbind(rowSums(clist$chronic))
> clist2$hospital[c(1,3,5,7), 1, drop=FALSE]

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

Now, fit the same model as before, but using the constraints in `clist2`.

```
> nmes2.nbin2 <- vglm(cbind(visits, nvisits, ovisits, novisits) ~ ., data = nmes2,
+                     constraints = clist2,
+                     family = negbinomial(zero = NULL))
```

The coefficients for the constrained model are shown below. As you can see, the coefficients for `hospital` and `chronic` have the same estimates for all four responses.

```
> coef(nmes2.nbin2, matrix=TRUE)[,c(1,3,5,7)]

              loge(mu1) loge(mu2) loge(mu3) loge(mu4)
(Intercept)    0.918003 -0.835090 -0.864250 -1.175689
hospital        0.244655  0.244655  0.244655  0.244655
healthpoor      0.293335 -0.315479  0.366406  0.172434
healthexcellent -0.334959  0.047829 -0.538294 -1.044777
chronic          0.178562  0.178562  0.178562  0.178562
gendermale     -0.127956 -0.272264  0.330587  0.071212
school          0.026829  0.064401  0.031764 -0.028982
insuranceyes    0.222531  0.477475 -0.529976  0.507133
```

A likelihood-ratio test prefers the reduced model with equal coefficients for these two predictors. The degrees of freedom for this test (6) is the number of constrained parameters in the smaller model.

```
> lrtest(nmes2.nbin, nmes2.nbin2)

Likelihood ratio test

Model 1: cbind(visits, nvisits, ovisits, novisits) ~ .
Model 2: cbind(visits, nvisits, ovisits, novisits) ~ .
      #Df LogLik Df Chisq Pr(>Chisq)
1 35212 -25394
2 35218 -25413   6  39.2    6.4e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Alternatively, these tests can be performed as tests of linear hypotheses (see Section 11.1.2) on the coefficients B from the original model without refitting. Using `linearHypothesis()`, a hypothesis matrix L specifying equality of the coefficients for a given predictor can be easily generated using a character vector of the coefficient names.

```
> lh <- paste("hospital:", 1:3, " = ", "hospital:", 2:4, sep="")
> lh

[1] "hospital:1 = hospital:2" "hospital:2 = hospital:3"
[3] "hospital:3 = hospital:4"
```

Using `lh` as the `linear.hypothesis` argument then gives the following result for the coefficients of `hospital`, rejecting the hypothesis that they are all equal across response variables.

```
> car::linearHypothesis(nmes2.nbin, lh)

Linear hypothesis test

Hypothesis:
hospital:1 - hospital:2 = 0
hospital:2 - hospital:3 = 0
hospital:3 - hospital:4 = 0

Model 1: restricted model
Model 2: cbind(visits, nvisits, ovisits, novisits) ~ .

    Res.Df Df  Chisq Pr(>Chisq)
1    35215
2    35212  3   26.4    7.8e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

△

To pursue this analysis further, you could investigate whether any interactions of these effects were interesting and important as in Example 11.14, but now for the multivariate response variables.

To interpret a given model visually, you could use effect plots for the terms predicting each of the responses, as in Example 11.15. The `effects` package cannot handle models fit with `VGAM` directly, but you can use `glm()` or `glm.nb()` to fit the equivalent submodels for each response separately, and then use the `plot(Effect())` methods to display the effects for interesting terms. Figure 11.44 shows one such plot, for the effects of health status on each of the four response variables.

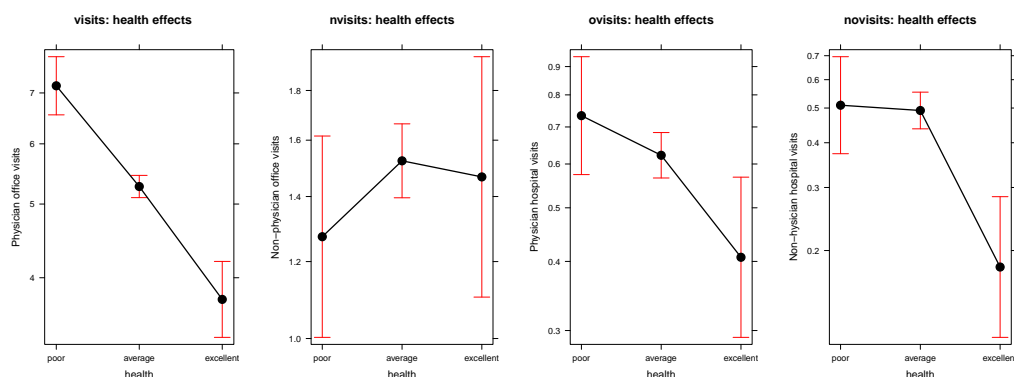


Figure 11.44: Effect plots for the effects of health status on the four response variables in the `nmes2` data.

{fig:nmes-eff-health}

11.8 Chapter summary

- The generalized linear model extends the familiar classical linear models for regression and ANOVA to encompass models for discrete responses and continuous responses for which the assumption of normality of errors is untenable.

- It does this by retaining the idea of a *linear predictor*— a linear function of the regressors, $\eta_i = \mathbf{x}_i^T \boldsymbol{\beta}$, but then allowing:
 - 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 $y_i | \mathbf{x}_i$ 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, $\mathcal{V}(y_i | \eta_i) = \mu_i$.
 - 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 $\mathcal{V}(y_i | \eta_i) = \mu_i + \alpha \mu_i^2$.
- 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 $\hat{\eta}^* = \mathbf{X}^* \hat{\boldsymbol{\beta}}$, using a score matrix \mathbf{X}^* 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

{lab:11.1}

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* 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 ...
```

- 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.
- Repeat (a) above, but now plotting $\log(\text{Matings}+1)$ against *Age* to approximate a Poisson regression with a log link and avoid problems with the zero counts.
- 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 $\text{Age}-27$ will make the intercept directly interpretable.)
- Check for non-linearity in the relationship by using the term $\text{poly}(\text{Age}, 2)$ in a new model. What do you conclude?
- Assess whether there is any evidence of overdispersion in these data by fitting analogous quasi-Poisson and negative-binomial models.

{lab:11.2}

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.

- 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?
- Re-fit this model as a quasi-Poisson model. Is there evidence of overdispersion? Test for overdispersion formally, using `dispersiontest()` from *AER*.
- Carry out the same significance tests and explain why the results differ from those for the Poisson model.

{lab:11.3}

Exercise 11.3 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
24	Pre	1	C4	no	mid	few	healthy	3
90	Pre	2	B1	no	mid	few	dead	2
103	Pre	2	B2	partial	low	few	dead	1
110	Pre	2	B2	full	high	few	healthy	5
126	Pre	3	C3	no	high	moderate	dead	2
132	Pre	3	B1	no	mid	few	healthy	3
173	Incubation	4	C4	no	high	few	healthy	1
181	Incubation	4	B1	no	high	few	healthy	7
185	Incubation	4	B1	partial	high	few	dead	1
234	Incubation	5	B2	no	high	few	healthy	12

- 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.
- Fit a main-effects Poisson GLM to these data and test the terms using `Anova()` from the `car` package.
- Interpret this model using an effects plot.
- 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.
- Test this model for overdispersion, using either a `quasipoisson` family or `dispersiontest()` in AER.

{lab:11.4}

Exercise 11.4 For the *CodParasites* data, recode the `area` variable as an ordered factor as suggested in footnote 10. Test the hypotheses that prevalence and intensity of cod parasites is linearly related to area.

{lab:11.5}

Exercise 11.5 In Example 11.10, we ignored other potential predictors in the *CodParasites* data: `depth`, `weight`, `length`, `sex`, `stage`, and `age`. Use some of the graphical methods shown in this case study to assess whether any of these are related to prevalence and intensity.

{lab:11.6}

Exercise 11.6 The analysis of the *PhdPubs* data in the examples in this chapter were purposely left incomplete, going only as far as the negative binomial model.

- Fit the zero-inflated and hurdle models to this data set, considering whether the count component should be Poisson or negative-binomial, and whether the zero model should use all predictors or only a subset. Describe your conclusions from this analysis in a few sentences.
- Using the methods illustrated in this chapter, create some graphs summarizing the predicted counts and probabilities of zero counts for one of these models.
- For your chosen model, use some of the diagnostic plots of residuals and other measures shown in Section 11.6 to determine if your model solves any of the problems noted in Example 11.17 and Example 11.18, and whether there are any problems that remain.

{lab:11.7}

Exercise 11.7 In Example 11.19 we used a simple analysis of $\log(\mathbf{y} + 1)$ for the multivariate responses in the *NMES1988* data using a classical MLM (Eqn. (11.16)) as a rough approximation of a multivariate Poisson model. The HE plot in Figure 11.40 was given as a visual summary, but did not show the data. Examine why the MLM is not appropriate statistically for these data, as follows:

- (a) Calculate residuals for the model `nmes.mlm` using

```
> resids <- residuals(nmes.mlm, type="deviance")
```

- (b) Make univariate density plots of these residuals to show their univariate distributions. These should be approximately normal under the MLM. What do you conclude?
- (c) Make some bivariate plots of these residuals. Under the MLM, each should be bivariate normal with elliptical contours and linear regressions. Add 2D density contours (`kde2d()`, or `geom_density2d()` in `ggplot2`) and some smoothed curve. What do you conclude?

```
> # detach(package:ggttern) ## detach any masking packages
> .locals$ch09 <- setdiff(ls(), .globals)
> #.locals$ch09
> remove(list=.locals$ch09[sapply(.locals$ch09, function(n) {!is.function(get(n))})])
```


References

- Agresti, A. (2013). *Categorical Data Analysis*. Wiley Series in Probability and Statistics. New York: Wiley-Interscience [John Wiley & Sons], 3rd edn.
- Akaike, H. (1973). Information theory and an extension of the maximum likelihood principal. In B. N. Petrov and F. Czaki, eds., *Proceedings of the 2nd International Symposium on Information*. Budapest: Akademiai Kiado.
- Atkinson, A. C. (1981). Two graphical displays for outlying and influential observations in regression. *Biometrika*, 68, 13–20.
- Atkinson, A. C. (1987). *Plots, Transformations and Regression: An Introduction to Graphical Methods of Diagnostic Regression Analysis*. New York: Oxford University Press.
- Box, G. E. P. (1979). Some problems of statistics and everyday life. *Journal of the American Statistical Association*, 74(365), 1–4.
- Box, G. E. P. and Cox, D. R. (1964). An analysis of transformations (with discussion). *Journal of the Royal Statistical Society, Series B*, 26, 211–252.
- Brockmann, H. J. (1996). Satellite male groups in horseshoe crabs, *Limulus polyphemus*. *Ethology*, 102(1), 1–21.
- Cameron, A. C. and Trivedi, P. K. (1990). Regression-based tests for overdispersion in the poisson model. *Journal of Econometrics*, 46, 347–364.
- Cameron, A. C. and Trivedi, P. K. (1998). *Regression analysis of count data*. Econometric society monographs. Cambridge (U.K.), New York: Cambridge University Press.
- Cameron, A. C. and Trivedi, P. K. (2013). *Regression analysis of count data*. Econometric society monographs. Cambridge (U.K.), New York: Cambridge University Press, 2nd edn.
- Cragg, J. G. (1971). Some statistical models for limited dependent variables with application to the demand for durable goods. *Econometrica*, 39, 829–844.
- Deb, P. and Trivedi, P. K. (1997). Demand for medical care by the elderly: A finite mixture approach. *Journal of Applied Econometrics*, 12, 313–336.
- Fox, J. (1987). Effect displays for generalized linear models. In C. C. Clogg, ed., *Sociological Methodology, 1987*, (pp. 347–361). San Francisco: Jossey-Bass.
- Fox, J. (2008). *Applied Regression Analysis and Generalized Linear Models*. Thousand Oaks, CA: Sage, 2nd edn.
- Fox, J. and Andersen, R. (2006). Effect displays for multinomial and proportional-odds logit models. *Sociological Methodology*, 36, 225–255.

- Fox, J., Friendly, M., and Monette, G. (2009). Visualizing hypothesis tests in multivariate linear models: The *heplots* package for R. *Computational Statistics*, 24(2), 233–246. (Published online: 15 May 2008).
- Friendly, M. (2007). HE plots for multivariate general linear models. *Journal of Computational and Graphical Statistics*, 16(2), 421–444.
- Friendly, M., Monette, G., and Fox, J. (2013). Elliptical insights: Understanding statistical methods through elliptical geometry. *Statistical Science*, 28(1), 1–39.
- Hastie, T. J. and Tibshirani, R. J. (1990). *Generalized Additive Models*. London: Chapman & Hall.
- Hemmingsen, W., Jansen, P. A., and Mackenzie, K. (2005). Crabs, leeches and trypanosomes: an unholy trinity? *Marine Pollution Bulletin*, 50(3), 336–339.
- Hilbe, J. M. (2014). *Modeling Count Data*. New York, NY: Cambridge University Press.
- Hofmann, H. and Theus, M. (2005). Interactive graphics for visualizing conditional distributions. Unpublished manuscript.
- Hummel, J. (1996). Linked bar charts: Analyzing categorical data graphically. *Computational Statistics*, 11, 23–33.
- King, G. (1989). A seemingly unrelated Poisson regression model. *Sociological Methods and Research*, 17(3), 235–255.
- Kleiber, C. and Zeileis, A. (2014). Visualizing count data regressions using rootograms. Working papers, Faculty of Economics and Statistics, University of Innsbruck.
- Lambert, D. (1992). Zero-inflated poisson regression, with an application to defects in manufacturing. *Technometrics*, 34, 1–14.
- Long, J. S. (1997). *Regression Models for Categorical and Limited Dependent Variables*. Thousand Oaks, CA: Sage Publications.
- Mc Rae, M. (2015). *Spatial, Habitat and Frequency Changes in Double-crested Cormorant Advertising Display in a Tree-nesting Colony*. Masters project, environmental studies, York University.
- McCulloch, C. E. and Neuhaus, J. M. (2005). Generalized linear mixed models. In *Encyclopedia of Biostatistics*. John Wiley & Sons, Ltd.
- Mullahy, J. (1986). Specification and testing of some modified count data models. *Journal of Econometrics*, 33, 341–365.
- Nelder, J. A. and Wedderburn, R. W. M. (1972). Generalized linear models. *Journal of the Royal Statistical Society, Series A*, 135, 370–384.
- Poole, J. H. (1989). Mate guarding, reproductive success and female choice in African elephants. *Animal Behavior*, 37, 842–849.
- Ramsey, F. L. and Schafer, D. W. (2002). *The Statistical Sleuth: A Course in Methods of Data Analysis*. Belmont, CA: Duxbury, 2nd edn.
- Ramsey, F. L., Schafer, D. W., Sifneos, J., and Turlach, B. A. (2012). *Sleuth2: Data sets from Ramsey and Schafer's "Statistical Sleuth (2nd ed)"*. R package version 1.0-7.
- Schwartz, G. (1978). Estimating the dimensions of a model. *Annals of Statistics*, 6, 461–464.

- Tukey, J. W. (1962). The future of data analysis. *Annals of Mathematical Statistics*, 33, 1–67 and 81.
- Vuong, Q. H. (1989). Likelihood ratio tests for model selection and non-nested hypotheses. *Econometrica*, 57(2), pp. 307–333.
- Williams, D. A. (1987). Generalized linear model diagnostics using the deviance and single case deletions. *Applied Statistics*, 36, 181–191.
- Wood, S. N. (2004). Stable and efficient multiple smoothing parameter estimation for generalized additive models. *Journal of the American Statistical Association*, 99(467), 673–686.
- Wood, S. N. (2006). *Generalized Additive Models: An Introduction with R*. Chapman and Hall/CRC Press.
- Yip, K. C. and Yau, K. K. (2005). On modeling claim frequency data in general insurance with extra zeros. *Insurance: Mathematics and Economics*, 36(2), 153–163.
- Zeileis, A., Kleiber, C., and Jackman, S. (2008). Regression models for count data in R. *Journal of Statistical Software*, 27(8).