4 GLMM for strictly positive data: biomass of rainforest trees

Alain F Zuur, Christopher D Philipson, Andy Hector, Elena N Ieno, Joseph M Hilbe

4.1 Rainforest tree species

Explaining the coexistence of large numbers of rainforest tree species is a major challenge in tropical ecology. A possible coexistence mechanism is the partitioning of the forest light environment; species that grow well under a given light regime may not thrive in another. To test this hypothesis, Philipson et al. (2012) investigated how tree species respond to environmental light. They estimated growth rates of 21 tropical lowland rainforest tree species from Malaysian Borneo grown in shade houses for 2 years under three light regimes. Based on monitoring of height, diameter, and aboveground biomass, they analyzed growth over time, using non-linear models, and calculated the growth rate of each species.

In July 2004, seedlings were placed in shade houses and left to acclimate for 6 weeks before the first measurements were taken. To assess initial size for each seedling, Philipson et al. (2012) counted the number of leaves, the diameter at the base of the stem, and the height to the apex of the stem. These measurements were repeated on days 65, 220, 442, and 643.

Accurate estimates of initial size are important to the fit of the non-linear models. To estimate the initial aboveground biomass of each seedling and the aboveground biomass on subsequent monitoring dates, Philipson et al. (2012) established regression relationships for each of the species. The following linear model was fitted for each mass fraction (leaves and woody biomass) of each species:

$$\log(mass) = \beta_1 + \beta_2 \times \log(diameter) + \beta_3 \times \log(height) + \beta_4 \times Number\ of\ leaves$$

The results of the linear regression models were used in analyses to estimate biomass growth curves for individual plants, which could only be harvested once. In this chapter we use a subset of the data measured at the start of the experiment to illustrate the issues encountered when multiple linear regression models with normal distribution are used to analyze

strictly positive data such as biomass. We apply gamma GLM and Tobit models as a solution.

We have biomass data of 290 seedlings of 21 species. No treatment has yet been applied, and there is no temporal or spatial dependency in the biomass data. The underlying task is to model biomass as a function of diameter, height, and the number of leaves. We apply the model on the data of all species, and investigate whether the relationships are consistent among species. This means that we need to include interactions. From a biological point of view, we expect that the biomass/diameter relationship changes per species.

Required knowledge for this chapter

- Chapter 1 and 2 of this book.
- The GLMM component of Chapter 3 of this book.
- Data exploration; see for example Zuur et al. (2010).
- Multiple linear regression with R. See for example Chapter 1 in Zuur (2012b). You have free access to this chapter.
- Bayesian statistics and MCMC. See for example Chapter 1 in Zuur et al. (2012a). You have free access to this chapter.

A model for this data containing an interaction term adds 20 parameters for each interaction term, and this is a large number. Are we really interested in obtaining the relationship for each species, or can we obtain a relationship for all species, and allow for random variation in the relationship for each species? This question drives us to decide whether the covariate *species* should be used as a fixed term or as a random term. In the latter case we can apply a mixed effects model in which *species* is used as a random intercept and include random slopes. The added benefit of this method is that atypical plants will have less effect on the regression for their species, as each species borrows strength from the entire dataset. This shrinkage is particularly useful when some of the species consist of fewer data points.

It should be noted that the response variable is *biomass*, which is always greater than 0. We need to ensure that the models do not predict negative biomass values. The statistical techniques that we discuss are the gamma generalized linear (mixed effects) model and Tobit model.

Table 4.1 presents a list of the available variables.

If the response variable is strictly positive, ensure that the fitted values are positive.

Table 4.1. Variables in the seedlings dataset.

Variable	Description	Type
Biomass	Biomass of stem and leaves	Continuous response (>0)
Diameter	Diameter at base of stem	Continuous covariate
Height	Height to apex of stem	Continuous covariate
Number of leaves	Number of leaves	Continuous covariate
Species	Identity of species	Categorical covariate

4.2 Importing the data and housekeeping

The following R code imports the data:

```
> Seedlings <- read.table(file = "Seedlings.txt",
                          header = TRUE)
```

The results of the `str(Seedlings)` command are not presented here, but show that the variable *species* is imported as a categorical variable, and all other variables are numeric or integers. The `names` function prints the variable names in the `Seedlings` data frame.

```
> names(Seedlings)
```

```
 [1] "Genus"         "Species"        "Identifier"
 [4] "Year"          "crown.mm"       "diam.1"
 [7] "diam.2"        "leaves"         "stem.biomass"
[10] "leafy.biomass" "leaf.area.1"    "leaf.area.2"
```

First we rename some of the variables and load packages and functions that will be used in the analysis. We combine the two mass fractions (stem and leaves), and the resulting variable, *biomass*, is the response variable. In cross section, the seedling stems are often oval rather than circular. To reduce error, the diameters were measured twice and the average was calculated. The variable `crown.mm` is the height, but we prefer to use unambiguous variable names with a capital, hence we rename it `Height`. Obviously we can do all these manipulations in Excel, but we prefer to do it in R, as it minimizes the modifications made to the original data file and demonstrates coding the variables for other data.

```
> Seedlings$Biomass  <- Seedlings$stem.biomass +
                        Seedlings$leafy.biomass
> Seedlings$Diameter <- (Seedlings$diam.1 +
                        Seedlings$diam.2) / 2
> Seedlings$Height   <- Seedlings$crown.mm
> Seedlings$Leaves   <- Seedlings$leaves
```

Next we load the packages and support functions. The `lattice` package is used for data exploration, and later we will apply Markov Chain Monte Carlo (MCMC) techniques; hence the `R2jags` package and our own support file, which can be downloaded from the book website.

```
> library(lattice)  #Needed for data exploration
> library(mgcv)     #Needed for data exploration
> library(R2jags)   #MCMC
> source(file = "HighstatLibV4.R") #Data explor.
> source(file = "MCMCSupportHighstat.R")  #MCMC
```

4.3 Data exploration

Data exploration is applied following the protocol described in Zuur et al. (2010). We start with outliers, followed by collinearity, and finally present relationships between the response variable *biomass* and the covariates.

4.3.1 Outliers

Cleveland dotplots of the response variable and the three continuous covariates are presented in Figure 4.1. None of the variables has any obvious outliers, so we do not apply transformations.

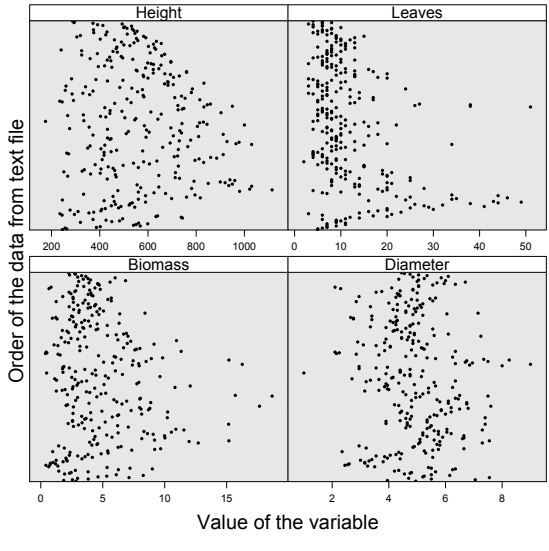

Figure 4.1. Cleveland dotplots of the response variable Biomass and the continues covariates.

R code to make Figure 4.1 follows. We create a character vector, `MyVar`, with the names of the variables, and the `Mydotplot` function is a set of commands written as a wrapper around the `dotplot` function from the `lattice` package. It is in the `HighstatLibV4.R` file that we sourced in Section 4.2.

```
> MyVar <- c("Biomass", "Diameter", "Height",
             "Leaves")
> Mydotplot(Seedlings[, MyVar])
```

We also need to know whether any of the species has a considerably lower or higher number of observations, and for this we use the `table` function.

```
> table(Seedlings$Species)
```

```
argentifolia     beccariana       beccarii      conformis
          10             10             20             10
  faguetiana        gibbosa          guiso     johorensis
          10             10             20             10
  lanceolata      leprosula    macrophylla     macroptera
          10             10             10             10
  malaanonan        nervosa         oleosa         ovalis
          10             40             20             10
  parvifolia parvistipulata         sangal        superba
          10             20             10             20
  tomentella
          10
```

We have 10, 20, or 40 observations per species. If one species had only 1 or 2 observations, and another species 40, it would be better to remove the data from the species with 1 or 2 observations, but that is not necessary here.

4.3.2 Collinearity

Next we look at collinearity among the covariates. Figure 4.2 shows a pairplot for the continuous covariates. A Pearson correlation of 0.5 means moderate collinearity between *diameter* and *height*. We can decide to drop one of the variables, but we can also keep both. Collinearity reduces *p*-values in regression-type models. If the relationship between the response variable and the covariates is weak, a moderate amount of collinearity means that none of the covariates may be significant. On the other hand, if the relationships between the response variable and covariates are strong, the model can cope with a moderate, or even high, amount of collinearity., and the results of the model are still valid. We will see that the relationship between *biomass* and some of the covariates is strong, and therefore we decide to keep all continuous covariates in the model. Another way to assess collinearity is through variance inflation factors (VIF) (Zuur et al. 2010). All VIFs are smaller than 3, which is further confirmation that all three covariates can be used in the models.

If strong relationships exist between the response variable and covariates, we can allow a certain degree of collinearity. If relationships are weak, we must eliminate all collinearity.

R code to make Figure 4.2 is elementary. We again use a wrapper (`Mypairs`) around an existing function (`pairs`).

```
> MyVar <- c("Diameter", "Height", "Leaves")
> Mypairs(Seedlings[, MyVar])
```

To assess collinearity between the categorical variable *species* and continuous covariates we make boxplots of each continuous covariate conditional on *species*. These graphs are not presented here, but there is minimal to moderate collinearity between *species* and, for example, *diameter*.

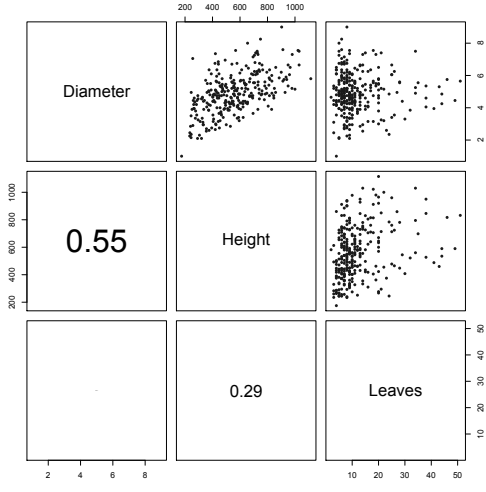

Figure 4.2. Pairplot of the continuous covariates in the seedlings data. The lower diagonal panels show Pearson correlations.

4.3.3 Relationships

Figure 4.3 shows multiple scatterplots of the response variable `Biomass` versus each of the continuous covariates. There are clear patterns in the data, which is good if we expect the model to show significant effects. The relationship between *biomass* and *diameter* seems to be exponential.

The graph was made with the following R code. The function `Myxyplot` is a wrapper around the `xyplot` function from the `lattice` package.

```
> Myxyplot(Seedlings, MyVar, "Biomass")
```

Another interesting graph is the boxplot of *biomass* conditional on *species* (Figure 4.4). Note that *biomass* seems to differ among species. We wrote a wrapper function around the `bwplot` to make this graph:

```
> Mybwplot(Seedlings, c("Biomass"), "Species")
```

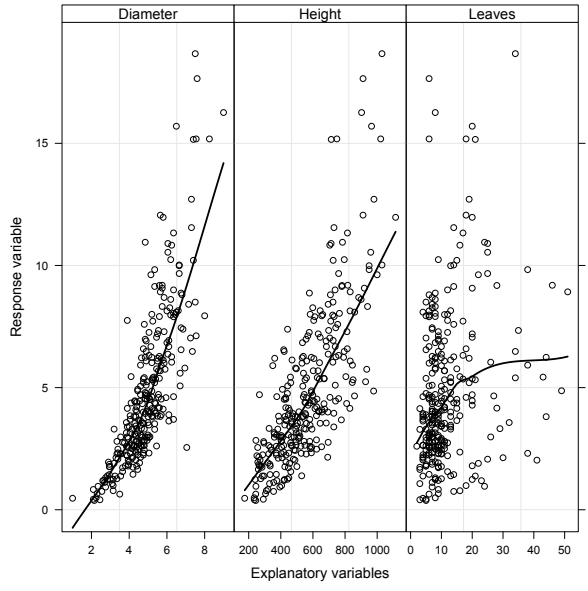

Figure 4.3. Scatterplot of biomass versus each of the continuous variables. A LOESS smoother is added to aid visual interpretation.

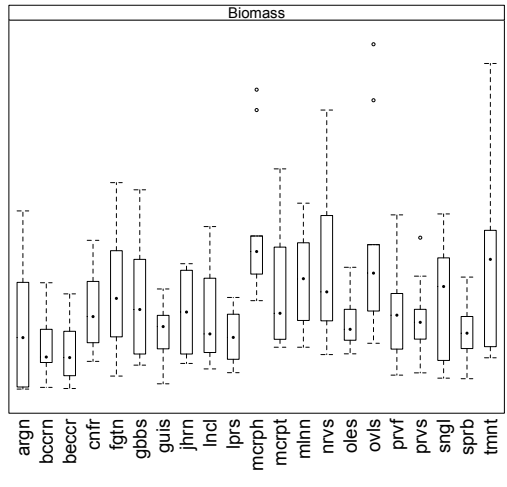

Figure 4.4. Boxplot of biomass conditional on species. Species names along the *x*-axis are abbreviations.

One of the underlying questions was whether the relationship between *biomass* and *diameter* differs per species. The multi-panel scatterplot in Figure 4.5 is a useful graph to explore whether this is the case. It shows

the relationship between *biomass* and *diameter* for each species. There are 21 species, and, if we include an interaction term between *species* and *diameter* in the model, this term consumes 20 degrees of freedom, which is a high number. The more the slopes of the lines in Figure 4.5 differ from one another, the more likely it is that the interaction is important. Instead of using 20 degrees of freedom, we can apply a generalized linear mixed model (GLMM) with a random intercept (*species*) and slope (*diameter*). We discuss and apply such models later in this chapter. As to the slopes of the lines in Figure 4.5, they do not show extensive differences.

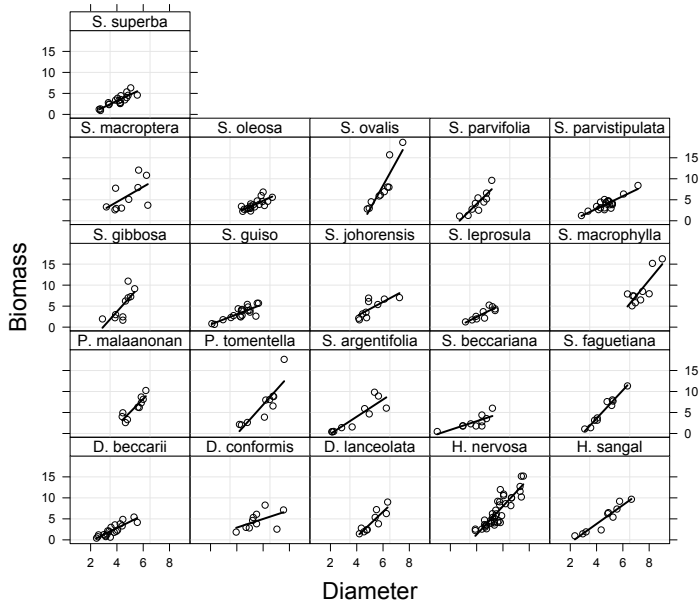

Figure 4.5. Scatterplot of *biomass* versus *diameter* for each species. A linear regression line is added to aid visual interpretation.

R code to make this figure is extensive and is presented below. We created a new categorical variable `Gen_Sp` that contains the first initial of the genus of a species along with the species name.

```
> Seedlings$Gen_Sp <- factor(paste(
             substr(Seedlings$Genus, 1, 1),
             Seedlings$Species, sep = ". "))
```

The first two lines in the `xyplot` code is the only part that needs to be changed for other data. The rest of the code determines the colour of the strips above each panel, the scales (try changing `same` to `free` and see what happens), and the *x*- and *y*-labels (which may also be changed

according to the data presented). The `panel` function draws the points and regression lines.

```
> xyplot(Biomass ~ Diameter | Gen_Sp,
       data = Seedlings,
       strip = function(bg = 'white', ...)
               strip.default(bg = 'white', ...),
       scales = list(alternating = T,
                     x = list(relation = "same"),
                     y = list(relation = "same")),
       xlab = list(label = "Diameter", cex = 1.5),
       ylab = list(label = "Biomass", cex = 1.5),
       panel = function(x,y){
            panel.grid(h = -1, v = 2)
            panel.points(x, y, col = 1,pch = 1)
            tmp <- lm(y ~ x)
            MyData <- data.frame(x = seq(min(x),
                                         max(x),
                                         length = 25))
            p1 <- predict(tmp, newdata = MyData)
            panel.lines(MyData$x, p1, col = 1,
                        lwd = 2)})
```

4.4 Multiple linear regression: a frequentist approach

To model the relationship between *biomass* and *diameter*, *height*, *number of leaves*, and *species* we apply the following multiple linear regression model:

$$\begin{aligned} &Biomass_i = Intercept + Species_i \times Diameter_i + Height_i + Leaves_i + \varepsilon_i \\ &\varepsilon_i \sim N(0_i, \sigma^2) \end{aligned} \tag{4.1}$$

To simplify mathematical notation we dropped the regression parameters preceding the covariates. The model contains an interaction term between *species* and *diameter*. Note that the model can also be written in a GLM-type notation as:

$$\begin{aligned} &Biomass_i \sim N(\mu_i, \sigma^2) \\ &E\left(Biomass_i\right) = \mu_i \qquad \text{var}\left(Biomass_i\right) = \sigma^2 \\ &\mu_i = Intercept + Species_i \times Diameter_i + Height_i + Leaves_i \end{aligned}$$

Implementation of the multiple linear regression model in Equation (4.1) in R is as follows. The `lm` function is used to estimate the parameters.

```
> M1 <- lm(Biomass  ~ Species * Diameter +Height +
           Leaves, data = Seedlings)
> summary(M1)          #Results not presented here
> drop1(M1, test="F")  #Results not presented here
```

The results of the `summary` function are not presented here, as the *species* term and the *species* × *diameter* term use 40 lines in the output. The explained percent of variation, R^2, is 90%, which means that the model fit is good. The results of the `drop1` function are not presented here, but show that the *species* × *diameter* interaction is significant at the 5% level as are the two main terms, *height* and *leaves*.

Figure 4.6 shows a graph of residuals plotted versus fitted values. There are no clear non-linear patterns or outliers in the residuals, although there seems to be minor heterogeneity. However, there is a major problem with the results of the multiple linear regression model: some of the fitted values are negative, which does not make sense, as *biomass* is strictly positive. The negative fitted values are the points on the left side of the vertical dotted line.

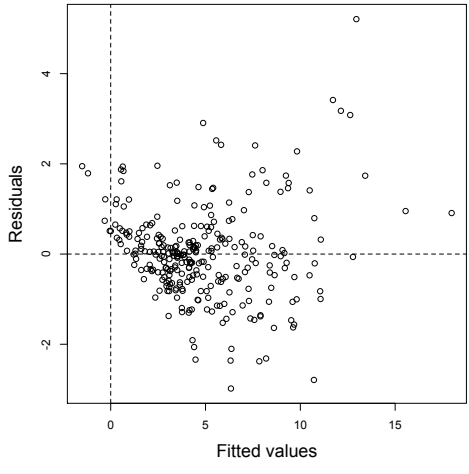

Figure 4.6. Plot of fitted values versus residuals for the multiple linear regression model.

R code to make Figure 4.6 follows. The `rstandard` function is used to extract standardized residuals (Zuur 2012).

```
> E1 <- rstandard(M1)
> F1 <- fitted(M1)
> plot(x = F1, y = E1, xlab = "Fitted values",
       ylab = "Residuals")
> abline(h = 0, v = 0, lty = 2)
```

If a model predicts negative values for a response variable that is strictly positive, it is time to take a different approach!

So what can we do about the negative fitted values? Some of the *biomass* values are smaller than 1, and if we apply a logarithmic transformation on *biomass*, such values become smaller than 0. We can

apply the following multiple linear regression on the logarithmic transformed biomass:

$$\log(Biomass_i) = Intercept + Species_i \times Diameter_i + Height_i + Leaves_i + \varepsilon_i$$
$$\varepsilon_i \sim N(0_i, \sigma^2)$$

and negative fitted values are no longer a concern. However, the transformation changes the relationships between the response variable and the covariates. Zuur (2012) presented an example in which a log transformation on a response variable resulted in scatterplots showing almost no interaction, while scatterplots for the original data showed clear interactions. The *p*-value of the *species*/*diameter* interaction term has 14 zeros, whereas for log transformed *biomass* it is 0.003. This is still significant, but it shows that applying a transformation makes the slopes of the lines in Figure 4.5 more similar.

A log-transformation changes the relationships.

There are other methods of avoiding negative fitted values. Two approaches are the Tobit model and the gamma generalized linear model (GLM). A Tobit model is similar to the multiple linear regression model, except that the normal distribution is adjusted so that the probability of negative realisations is eliminated. In a gamma GLM, a gamma distribution is used, which is for strictly positive data. Software routines for Tobit and gamma models are widely available in R, but when extensions are needed for the inclusion of random effects, we quickly turn to MCMC techniques. The gamma GLM is more frequently used compared to Tobit models to analyse strictly positive data, and we begin with a discussion of this approach. The Tobit model will be illustrated later.

When comparing results of multiple linear regression, gamma GLM, and Tobit models it is useful to present compact output. If we include the categorical covariate *species* and the interaction between *species* and *diameter*, we introduce 40 additional lines of numerical output into each model, making comparison of the three approaches cumbersome. Therefore we use a model with only the three continuous covariates: *diameter*, *height* and *number of leaves*. Once we understand the principle of gamma GLM and Tobit models and learn how to fit such models in R and read the output, we will extend the models with *species* information, first as a fixed term and then as a random intercept.

4.5 Gamma GLM using a frequentist approach

4.5.1. Formulating the gamma GLM

A gamma distribution can be used for a response variable that is continuous and positive. Negative values and zeros are not allowed with this distribution. A gamma distribution is useful when the response variable is *biomass*, *length*, *density*, etc. A variable that is gamma distributed with mean μ and parameter ν has variance μ^2 / ν, allowing for a quadratic variance structure.

For the seedling data we fit the following gamma GLM:

$$\begin{aligned} &Biomass_i \sim Gamma(\mu_i, \tau) \\ &E(Biomass_i) = \mu_i \qquad \text{and} \qquad \text{var}(Y_i) = \frac{\mu_i^2}{\tau} \\ &\log(\mu_i) = \beta_1 + \beta_2 \times Diameter_i + \beta_3 \times Height_i + \beta_4 \times Leaves_i \end{aligned} \tag{4.2}$$

The response variable $Biomass_i$ for seedling i is assumed to follow a gamma distribution with mean μ_i, and μ_i is modelled in the same way as in a Poisson or negative binomial GLM (see Chapter 2). The assumption of a gamma distribution determines the variance; notice that it is quadratic. So what is a gamma distribution? What does it look like, what is its density function, and how many parameters does it have? We address these questions in the next two subsections.

4.5.2 Scale and shape

The general notation for a gamma distribution and its density function in R is as follows (see the help file of `dgamma`):

$$\begin{aligned} &Y_i \sim Gamma(\text{scale}_i, \text{shape}) \\ &E(Y_i) = \text{shape} \times \text{scale}_i \qquad \text{and} \qquad \text{var}(Y_i) = \text{shape} \times \text{scale}_i^2 \\ &f(Y_i; Y_i > 0) = \frac{1}{\text{scale}_i^{\text{shape}} \times \Gamma(\text{shape})} \times Y_i^{\text{shape}-1} \times e^{-\frac{Y_i}{\text{scale}_i}} \end{aligned} \tag{4.3}$$

A gamma distribution has two parameters, the scale and the shape. The expression in Equation (4.2) is a mathematical expression, and, as you can see from Equation (4.3), the implementation in the software can differ.

In Equation (4.3) we have included an index, i, in the scale parameter, but not in the shape parameter, to emphasize that R models the scale parameter as a function of the covariates. The confusing aspect of Equation (4.2) and (4.3) is that the R function `glm` with the gamma distribution presents the expected values μ_i if we use the `predict` function. So, in principle, the scale and shape parameters are not a concern. It is only when we do the gamma GLM in JAGS that we have to know what is what.

4.5.3 Visualizing the gamma distribution

Figure 4.7A shows a simulated relationship between *biomass* and a covariate. The fact that the covariate is between 0 and 1, and that the relationship is exponential, is irrelevant. The range of *biomass* in this graph is between 0 and 20 (excluding 0). Panel B shows 6 gamma density curves. In all these curves, the mean of the gamma distribution is 5. This means that we can visualize these 6 curves superimposed above the line in panel A at the *biomass* value 5. The horizontal range in panel B indicates the probability of other *biomass* values. The density curve with the highest peak in panel B has shape = 15. The other 5 curves have shape values of 2, 3, 5, and 10, and the odd line that goes up when *biomass* reaches small values has a shape value of 1. The density curve with the highest peak indicates that if the mean *biomass* is 5, *biomass* values are likely to fall between 3 and 8. Obviously, higher values are possible but have small probability.

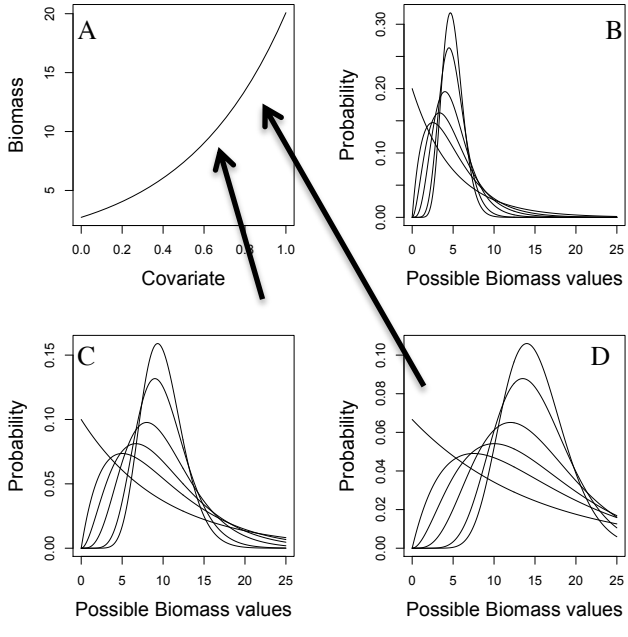

Figure 4.7. A: Simulated relationship between *biomass* and a covariate. B: Gamma density curves when the mean μ = 5. Different values of scale are used. C: As panel B with μ = 10. D: As panel B with μ = 15.

In panel C, the mean *biomass* is 10, and we use the same shape values. Note that the smaller the scale, the more asymmetric the density curve

becomes. We need to imagine the density curve in panel C superimposed on the curve in panel A at *biomass* 10, as indicated by the arrow. In panel D, the mean of the gamma distribution is 15, and the same shape values are used. Note that the spread of the density curves increases, which makes sense as the variance of a gamma distribution is quadratic.

In Figure 4.8 we plotted the gamma density curves with scale = 15 on the curve in Figure 4.7A. It clearly shows that for larger fitted values the *biomass* spread increases.

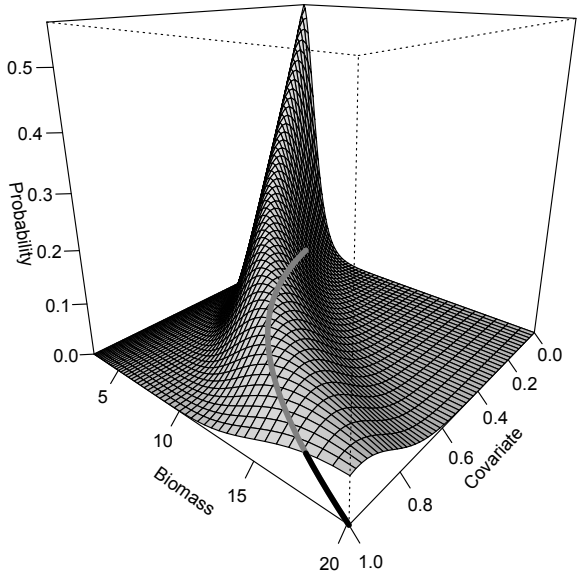

Figure 4.8. Three-dimensional presentation of the gamma distribution for a gamma GLM with log link. Gamma density curves are superimposed on the simulated curve in Figure 4.7A.

4.5.4 Different link functions

Let us start simply and apply the gamma GLM using the `glm` function in R. In Equation (4.2) we use a log link function to model the expected values of *biomass* as a function of the covariates. The default (canonical) link is the inverse link function. A third option available in the `glm` function is the identity link. This means that we have the following three link functions available:

$$\begin{aligned}\log(\mu_i) &= \beta_1 + \beta_2 \times Diameter_i + \beta_3 \times Height_i + \beta_4 \times Leaves_i \\ 1/\mu_i &= \beta_1 + \beta_2 \times Diameter_i + \beta_3 \times Height_i + \beta_4 \times Leaves_i \\ \mu_i &= \beta_1 + \beta_2 \times Diameter_i + \beta_3 \times Height_i + \beta_4 \times Leaves_i\end{aligned}$$

The choice between these three link functions depends on the functional relationship between the expected values of *biomass* and the covariates. Choosing the wrong link function may result in non-convergence of the mathematical algorithm, negative fitted values, patterns in the residuals, and further modelling misery. Our choice for the log link function is based on the observation in Figure 4.3 that the relationships are potentially exponential. We may try the identity link if the scatterplots show clear linear patterns. The Akaike Information Criterion (AIC) can be used to compare models with different link functions. Alternatively, we can use residual plots to choose a link function.

4.5.5 Running the Gamma GLM using the glm function

To fit the gamma GLM with log link with the `glm` function we use

```
> M2 <- glm(Biomass  ~ Diameter + Height +
            Leaves, data = Seedlings,
            family = Gamma(link = "log") )
```

The output of the GLM with gamma distribution and log link is:

```
> print(summary(M2), signif.stars = FALSE,
       digits = 2)

             Estimate Std. Error t value Pr(>|t|)
(Intercept) -1.220945   0.069072  -17.68  < 2e-16
Diameter     0.353905   0.015724   22.51  < 2e-16
Height       0.001362   0.000106   12.81  < 2e-16
Leaves       0.009523   0.001956    4.87  1.9e-06

(Dispersion parameter for Gamma family taken to be
0.07424697)

    Null deviance: 126.546  on 289 df
Residual deviance:  20.481  on 286 df
AIC: 851.7
```

All regression parameters are significant at the 5% level. We can also use the `drop1` function, but we need to specify the *F*-test.

4.5.6 Scale confusion

Previously, when working with a Poisson GLM, our first step was to look at over- or underdispersion. However, the concept of overdispersion as in Poisson GLM or negative binomial GLM does not apply to a gamma GLM, as this is a 2-parameter distribution, similar to a normal distribution.

Note that when using R's `glm` function for a gamma regression, the shape parameter is not directly estimated. The output does present a value for the shape parameter but it is calculated in a rudimentary way using Pearson residuals. This is because the GLM contains a single-parameter estimation algorithm. By this we mean that for a Poisson, binomial, normal, and inverse Gaussian distribution, the algorithm is formulated in a way does not allow for a shape parameter from a gamma distribution (or the parameter k from a negative binomial distribution). Technically, the gamma GLM and the negative binomial GLM are not GLMs unless the scale is entered into the model as a constant. However, for the gamma model, we do not need the shape parameter to estimate the regression parameters (the betas in the predictor function). Hence we can still use the `glm` function to estimate the betas.

To calculate the shape parameter, R first calculates the dispersion parameter as the sum of squared Pearson residuals, divided by the sample size, minus the number of parameters. This means we need Pearson residuals first. These are given by

$$E_i = \theta^{1/2} \times \frac{\left(Biomass_i - E\left(Biomass_i\right)\right)}{\sqrt{\text{var}\left(Biomass_i\right)}}$$

The θ enters the equation in the same way as for the quasi-Poisson model discussed in Chapter 1. At this point the `glm` function becomes simplified and sets the shape equal to 1 / θ. Recall that the variance of *biomass* equals μ_i^2 / shape; use equation (4.3) to obtain this expression. This means that the θ and shape parameter both are eliminated from the formula, giving

$$E_i = \frac{\left(Biomass_i - \mu_i\right)}{\sqrt{\mu_i^2}}$$

The code `variance = mu^2` is contained within the `Gamma` function in R. Strictly speaking this is incorrect, as the variance contains the shape parameter, but, if the dispersion θ is set to the reciprocal of the shape, it does not matter. However, when we later run JAGS, we need to be able to properly estimate the shape parameter, so are better adhering to the correct equations for the Pearson residuals.

To confirm that we understand the `glm` function, we attempt to reproduce its output. First we calculate a matrix, **X,** that contains the intercept and covariates. In the next section we will explain **X** in more detail. Using the `coef` function we extract the estimated regression parameters, calculate `eta`, and calculate the fitted values, `mu`. Next we calculate the Pearson residuals and dispersion as above. Results are the same as the dispersion parameter reported by the `summary` function.

```
> X <- model.matrix(~ Diameter + Height + Leaves,
```

```
                    data = Seedlings)
> eta <- X %*% coef(M2)
> mu  <- exp(eta)
> EPearson  <- (Seedlings$Biomass - mu) / mu
> N <- nrow(Seedlings)
> p <- length(coef(M2))
> dispersion <- sum(EPearson^2) / ( N - p)
> dispersion
0.07424697
> summary(M2)$dispersion
0.07424697
```

The dispersion parameter is then used to calculate the shape parameter: shape = 1 / dispersion = 13.468. Summarizing, if we set the shape parameter equal to the reciprocal of the dispersion, both terms disappear from the Pearson residuals, and also from the equations for the variance, standard errors, and regression parameters. So, we don't need the shape parameter at all. But, what if setting the shape to the reciprocal of the dispersion produces an estimator that is not so good? What if we estimate the shape parameter and the regression parameters simultaneously using maximum likelihood? Hilbe (2011) showed that if the shape parameter is estimated using a maximum likelihood function that estimates both the shape and scale parameters, setting the shape value of the `glm` function to the value obtained in the shape and scale parameter model results in standard errors that are identical in the models. Not estimating the shape typically results in standard errors that differ little from those of a jointly estimated scale and shape model.

Unfortunately there is no way to provide a more precise estimation of the shape using the `glm` function than that provided by the software. Results of a two-parameter log-gamma model are required. We can obtain these values using JAGS, as we illustrate in the following section.

This may seem hopelessly complicated, but, as mentioned earlier, there is no immediate need to be familiar with all this information, as the shape and scale parameters play out their roles in the background. However, as we will later apply MCMC, it is better to be familiar with the information in the black box. If, for some reason, calculation of the scale parameter is required, use:

```
> Scale <- fitted(M2, type = response) / 13.46856
```

Here is a reason that we may want to calculate the scale. The ratio of the variance and the expected values of *biomass* are equal to the scale:

$$\frac{\text{var}\left(Biomass_i\right)}{E\left(Biomass_i\right)} = \frac{shape \times scale_i^2}{shape \times scale_i} = scale_i$$

In Figure 4.9 we plot the scale versus the fitted values. The resulting graph indicates that the variance is smaller than the expected values ($scale_i < 1$) up to the fitted value of approximately 15. For larger fitted values (> 15) the variance is larger than the fitted values ($scale_i > 1$). R code for Figure 4.9 follows. The fitted values are obtained with the `fitted` function, and the scale follows its definition.

```
> F2 <- fitted(M2, type = "response")
> Scale <- F2 / 13.46856
> plot(x = F2, y = Scale, xlab = "Fitted values",
       ylab = "Scale")
```

There is another way to demonstrate how the model uses the gamma distribution. We sketch fitted values versus *diameter*, for average values of *height* and *number of leaves* (Figure 4.10). The solid points are the fitted values and the line is the model fit. We should be careful when comparing the observed *biomass* data with these fitted values, as we used average *height* and average *number of leaves* values, and not all observations have such covariate conditions. Superimposed on the graph are 1,000 possible realizations from a gamma distribution, with scale and shape parameters taken from the fitted model, for 25 *diameter* values. Note that the variation increases for larger fitted values. R code to produce Figure 4.10 is extensive and is presented on the book website.

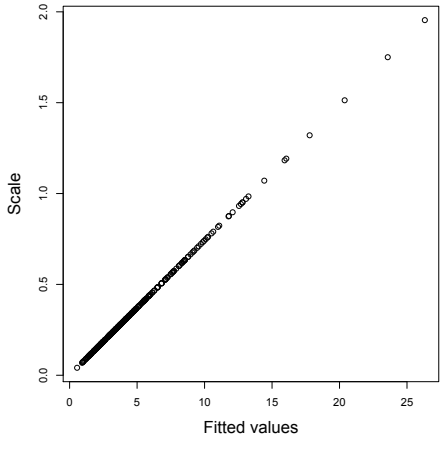

Figure 4.9. Fitted values plotted versus the scale parameters obtained by the gamma GLM with log link function. The scale parameter is equal to the ratio of the variance and expected values of *biomass*.

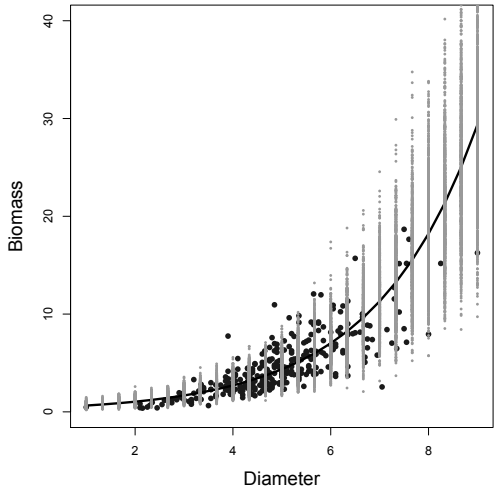

Figure 4.10. Fitted values (thick line) of the gamma GLM with log link function plotted versus *diameter*. The tick dots are the observed data. Values of *height* and *number of leaves* values are averages. The grey points represent 1,000 possible realizations from a gamma distribution at each of the selected 25 *diameter* values.

4.5.7 Identity link and inverse link function

In the previous subsections we fitted a gamma GLM with a log link to model the mean of *biomass*. The gamma distribution will ensure positive fitted values, irrespective of the link function, so the choice between a log link and an identity link should be based on the functional relationship between *biomass* and the covariates. We were curious to know how other link functions perform and tried a gamma GLM with identity link. This can be fitted with the following R code:

```
> M2B <- glm(Biomass  ~ Diameter + Height +
             Leaves, data = Seedlings,
             family = Gamma(link = "identity"),
             start = c(0, 0.5, 0, 0))
```

For this specific dataset, the algorithm has problems fitting a gamma GLM with identity link, most likely because the algorithm is unable to handle negative fitted values. The functional relationship between *biomass* and *diameter* seems to be exponential, and a straight line is likely to give negative fitted values. The `glm` function gives various warning messages, and, without starting values for the regression parameters, it even crashes. The model fit is given in Figure 4.11. It shows that choosing the correct link function is important, as the model fit with the identity link function is poor.

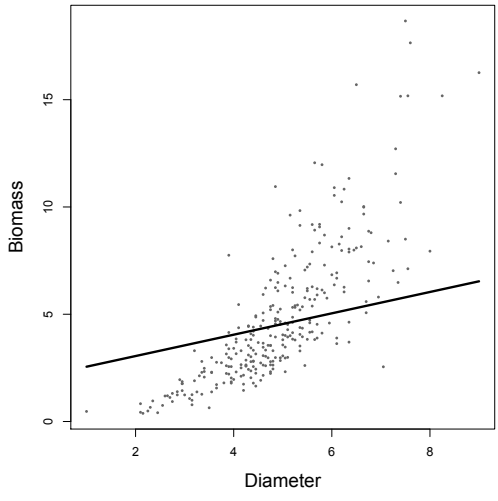

Figure 4.11. Model fit of the gamma GLM with identity link. Values for *height* and *number of leaves* are averages.

The canonical inverse link is not used, since we want to maintain a direct relationship between the linear predictor and fitted values, as allowed by the log link. It is interesting to note, however, that the gamma GLM with log link function in which the shape is set to 1 becomes a GLM with exponential distribution. Exponential regression is a single parameter model. The log-gamma model differs only in the standard errors, due to the internal scaling by the Pearson dispersion.

4.6 Fitting a Gamma GLM using JAGS

In this section we fit the gamma GLM with log link function in JAGS. We use the continuous covariates *diameter*, *height,* and *number of leaves*, but we formulate the JAGS modelling code in such a way that adding covariates (e.g. *species* and *species* × *diameter* interaction) requires minimal modification of the code.

4.6.1 Specifying the data for JAGS

In Chapter 3 we showed that centring covariates improves mixing of the chains. We therefore centre the continuous covariates in this instance, but, during initial MCMC runs, we encounter poor mixing, especially when the categorical variable *species* is included. When we standardize the covariates (centre each covariate and divide by the standard deviation) we obtain perfect mixing. The poor mixing using centred variables may have resulted from our choice of initial values for the chains of the regression parameters. Centring the covariates and making improvements in the starting values of the chains is an option, but we forego this and standardize the three continuous covariates for use in the remainder of this discussion. R code to standardize follows:

```
> Seedlings$DiameterS <- scale(Seedlings$Diameter)
> Seedlings$HeightS   <- scale(Seedlings$Height)
> Seedlings$LeavesS   <- scale(Seedlings$Leaves)
```

The `scale` function centres the variable and divides the centred data by the standard deviation. To confirm that our JAGS code is correct, we compare the JAGS results with those obtained by the `glm` function. If the results are similar (they will never be identical, as JAGS employs simulation), we can be confident that our JAGS code is correct and later extend it to gamma GLMM. We run the `glm` function again with the standardized variables and present its output as a reference point:

```
> M3 <- glm(Biomass  ~ DiameterS + HeightS +
            LeavesS, data = Seedlings,
            family = Gamma(link = "log") )
> print(summary(M3), digits = 3)

            Estimate Std. Error t value Pr(>|t|)
(Intercept)   1.3870     0.0160   86.68  < 2e-16
DiameterS     0.4383     0.0195   22.51  < 2e-16
HeightS       0.2608     0.0204   12.81  < 2e-16
LeavesS       0.0829     0.0170    4.87  1.9e-06

(Dispersion parameter for Gamma family taken to be
0.07424697)

    Null deviance: 126.546 on 289 df
Residual deviance:  20.481 on 286 df
AIC: 851.7
```

The remainder of this section focuses on producing these results with JAGS. Recall from Equation (4.2) that we used the log link function with the following form.

$$\begin{aligned} \log(\mu_i) &= \eta_i \\ \eta_i &= \beta_1 + \beta_2 \times Diameter_i + \beta_3 \times Height_i + \beta_4 \times Leaves_i \end{aligned}$$

We can write the predictor function η in matrix notation as

$$\begin{pmatrix} \eta_1 \\ \vdots \\ \eta_N \end{pmatrix} = \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix} \times \beta_1 + \begin{pmatrix} Diameter_1 \\ \vdots \\ Diameter_N \end{pmatrix} \times \beta_2 + \begin{pmatrix} Height_1 \\ \vdots \\ Height_N \end{pmatrix} \times \beta_3 + \begin{pmatrix} Leaves_1 \\ \vdots \\ Leaves_N \end{pmatrix} \times \beta_4$$

$$= \begin{pmatrix} 1 & Diameter_1 & Height_1 & Leaves_1 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & Diameter_N & Height_N & Leaves_N \end{pmatrix} \times \begin{pmatrix} \beta_1 \\ \beta_2 \\ \beta_3 \\ \beta_4 \end{pmatrix}$$

$$= \mathbf{X} \times \boldsymbol{\beta}$$

The matrix **X** is of dimension 290 by 4, where 290 is the number of observations (*N*). The vector **β** contains the intercept β_1 and the slopes β_2, β_3, and β_4. We can easily get the matrix **X** with the `model.matrix` function in R:

```
> X <- model.matrix(~ 1 + DiameterS + HeightS +
                      LeavesS, data = Seedlings)
```

The matrix **X** is of dimension 290 by 4 (use `dim(X)` in R to verify). Its first column contains only the values 1, and the second column consists of the standardized *diameter* data. Columns 3 and 4 contain the standardized *height* and *number of leaves* values. Entering `head(X)` will reveal that **X** indeed contains this information. We convert **X** to a matrix using

```
> X <- as.matrix(X)
```

To make our JAGS code as general as possible, we define the variable *K* as the number of columns in **X**.

```
> K <- ncol(X)
```

We create an object, `win.data`, that contains the *biomass* data, the matrix **X**, and the sample size *N*. The `b0` and `B0` are used for the priors of the regression parameters in **β** and will be explained.

```
> win.data <- list(Biomass = Seedlings$Biomass,
                   X       = X,
                   N       = nrow(Seedlings),
                   b0      = rep(0, K),
                   B0      = diag(0.0001, K))
```

4.6.2 JAGS modeling code

We store the JAGS modelling code in a file designated `SeedlingsGLM.txt`. It consists of the following:

```
sink("SeedlingsGLM.txt")
cat("
model{
  #1. Priors for regression coefficients
  beta ~ dmnorm(b0[], B0[,])
  r    ~ dgamma(0.01, 0.01)

  #2. Likelihood
  for (i in 1:N){
     Biomass[i] ~ dgamma(r, lambda[i])
     lambda[i]  <- r / mu[i]
     log(mu[i]) <- max(-20, min(20, eta[i]))
     eta[i]     <- inprod(beta[],X[i,])
  }
```

```
}
",fill = TRUE)
sink()
```

4.6.3 Priors

The following diffuse priors are used for the regression parameters:

$$\beta_i \sim N(0,100^2)$$

In matrix notation for all 4 regression parameters, this is:

$$\begin{pmatrix} \beta_1 \\ \beta_2 \\ \beta_3 \\ \beta_4 \end{pmatrix} \sim N\left(\begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 100^2 & 0 & 0 & 0 \\ 0 & 100^2 & 0 & 0 \\ 0 & 0 & 100^2 & 0 \\ 0 & 0 & 0 & 100^2 \end{pmatrix}\right)$$

In JAGS we can formulate this as

```
for (i in 1:K) { beta[i] ~ dnorm(0, 0.0001) }
```

Recall from Chapter 3 that the second argument in the function `dnorm` in JAGS is precision, which is defined as 1 / variance. This explains the 0.0001; it is $1 / 100^2$. In Chapter 3 we also explained that mixing improves if, instead of sampling each `beta[i]` from a normal distribution, we sample all K betas at once from a multivariate normal distribution. The JAGS command

```
beta ~ dmnorm(b0[], B0[,])
```

implements these priors. The vector `b0` contains K zeros, and `B0` is a K-by-K diagonal matrix with 0.0001 on its diagonal. Finally, a gamma prior of mean 1 is used for the `r` parameter of the gamma distribution.

4.6.4 Likelihood function

The `inprod` function is used for the predictor function `eta`, giving

$$eta_i = \beta_1 + \beta_2 \times Diameter_i + \beta_3 \times Height_i + \beta_4 \times Leaves_i$$

The log link is used, resulting in:

$$\mu_i = \mathrm{e}^{eta_i} = e^{\beta_1+\beta_2\times Diameter_i+\beta_3\times Height_i+\beta_4\times Leaves_i}$$

Next we discuss the gamma distribution for *Biomass*. The JAGS manual states that we need `dgamma(r, λ)`, and the matching density function is

$$Y_i \sim dgamma(\mathrm{r}, \lambda_i)$$

$$f(Y_i; Y_i > 0) = \frac{\lambda_i^r \times Y_i^{r-1}}{\Gamma(\mathrm{r})} \times e^{-\lambda_i \times Y_i}$$

Let us compare this to the `dgamma` function and the `glm` function in R. The r in JAGS is the shape parameter used by `glm`. The λ in JAGS is 1 / scale in `glm`. Therefore the expected values of *biomass* using JAGS are given by r × (1/ λ_i). The JAGS code for the gamma distribution is

```
Biomass[i] ~ dgamma(r, lambda[i])
lambda[i]  <- r / mu[i]
log(mu[i]) <- eta[i]
eta[i]     <- inprod(beta[],X[i,])
```

We dropped the `min` and `max` functions to increase the clarity of the code. The final two lines calculate μ_i. The expected value of *biomass* is given by:

$$E\left(Biomass_i\right) = r \times \frac{1}{\lambda_i} = r \times \frac{1}{r / \mu_i} = \mu_i$$

The variance is given by μ_i^2 / r. This demonstrates how JAGS parameterises a gamma distribution and how it differs from the `dgamma` used in R. Keep in mind that JAGS estimates the shape parameter. It will not use a rudimentary substitution as the `glm` function did.

4.6.5 Initial values and parameters to save

The following initial values are used, and we store the MCMC iterations from `beta` and `r`.

```
> inits <- function () {
   list(
    beta = rnorm(K, 0, 0.01),
    r    = 1 )  }
> params <- c("beta", "r")
```

4.6.6 Running JAGS from R

We execute the JAGS code from within R using the `jags` function from the `R2jags` package that was loaded in Section 4.2.

```
> load.module("glm")
> J0 <- jags(data       = win.data,
             inits      = inits,
             parameters = params,
             model.file = "SeedlingsGLM.txt",
             n.thin     = 10,
             n.chains   = 3,
             n.burnin   = 40000,
```

```
              n.iter      = 50000)
```

We use 3 chains with a burn-in of 40,000, thinning rate of 10, and 50,000 iterations. As a result, 3 × (50,000 – 40,000) / 10 = 3,000 observations are used for each posterior distribution.

4.6.7 JAGS results presented within R

Results are

```
> out <- J0$BUGSoutput
> out
```

```
Inference for Bugs model at "SeedlingsGLM.txt", fit using
jags, 3 chains, each with 50000 iterations (first 40000
discarded), n.thin = 10
 n.sims = 3000 iterations saved
          mean  sd  2.5%   25%   50%   75% 97.5% Rhat n.eff
beta[1]    1.4 0.0   1.4   1.4   1.4   1.4   1.4    1  2300
beta[2]    0.4 0.0   0.4   0.4   0.4   0.5   0.5    1   540
beta[3]    0.3 0.0   0.2   0.2   0.3   0.3   0.3    1  1300
beta[4]    0.1 0.0   0.1   0.1   0.1   0.1   0.1    1  3000
deviance 846.7 3.1 842.6 844.5 846.1 848.3 854.1    1   600
r         14.2 1.1  12.0  13.4  14.1  14.9  16.5    1  3000

For each parameter, n.eff is a crude measure of effective
sample size, and Rhat is the potential scale reduction factor
(at convergence, Rhat=1).

DIC info (using the rule, pD = var(deviance)/2)
pD = 4.6 and DIC = 851.4
DIC is an estimate of expected predictive error (lower
deviance is better).
```

The `Rhat` values are all close to 1, indicating good mixing, although we prefer to plot the chains and visually inspect the mixing. If there are many variables in the model, we use our own functions for presenting and post-processing the chains, as they present more condensed output from selected variables.

```
> OUT1 <- MyBUGSOutput(out, c(uNames("beta", K),
                              "r"))
> print(OUT1, digits = 3)
           mean     se    2.5%  97.5%
beta[1]  1.3876 0.0161  1.3579  1.419
beta[2]  0.4386 0.0199  0.3993  0.477
beta[3]  0.2608 0.0205  0.2217  0.301
beta[4]  0.0834 0.0169  0.0503  0.117
r       14.1514 1.1424 11.9647 16.453
```

The posterior mean of the iterations for the regression parameters are similar to those obtained by the `glm` function, as are the standard errors. The posterior mean of the scale parameter is also close to what we found with the `glm` function. The `glm` function provides a simple estimator for the scale parameter, as we explained earlier in this chapter. Venables and

Ripley (2002) provide a function to estimate the scale parameter using maximum likelihood:

```
> library(MASS)
> MyShape <- gamma.shape(M3)
> MyShape

Alpha: 14.324390
SE:     1.175985
```

The 14.324 obtained using the `gamma.shape` function from MASS is close to the 14.151 produced by JAGS.

To assess mixing, we plot the chains using the following R code:

```
> MyBUGSChains(out, c(uNames("beta", K), "r"))
```

The `uNames` function creates a character vector of the form `c("beta[1]", "beta[2]", "beta[3]", "beta[4]")`, and we add `r` to this vector before passing it on to the `MyBUGSChains` function. The resulting graph is presented in Figure 4.12, and mixing is good. If the model mixing is not satisfactory, we use

```
> J1.upd <- update(J0, n.iter = 50000, n.thin=10)
> out <- J1.upd$BUGSoutput
```

and continue updating until we have good mixing.

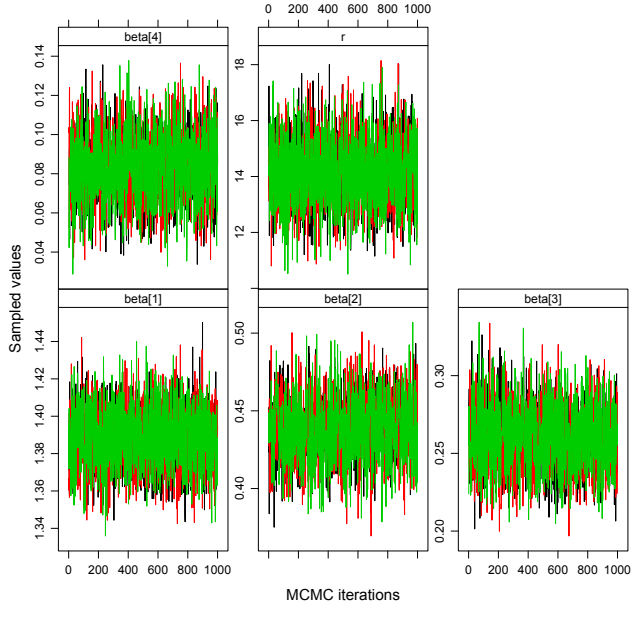

Figure 4.12. MCMC chains for all 4 regression parameters and r in the gamma GLM.

The first call to the `jags` function sets the `adapt` value, which is used for 'warming up the algorithm.' The `adapt` value is equal to the burn-in value, and, for more complicated models, it is worthwhile to use a large `adapt` value, which means specifying a large burn in. The definition of 'large' depends on sample size and model complexity.

Histograms for the posterior distributions are presented in Figure 4.13. The crucial point is whether 0 is within the range of the histogram. We can draw a vertical line at 0 or delineate a 95% credible interval at the base of each histogram. If 0 is inside the 95% credible interval, the parameter is not significant. We created Figure 4.13 using our own functions:

```
> MyHist(out, c(uNames("beta", K), "r"))
```

These histograms are easily produced once we have the 3,000 iterations. These can be obtained with

```
> AllPar <- out$sims.matrix[,c(uNames("beta",K),
                                "r")]
```

The object `AllPar` is of dimension 3,000 × 5 and contains the MCMC iterations for each parameter.

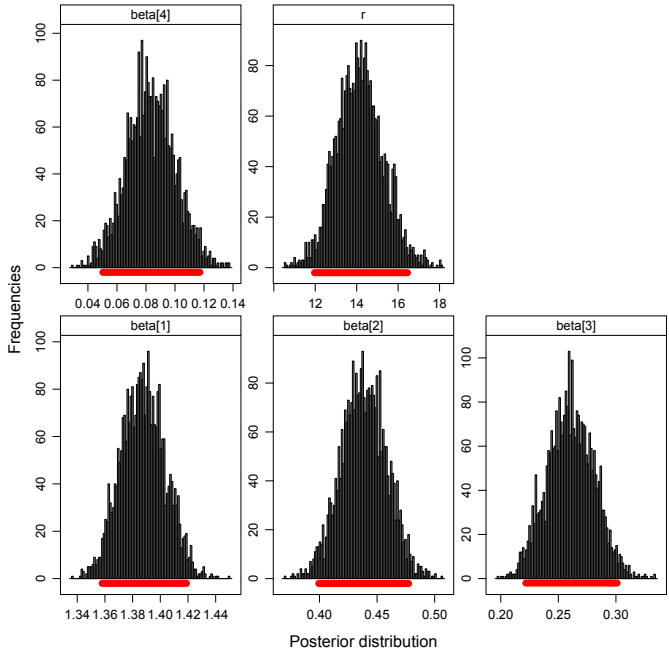

Figure 4.13. Histogram of the MCMC iterations for each parameter. The thick line at the base of the histogram is the 95% credible interval. Note that no 95% credible interval contains 0.

4.6.8 Model interpretation

Summarising, we have written a JAGS routine that calculates the MCMC chains, and the posterior means of these chains are similar to those obtained with the `glm` function. Next we address model validation and model interpretation. We should start with model validation, as there is no point in making nice graphs to illustrate the results in terms of biology, if the model validation later shows residual patterns, and another model must be fitted. Yet we ignore our own recommendations and dive into model interpretation, as it is the most interesting part of the analyses.

For model interpretation, we recommend always visualizing the results of the model. For example, we can plot the expected *biomass* values versus *diameter* for average *height* and *number of leaves*. There are different ways to calculate fitted values for specific covariate values. We can use the posterior means of the regression parameters and substitute these in the equations for the gamma GLM. This leads to

```
> Beta <- OUT1[1:4,1]
> r    <- OUT1[5,1]
> MyData <- data.frame(
     DiameterS = seq(min(Seedlings$DiameterS),
                     max(Seedlings$DiameterS),
                     length = 100),
     HeightS = 0,
     LeavesS = 0)
> X1 <- model.matrix(~ 1 + DiameterS + HeightS +
                     LeavesS, data = MyData)
> eta <- X1 %*% Beta
> mu  <- exp(eta)
```

The vector `Beta` contains the posterior means of the 4 regression parameters. We create a data frame `MyData` that contains 100 values for *diameter* (from the smallest to the largest observed standardized value), and *height* and *number of leaves* are set to 0. Recall that we are working with standardized variables; hence the value of 0 represents the average. The data frame `MyData` is converted to the matrix, **X1,** and we calculate the fitted values `mu`. We can plot `mu` versus `MyData$DiameterS` to visualize the fitted values.

A different approach is to use the 3,000 MCMC realisations for each parameter to calculate the fitted values 3,000 times at each of the 100 points. From these we can calculate a posterior mean and 95% credible intervals. We can do this inside JAGS, but there is no need to do so, as it increases computing time and digital storage. Here is how to do it. First we extract the 3,000 iterations for the betas:

```
> Beta3000 <- out$sims.matrix[, uNames("beta", K)]
> dim(Beta3000)
```

```
[1] 3000    4
```

Using the same covariate matrix, **X1,** we calculate the fitted values:

```
> eta <- X1 %*% t(Beta3000)
> mu  <- exp(eta)
> dim(mu)
```

```
100 3000
```

The object `mu` contains 3,000 iterations (fitted values) for each of the 100 diameter values, so each column of `mu` represents a line with fitted values. We can plot these 3,000 lines (Figure 4.14A). Alternatively, we can calculate and sketch a posterior mean and a 95% credible interval for each of the 100 points (Figure 4.14B). Note that these are the Bayesian equivalents of confidence intervals and not prediction intervals. R code to make Figure 4.14 is lengthy and is not presented here. It can be found on the book website.

To present a prediction interval, we follow the steps for creating Figure 4.10. In JAGS this is much easier than when using a frequentist approach. The simplest method is to pass `X1` on to JAGS and, inside the JAGS code (outside the existing loop), add

```
for (j in 1:100) {
  etaNew[j]     <- inprod(beta[],X1[j,])
  log(muNew[j]) <- etaNew[j]))
  lambdaNew[j]  <- r / muNew[j]
  YNew[j] ~  dgamma(r, lambdaNew[j])
}
```

This code samples new data for the covariate values specified in the matrix **X1** (not in **X**) from a gamma distribution. If we extract the MCMC iterations for `YNew`, its 95% credible intervals can be used to calculate prediction intervals. Obviously, you can also do this when JAGS has finished running, using the 3,000 iterations for the betas and r and drawing new *biomass* data from a gamma distribution using the R function `dgamma`.

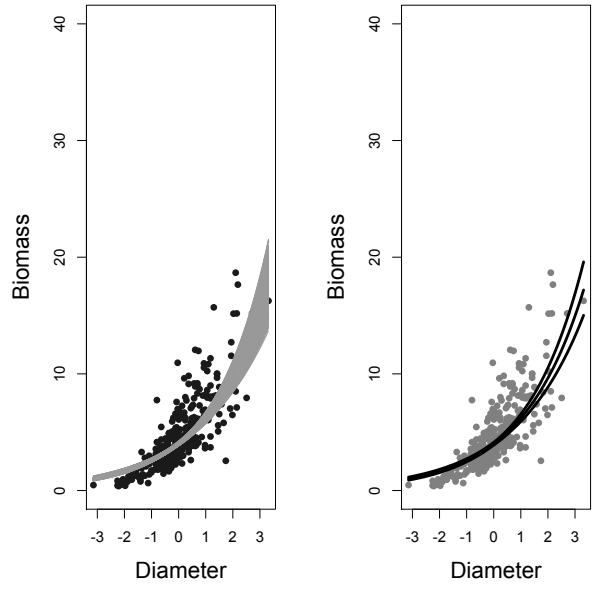

Figure 4.14. A: Observed *biomass* plotted versus (standardized) *diameter*. Three thousand lines representing fitted values for each MCMC iteration are added. B: Same as A, with the posterior mean and 2.5% and 97.5% quartiles.

4.6.9 Model validation

For model validation, we need to calculate Pearson residuals and plot them against fitted (expected) values and also versus each covariate in the model and each covariate not in the model.

A simple approach is to calculate posterior means of the regression parameters and the shape parameter. When we have these we can calculate Pearson residuals in the usual way.

Alternatively, we can calculate Pearson residuals inside the JAGS code and use posterior means of the Pearson residuals for each seedling. A variation on this approach is to use the 3,000 iterations for each regression parameter and calculate Pearson residuals when JAGS has finished running. We can then use the posterior means of the residuals. The advantage of this approach is that we save time in not having to calculate the residuals and store them inside JAGS.

To execute the latter approach we store the 3,000 iterations for the betas and r in the objects `Beta3000` and `r3000`. Because we have experimented with different design matrices X, we take X from `win.data` to be sure that we have the correct matrix. Fitted values and the variance are calculated using the underlying equations that were presented earlier in this chapter.

```
> Beta3000 <- out$sims.matrix[, uNames("beta", K)]
> r3000    <- out$sims.matrix[, "r"]
> eta      <- win.data$X %*% t(Beta3000)
> mu       <- exp(eta)
> ExpY     <- mu
> VarY     <- (1 / r3000) * mu^2
```

Now that we have the expected values and the variance, we can calculate Pearson residuals. Recall that these are given by:

$$E = \phi \times \frac{Biomass_i - E\left(Biomass_i\right)}{\sqrt{\text{var}\left(Biomass_i\right)}} = \phi \times \frac{Biomass_i - \mu_i}{\sqrt{\mu_i^2 / \tau}}$$

To avoid a loop, we create a matrix, `Biomass3000`. In each column we place a copy of the *biomass* data.

```
> Biomass3000 <- matrix(rep(win.data$Biomass,
                            3000),
                        nrow = 290)
```

Before we can calculate the Pearson residuals we need ϕ, which is defined as the reciprocal of the shape.

```
> phi3000 <- 1 / r3000
```

Now we have the tools required to calculate the Pearson residuals:

```
> E3000 <- sqrt(phi3000) * (Biomass3000 - mu)  /
                           sqrt(VarY)
```

Finally we take posterior means for the residuals and fitted values at each site:

```
> E   <- rowSums(E3000) / 3000
> Fit <- rowSums(mu) / 3000
```

We plot the Pearson residuals versus the fitted values, and we also make a boxplot of the residuals conditional on *species*. The boxplot shows that we need to include *species* as a covariate in the model. We do this in the next section.

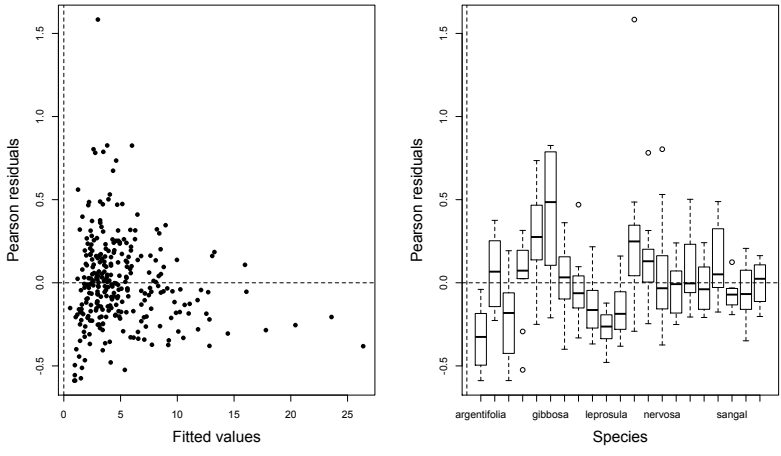

Figure 4.15. A: Posterior means of Pearson residuals plotted versus posterior means of fitted values. B: Posterior means of Pearson residuals plotted versus *species*. Note that space does not permit labelling of all species. To resolve this, we could rotate the axis labels and use abbreviations.

R code to make Figure 4.15 follows:

```
> plot(x = Fit, y = E,
       xlab = "Fitted values",
       ylab = "Pearson residuals", pch = 16)
> abline(h = 0, v = 0, lty = 2)
> boxplot(E ~ Species, data = Seedlings,
          xlab = "Species",
          ylab = "Pearson residuals")
> abline(h = 0, v = 0, lty = 2)
```

As part of the model validation process we also need to plot the Pearson residuals versus each covariate in the model (Figure 4.16). Any non-linear pattern means that we need to extend the model. In this case there are no clear non-linear patterns. R code to make the graph uses our wrapper function around the `xyplot` function from the `lattice` package.

```
> Seedlings$E <- E
> MyVar       <- c("DiameterS", "HeightS",
                   "LeavesS")
> Myxyplot(Seedlings, MyVar, "E")
```

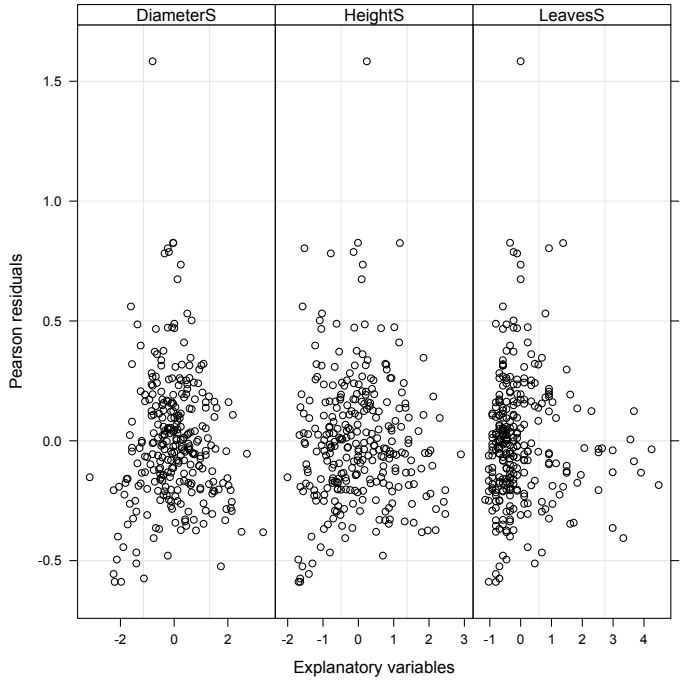

Figure 4.16. Scatterplots of posterior mean Pearson residuals versus each covariate.

4.7 Adding more covariates to the gamma GLM in JAGS

Results in the previous section indicated that the residuals contain species information and we therefore need to include the categorical covariate *species* and the interaction between *species* and *diameter* to the model. This is the model that we should have started with, as it conforms to the underlying question. We can consider *species* as a fixed term, in which case the number of parameters increases by 40, or we can treat *species* as a random term, leading to a gamma GLMM. In this section we treat *species* as a fixed term, and in the next section we show how to fit the gamma GLMM.

Adding more covariates and interactions is an elementary coding exercise. We only need to adjust the design matrix **X** to

```
> X <- model.matrix(~ 1 + Species * DiameterS +
                    HeightS + LeavesS,
                    data = Seedlings)
> K <- ncol(X)
```

No further changes are necessary. Computing time will increase, as this model has an additional 40 parameters. At this stage, mixing is poor when

using centred continuous covariates. Changing to standardized covariates improves mixing. No further output is presented here as it follows the same steps and code as presented in the previous section. The model no longer contains residual patterns.

4.8 Gamma GLMM

Everything presented up to this point can be executed with the `glm` function in R. Adding random effects means that we need R functions that can apply a gamma GLM with random effects. One option may be the `VGAM` package (Yee and Wild, 1996), but it is not an easy package to use. This is the point from which MCMC and JAGS become invaluable.

In the previous section *species* was treated as a fixed effect, but now we use it as a random effect. This reduces the number of parameters, as random effects can be considered residuals. However, from a pure Bayesian point of view, residuals are parameters. An additional benefit of GLMM is that it automatically imposes a correlation on the seedling biomass observations of the same species. Using MCMC it is possible to calculate this correlation (Ntzoufras, 2010).

The model that we select is a random intercept and slope model:

$$\begin{aligned} &Biomass_{ij} \sim Gamma(\mu_{ij}, \tau) \\ &\log(\mu_{ij}) = \beta_1 + \beta_2 \times Diameter_{ij} + \beta_3 \times Height_{ij} + \beta_4 \times Leaves_{ij} + \\ &\qquad a_i + b_i \times Diameter_{ij} \\ &a_i \sim N(0, \sigma_1^2) \\ &b_i \sim N(0, \sigma_2^2) \end{aligned} \tag{4.4}$$

We have changed the indices; $Biomass_{ij}$ is now the j^{th} observation of *species* i. We have 21 species; hence i = 1,.., 21. The number of observations per species ranges from 10 to 40. The term a_i is a random intercept and allows for random variation on the intercept β_1 and is assumed to be normally distributed with mean 0 and variance σ_1^2. The term b_i is a random slope and allows for random variation around the slope for *diameter*. It is also normally distributed, although with a different variance from that of the random intercept.

From a Bayesian point of view, this is just another set of covariates, so there are not many changes. We have seen in Chapter 3 how to include random intercepts. A random intercept and slope involves a simple extension of the code that we used in Chapter 3.

4.8.1 R code for a gamma GLMM in JAGS

First we define the matrix, **X**, that contains the covariates of the fixed part of the model, excluding any random effects.

```
> X <- model.matrix(~ 1 + DiameterS +
                      HeightS + LeavesS, data = Seedlings)
> K <- ncol(X)
```

We need to recode the categorical variable *species* as sequential integers beginning with 1.

```
> re <- as.numeric(Seedlings$Species)
> NumSpecies <- length(unique(Seedlings$Species))
```

The object `re` contains integer values indicating which observation belongs to a given species, and `NumSpecies` is the number of different species (21).

We put the response variable, covariates, object `re`, and tools for the priors (`b0`, `B0`, `a0`, `A0`) into a list called `win.data`.

```
> win.data <- list(Biomass  = Seedlings$Biomass,
                   X        = X,
                   N        = nrow(Seedlings),
                   b0       = rep(0, K),
                   B0       = diag(0.0001, K),
                   re       = re,
                   a0       = rep(0, NumSpecies),
                   A0       = diag(1, NumSpecies)
                  )
```

Next we show the JAGS code. It is a combination of the gamma GLM code presented in the previous section and that of the Poisson GLMM presented in Chapter 3. We have two sets of random effects: `a` and `b`. We use a multivariate normal prior for each of them. The square root of the variances of the random effects are given by `sigma.ri` and `sigma.rs` (ri indicates random intercept and rs random slope). We use Half-Cauchy(25) priors for them; see Chapter 3 for a detailed explanation. The predictor function `eta[i]` contains two additional terms not in the gamma GLM presented in Section 4.7: `a[re[i]]` and `b[re[i]] * X[i,2]`. The first is the random intercept, and the second is the random slope, and the second column of **X** contains the (standardized) *diameter*.

```
sink("SeedlingsGLMM.txt")
cat("
model{
  #1. Priors for regression coefficients
  beta     ~ dmnorm(b0[], B0[,])
  tau      ~ dgamma( 0.01, 0.01 )
  a        ~ dmnorm(a0[], tau.ri * A0[,])
  b        ~ dmnorm(a0[], tau.rs * A0[,])
  num.ri   ~ dnorm(0, 0.0016)
  num.rs   ~ dnorm(0, 0.0016)
  denom.ri ~ dnorm(0, 1)
  denom.rs ~ dnorm(0, 1)
  sigma.ri <- abs(num.ri / denom.ri)
  sigma.rs <- abs(num.rs / denom.rs)
  tau.ri   <- 1 / (sigma.ri * sigma.ri)
```

```
  tau.rs   <- 1 / (sigma.rs * sigma.rs)

  #2. Likelihood
  for (i in 1:N){
     Biomass[i] ~  dgamma(tau, mu.eff[i])
     mu.eff[i]  <- tau / mu[i]
     log(mu[i]) <- max(-20, min(20, eta[i]))
     eta[i]     <- inprod(beta[],X[i,])  +
                   a[re[i]] +
                   b[re[i]] * X[i,2]
  }
}
",fill = TRUE)
sink()
```

The initial values and parameters to save are given by

```
> inits <- function () {
   list(
    beta     = rnorm(K, 0, 0.01),
    tau      = 1,
    a        = rnorm(NumSpecies, 0, 0.1),
    b        = rnorm(NumSpecies, 0, 0.1),
    num.ri   = rnorm(1, 0, 25),
    num.rs   = rnorm(1, 0, 25),
    denom.ri = rnorm(1, 0, 1),
    denom.rs = rnorm(1, 0, 1))}
> params <- c("beta", "tau", "sigma.ri",
              "sigma.rs", "a", "b")
```

Again, this is merely a combination of code presented in Chapter 3 and in Section 4.7.

4.8.2 Results from JAGS for the gamma GLMM

The model is executed with the `jags` function. The thinning rate is 10, 3 chains are used, and we use the `update` function to get 15,000 iterations for the posterior distribution of each parameter.

```
> J0 <- jags(data       = win.data,
             inits      = inits,
             parameters = params,
             model.file = "SeedlingsGLMM.txt",
             n.thin     = 10,
             n.chains   = 3,
             n.burnin   = 40000,
             n.iter     = 50000)
> J1.upd <- update(J0, n.iter = 50000, n.thin =10)
> out    <- J1.upd$BUGSoutput
```

Mixing of the chains is good, and the numerical output is given below:

```
> OUT1 <- MyBUGSOutput(out, c(uNames("beta", K),
                      "tau","sigma.ri","sigma.rs"))
> print(OUT1, digits = 3)
```

```
            mean     se    2.5%  97.5%
beta[1]   1.4054 0.0592  1.2919  1.523
beta[2]   0.3877 0.0282  0.3323  0.444
beta[3]   0.2485 0.0183  0.2122  0.284
beta[4]   0.2295 0.0214  0.1866  0.270
tau      28.9998 2.5904 24.1494 34.299
sigma.ri  0.2748 0.0521  0.1912  0.394
sigma.rs  0.0771 0.0251  0.0314  0.132
```

All regression parameters are significant, and the mean posterior distribution for σ_i (denoted by `sigma.ri`) and σ_s (denoted by `sigma.rs`) are relatively distant from 0. In Chapters 5 and 6 we discuss how to calculate an AIC for a GLMM estimated within a Bayesian framework. In principle, we can calculate the AIC for the gamma GLMM with random intercept and slope and also for the gamma GLMM with random intercept, to compare the models. We can also leave the model as it is and inspect the residuals for patterns.

Note that because *species* is used as a random intercept and also for the random slope, we can no longer use it in the fixed part of the model.

The saga continues with model validation and model interpretation. The steps are identical to those in Section 4.7 and are not repeated here.

4.9 Truncated Gaussian linear regression

In previous sections we showed how to fit a gamma GLM and a gamma GLMM in JAGS. In this section we apply a different model: a truncated Gaussian multiple linear regression model, also called a Tobit model (Tobin 1958)). Functions for Tobit models exist in R, for example in the `VGAM` package, but adding random effects is complicated. Fitting a multiple linear regression model with a truncated Gaussian distribution in JAGS requires the so-called zero trick. We will begin by illustrating it, show how ordinary multiple linear regression can be fitted using the zero trick, and proceed to modify the JAGS code so that a truncated Gaussian distribution can be used. When this is running we can easily add the random effects.

4.9.1 Zero trick to fit any statistical distribution in JAGS

The zero trick allows fitting any distribution in JAGS. A detailed explanation is given in Nzoufras (2010), and we briefly discuss how it works. Suppose we have a variable y_i that follows a particular distribution with density function $f(y_i, \boldsymbol{\theta})$. The distribution contains unknown (e.g. regression) parameters, which are in $\boldsymbol{\theta}$. The likelihood of y_i is given by

$$L = \prod_{i=1}^{N} f(y_i, \boldsymbol{\theta})$$

If we define $l_i = \log(f(y_i, \boldsymbol{\theta}))$, the likelihood function for all the observations y_1 to y_N can be written as

$$L(y_1,\cdots,y_N;\boldsymbol{\theta}) = \prod_{i=1}^{N} f(y_i;\boldsymbol{\theta}) = \prod_{i=1}^{N} e^{\log(f(y_i,\boldsymbol{\theta}))} = \prod_{i=1}^{N} e^{l_i}$$

Now we do a bit of basic mathematics:

$$L(y_1,\cdots,y_N;\boldsymbol{\theta}) = \prod_{i=1}^{N} e^{l_i} = \prod_{i=1}^{N} \frac{e^{-(-l_i)} \times (-l_i)^0}{0!}$$

The term on the right is the product of a density function of a Poisson distribution with observed value 0 and mean l_i. We can also write this as

$$L(y_1,\cdots,y_N;\boldsymbol{\theta}) = \prod_{i=1}^{N} f_{Poisson}(0;-l_i)$$

Hence the likelihood function for the observations y_1 to y_N can be written as the product of Poisson distributions multiplied by observed value 0 and mean $-l_i$. Because the mean of a Poisson distribution must be non-negative, we add a positive constant C to each term l_i. This is the same as multiplying the likelihood by $\exp(-C)$ and does not affect the likelihood itself (Ntzoufras, 2010). We must select C such that $C - l_i > 0$ for all i. As a result we have

$$L(y_1,\cdots,y_N;\boldsymbol{\theta}) = \prod_{i=1}^{N} f_{Poisson}(0;-l_i + C)$$

In the next subsection we illustrate this approach in JAGS.

4.9.2 Multiple linear regression in JAGS with the zero trick

In this subsection we show how the zero trick can be used for a multiple linear regression model of the form

$$\begin{aligned} &Biomass_i = Intercept + Diameter_i + Height_i + Leaves_i + \varepsilon_i \\ &\varepsilon_i \sim N(0_i,\sigma^2) \end{aligned} \tag{4.5}$$

We use standardized covariates and for the moment ignore the *species* covariate. The model had 4 unknown regression parameters (including the intercept) and a variance that we must estimate using MCMC. First we again create a matrix, **X,** that contains all standardized covariates.

```
> X <- model.matrix(~ 1 + DiameterS + HeightS +
                      LeavesS, data = Seedlings)
> K <- ncol(X)
```

The object `win.data` contains all the variables that we need for MCMC. The new aspect is the variable `Zeros`; it is a vector consisting of zeros.

```
> win.data <- list(
                Biomass = Seedlings$Biomass,
```

```
X       = X,
N       = nrow(Seedlings),
b0      = rep(0, K),
B0      = diag(0.0001, K),
Zeros   = rep(0, nrow(Seedlings))
)
```

The model in Equation (4.5) uses a normal distribution and, for the zero trick in JAGS, we need to calculate the logarithm of the likelihood function. Recall from your first year statistics course that if y_i follows a normal distribution, its density function is given by

$$f(y_i \mid \mu,\sigma) = \frac{1}{\sqrt{2\times\pi\times\sigma^2}}\exp\left(-\frac{(y_i-\mu)^2}{2\times\sigma^2}\right)$$

The logarithm of this density function is as follows:

$$\log\left(f(y_i \mid \mu,\sigma)\right) = -\frac{1}{2}\times\log(2\times\pi) - \frac{1}{2}\times\log(\sigma^2) - \frac{(y_i-\mu)^2}{2\times\sigma^2}$$

The JAGS code is given below. The components in the log-likelihood are designated `l1[i]` and `l2[i]`. The value of C is set at 1000, which is sufficiently large. Diffuse priors are used for the four regression parameters, and a Half-Cauchy(25) distribution is used for σ.

```
sink("SeedlingsLM.txt")
cat("
model{
  #1. Priors for regression coefficients
  beta  ~ dmnorm(b0[], B0[,])
  num   ~ dnorm(0, 0.0016)
  denom ~ dnorm(0, 1)
  sigma <- abs(num / denom)

  #2. Likelihood
  C <- 10000
  for (i in 1:N){
     Zeros[i] ~ dpois(Zeros.mean[i])
     Zeros.mean[i] <- -L[i] + C
     l1[i] <- -0.5 * log(2 * 3.1415) -
              0.5 * log(sigma)
     l2[i] <- -0.5 * pow(Biomass[i]-mu[i],2)/sigma
     L[i]   <- l1[i] + l2[i]
     mu[i]  <- eta[i]
     eta[i] <- inprod(beta[],X[i,])
  }
}
",fill = TRUE)
sink()
```

Initial values and parameters to save are as follows:

```
>inits <- function () {
   list(
    beta  = rnorm(K, 0, 0.01),
    num   = rnorm(1,0, 25),
    denom = rnorm(1,0, 1))}
> params <- c("beta", "sigma")
```

To execute the model we use

```
> J0 <- jags(data      = win.data,
           inits      = inits,
           parameters = params,
           model.file = "SeedlingsLM.txt",
           n.thin     = 10,
           n.chains   = 3,
           n.burnin   = 40000,
           n.iter     = 50000)
> J1.upd <- update(J0, n.iter = 50000, n.thin =10)
> out    <- J1.upd$BUGSoutput
```

In general, using the zero trick requires a greater number of iterations to obtain good mixing. We obtain the numerical output (mixing is good) and compare it to the results obtained by the `lm` function:

```
> OUT1 <- MyBUGSOutput(out, c(uNames("beta", K),
                              "sigma"))
> print(OUT1, digits = 5)

           mean       se    2.5%  97.5%
beta[1] 4.80688 0.086681 4.63512 4.9776
beta[2] 1.78861 0.105167 1.58105 1.9935
beta[3] 1.29046 0.109187 1.07253 1.5048
beta[4] 0.41978 0.092476 0.24257 0.6009
sigma   2.18573 0.185263 1.84910 2.5755
```

```
> M4 <- lm(Biomass  ~ DiameterS + HeightS +
           LeavesS, data = Seedlings)
> summary(M4)

             Estimate Std. Error t value Pr(>|t|)
(Intercept)    4.8058     0.0862   55.73  < 2e-16
DiameterS      1.7906     0.1049   17.06  < 2e-16
HeightS        1.2899     0.1097   11.76  < 2e-16
LeavesS        0.4186     0.0918    4.56  7.5e-06
```

Note that the posterior means obtained by JAGS are similar to those obtained by the `lm` function. Hence the approaches (zero trick and ordinary least squares) give similar results, which indicates that our JAGS code is correct.

4.9.3 Tobit model in JAGS

In the previous subsection we used a normal distribution for *biomass*, but encountered problems because the fitted values were negative. Figure 4.17 illustrates this problem. The graph shows the predicted *biomass* data versus *diameter* for *average height* and *number of leaves*. The grey area is a 95% prediction interval and shows the range in which new *biomass* values are likely to fall. Note that for standardized *diameter* values smaller than -1, the model can predict negative biomass values! The actual fitted values are also smaller than 0 for *diameter* values smaller than -2.5.

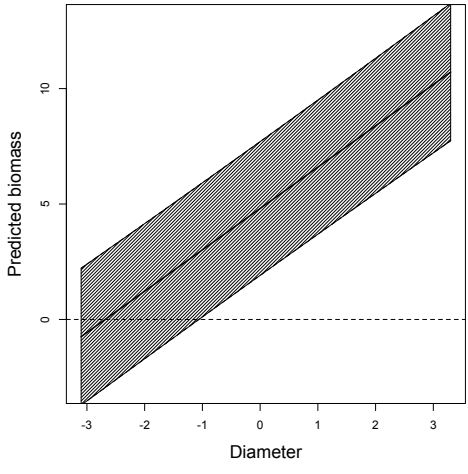

Figure 4.17. Predicted *biomass* plotted versus (standardized) *diameter* values for *average height* and *number of leaves*.

The grey area around the line in Figure 4.17 is a normal distribution. Imagine the normal density curve in Figure 4.18 superimposed on the fitted line in Figure 4.17, although the width and height of the density curve will be different.

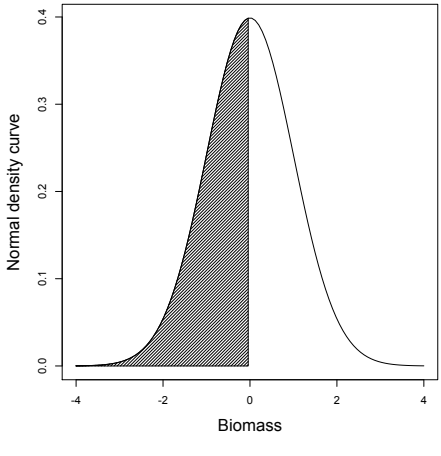

Figure 4.18. Normal density curve.

So what can we do to avoid negative fitted values? In previous sections we used the gamma GLM as a solution. In this section we use a truncated normal distribution, which means that we remove the grey side of the normal distribution in Figure 4.18. Since a distribution must add up to 1, we need to modify the right side. This is done by dividing the probabilities on the right side by 1 – P($y \leq 0$), resulting in the following truncated normal density function.

$$f(y \mid y > 0) = \frac{f(y)}{1 - P(y \leq 0)} = \frac{f(y)}{1 - \Phi(\alpha)}$$

where *f(y)* is the normal distribution, and $\alpha = -\mu / \sigma$. The term $\Phi(.)$ is the standard cumulative normal density function, given by

$$\Phi(\alpha) = \Phi(-\frac{\mu}{\sigma}) = \frac{1}{\sqrt{2 \times \pi}} \times \exp\left(-\frac{\mu^2}{2 \times \sigma^2}\right)$$

In order to run the multiple linear regression with a truncated normal distribution, we adjust the likelihood function in JAGS. R code for this is

```
sink("SeedlingsLMTrunc.txt")
cat("
model{
  #1. Priors for regression coefficients
  beta  ~ dmnorm(b0[], B0[,])
  num   ~ dnorm(0, 0.0016)
  denom ~ dnorm(0, 1)
  sigma <- abs(num / denom)
  tau <- 1 / (sigma * sigma)
  #2. Likelihood
  C <- 10000
  for (i in 1:N){
     Zeros[i] ~ dpois(Zeros.mean[i])
     Zeros.mean[i] <- -L[i] + C
     l1[i]  <- -0.5 * log(2 * 3.1415) -
               0.5 * log(sigma)
     l2[i]  <- -0.5 * pow(Biomass[i] - mu[i], 2) /
               sigma
     alpha[i] <- -mu[i] / sigma
     phi[i] <- (1 / sqrt(2 * 3.1415)) *
               exp(-0.5 *  (alpha[i]^2))
     l3[i]  <- log(1 - phi[i])
     L[i]   <-  l1[i] + l2[i] - l3[i]
     mu[i]  <- eta[i]
     eta[i] <- inprod(beta[],cX[i,])
  }
}
",fill = TRUE)
```

```
sink()
```

The results are as follows:

```
           mean      se    2.5%   97.5%
beta[1] 4.58407 0.12330 4.34479 4.82178
beta[2] 1.77464 0.14776 1.49005 2.06169
beta[3] 1.30206 0.15694 1.00491 1.60504
beta[4] 0.42219 0.13107 0.16584 0.67385
sigma   4.28973 0.50616 3.39138 5.38310
```

The mean and variance of the truncated normal distribution is different from the mean and variance of a normal distribution. They are given by

$$E\left(y \mid y>0\right)=\mu+\sigma\times\lambda(\alpha)$$

$$\text{var}\left(y \mid y>0\right)=\sigma^2\times(1-\delta(\alpha))$$

where

$$\alpha=\frac{0-\mu}{\sigma}$$

$$\lambda(\alpha)=\frac{\phi(\alpha)}{1-\Phi(\alpha)}$$

$$\delta(\alpha)=\lambda(\alpha)\times\left(\lambda(\alpha)-\alpha\right)$$

$\phi(.)$ is the standard normal density function. These equations look complicated, but are easy to calculate within JAGS.

4.9.4 Tobit model with random effects in JAGS

The Tobit model presented in Subsection 4.9.3 can easily be extended with a random intercept, or a random intercept and slope. See Section 4.8 for code. We only need to change the predictor function `eta[i]` to

```
eta[i] <- inprod(beta[],X[i,])  + a[re[i]] +
                  b[re[i]] * X[i,2]
```

The JAGS code also requires diffuse priors for the random intercepts and slopes. We leave it as an exercise for the reader.

4.10 Discussion

A gamma GLM is a component of all GLM software. The distribution is employed for positive-only continuous response models. An alternative distribution for positive-only data is the inverse-Gaussian, which was not discussed here. We focused on the gamma model in this chapter, since it is used substantially more often in research. Negative binomial models can be considered as Poisson-gamma mixture models, with the negative binomial dispersion parameter being the gamma scale parameter.

GLMs are single-parameter models in which the mean, μ, is estimated. However, the traditional gamma distribution is similar to the normal

distribution in that it has a second scale parameter. Two-parameter gamma models allow much more flexibility in the model than single-parameter GLM models. Following an examination of GLM gamma models, we used JAGS to estimate the parameters of two-parameter gamma models and subsequently of two-parameter gamma mixed models.

Finally we looked at difficulties modelling non-negative continuous response data, in which zero counts are not only allowed. We looked at Tobit models when considering such data as being a constituent of an underlying normal distribution.

We found that it is relatively easy to model a wide variety of continuous response data using JAGS.

4.11 What to present in a paper

For a paper, we strongly suggest presenting Figure 4.3, as it clearly shows the patterns in the data. Present the mathematical formulation of the model that is used, and mention that estimations were calculated using JAGS. See Section 3.10 for sentences on MCMC (number of iterations, thinning rate, burn-in, mixing, posterior means, etc.). We also recommend including Figure 4.10, as it shows the covariate effects.