Practical 4: Generalised linear models

BIOM4025 - Statistical Modelling

Erik Postma - e.postma@exeter.ac.uk

Weeks 7-8: 4-15 November 2024

Contents

1 Introduction

In many species, testosterone (T) levels are positively correlated with male reproductive success. For example, males with high levels of T may be larger, more aggressive and develop more elaborate secondary sexual characters. As a consequence, they will be more successful in male-male competition, and more attractive to females. However, if there are such obvious benefits to high T levels, why do we still see substantial variation among individuals? One appealing and often hypothesised explanation for the maintenance of variation in T, and in fitness-related traits in general, is that there are not only benefits, but also downsides to high T levels, *i.e.* there are trade-offs. For example, T may have immunosuppressant effects and increase an individual's susceptibility to parasites.

Despite a substantial body of work on the role of androgens in mediating a trade-off between reproductive success and health in *males*, Smyth *et al.* (2016) set out to investigate its role in *female* meerkats (*Suricata suricatta*), a cooperatively breeding species. In this species, dominant females have significantly higher levels of androgens than subordinates. Although dominant females have a dramatically higher reproductive success, do they suffer from higher levels of parasitism?

PREPARATION
To familiarise yourself with the topic and study system, read the paper that accompanies the data set we will be using for this practical: Smyth *et al.*, 2016. Androgens predict parasitism in female meerkats: a new perspective on a classic trade-off. *Biology Letters* **12**: 20160660. http://dx.doi.org/10.1098/rsbl.2016.0660. A PDF of the paper can also be found on the ELE page.

Many journals, including *Biology Letters*, have made it compulsory to make the data on which an article is based publicly available. This is good news for us, as it allows us to reanalyse their data and compare our results to those presented in the paper. For this article, the data can be downloaded here or from the ELE page.

NOTE 1
Only have a look at the **HARD EXERCISES** once you have finished with all the others, and don't worry if you can't solve all of them: Some of them are pretty hard!

NOTE 2
Whereas Sections 2 and 3 cover data handling, transformations, linear models and non-parametric tests, the use of GLMs is covered in Sections 4-6. If you are particularly interested in learning more about the latter, you might want to skip 2 and 3 for now.

2 Getting started

2.1 Importing the data

Copy the data file into a folder on your computer and have a look at it using *Excel* or the text editor of your choice. Do the column names contain any spaces? Are they short but informative? Do any of the columns contain missing values? Make any changes you think are necessary. Note that below I may use slightly different column names from you, so be careful when copy-pasting any of the code.

Once you are happy with what your data file looks like, import it into `R`. You will have to tell `R` where to find the file using `setwd()` and then use `read.table()` to read it into `R`:

```
setwd("/This/Is/Path/To/My/Working/Directory")
t.data <- read.table("Dataset.csv", header=TRUE, sep = ",",
                      stringsAsFactors = FALSE)
```

Have the data imported correctly? Are all the rows and columns there? Are the column names okay? Are the columns containing numbers `numeric` and the others of type `character`? Any `factors`?

TIP
Sometimes you might want to change the column names of your data frame after you have imported your data. You can do this using `colnames()`. This creates a vector containing the column names, and you can then change (specific elements of) this vector. For example, it might be a good idea to shorten the names of the columns that contain the parasite data (columns 11 to 16):

```
colnames(t.data)
```

```
[1] "individual"            "date"                  "status"
[4] "age"                   "weight"                "group.size"
[7] "rainfall"              "pregnant"              "fam"
[10] "psr"                   "Strongyle_abundance"   "Toxocara_abundance"
[13] "Oxynema_abund"         "Pseudandrya_abundance" "Spirurida_abundance"
[16] "Coccidia_abundance"
```

```
colnames(t.data)[11:16] <- c("strong", "toxo", "oxy", "pseuda", "spiru", "cocci")
colnames(t.data)
```

```
[1] "individual" "date"       "status"     "age"        "weight"
[6] "group.size" "rainfall"   "pregnant"   "fam"        "psr"
[11] "strong"     "toxo"       "oxy"        "pseuda"     "spiru"
[16] "cocci"
```

2.2 Descriptive statistics

Before we start applying linear and general linear models to these data, let's try and get a better understanding of what is in each of the columns.

First of all, let's count the number of observations per individual:

```
table(t.data$individual)
```

```

VAZF029 VAZF036 VAZF052 VBBF069 VBBF083 VBBF093  VDF115  VDF146  VDF162  VDF163
      1       1       3       2       1       1       2       1       2       2
VJXF035 VJXF052 VJXF053 VJXF057 VKUF019 VKUF055 VKUF065 VKUF069  VLF102  VLF111
      1       1       1       1       2       1       3       2       2       1
 VLF134  VLF180  VLF194 VPAF009 VRRF116 VRRF146 VRRF151 VRRF156 VSQF005 VSQF011
      1       1       3       1       1       1       2       2       1       1
VSQF012 VTYF042 VVHF029 VVHF035 VVHF064  VWF176  VWF177
      2       1       1       1       1       3       1
```

Whereas this gives the number of times each individual appears in `data$individual`, we would also like to know how many individuals have been included once, twice and three times. We can do this by creating a table of a table:

```
counts.id <- table(t.data$individual)
table(counts.id)
```

```
counts.id
 1  2  3
23 10  4
```

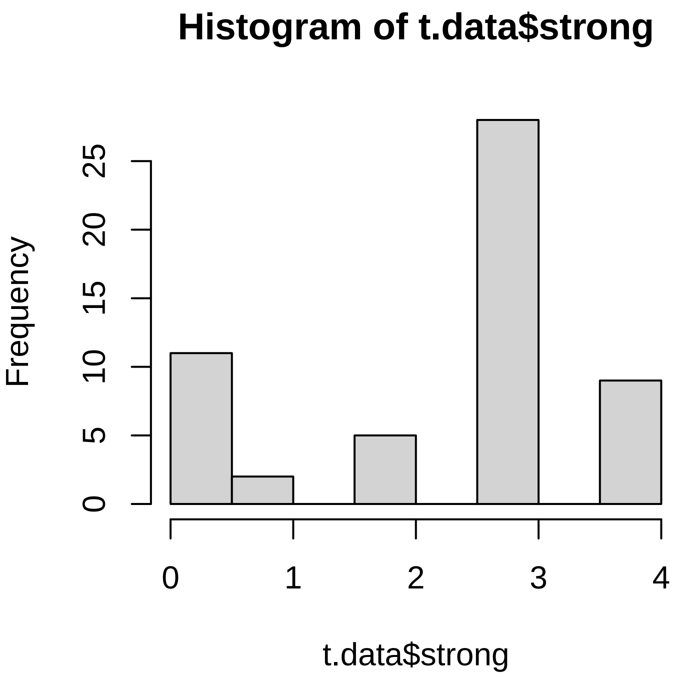
This reveals that although most individuals have been included only once, there are ten individuals that have been sampled twice, and three individuals have been sampled three times. To account for this, the authors use mixed models for the analysis for some, but not all, of the variables. Mixed models are the topic of the next two lectures and next week's practical. Therefore we leave the potential problems posed by repeated measures for now.

> **EXERCISE 4.1**
> Confirm the sample sizes provided in section 2a of the paper ('Study site and subjects') using `length()`, `table()` and `unique()`. The latter removes any duplicates from a vector or data frame.

2.3 Quantifying parasite abundance and species richness

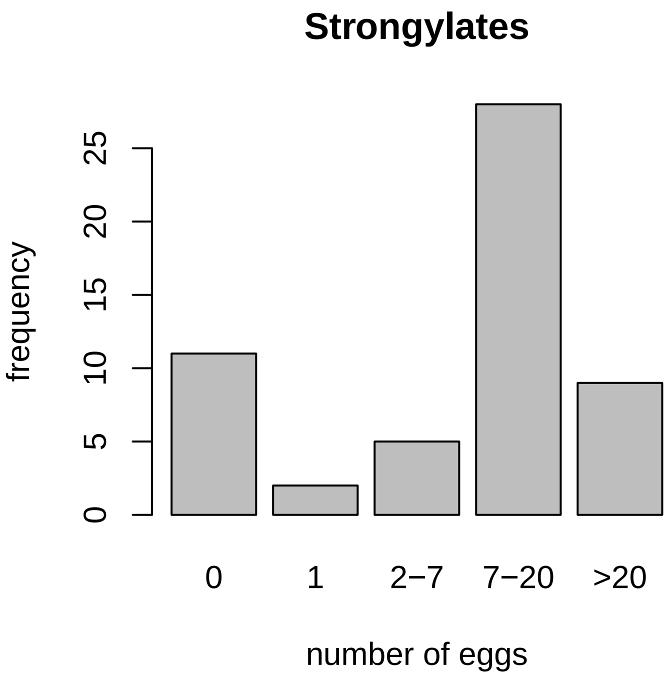
Abundance of each of the parasites is scored on a scale from 0 to 4. Trying to plot the distribution of discrete count data using `hist()` often doesn't works so well. For example, let's visualise the parasite abundance for `strong` using `hist()`:

```
hist(t.data$strong)
```



Instead, we can first create a table using `table()`, and then create a `barplot()` based on this table:

```
table.strong <- table(t.data$strong)
barplot(table.strong, xlab="number of eggs", ylab="frequency",
        main = "Strongylates",
        names.arg=c("0", "1", "2-7", "7-20", ">20"))
```



EXERCISE 4.2
Discuss the pros and cons of the authors' measure of parasite abundance. And do you think it is appropriate to analyse this with a GLM with Poisson errors?

In addition to their measures of the presence and abundance of each parasite species, the authors also used a measure of parasite species richness (PSR), which they obtained by counting the number of parasite taxa that were present in a sample.

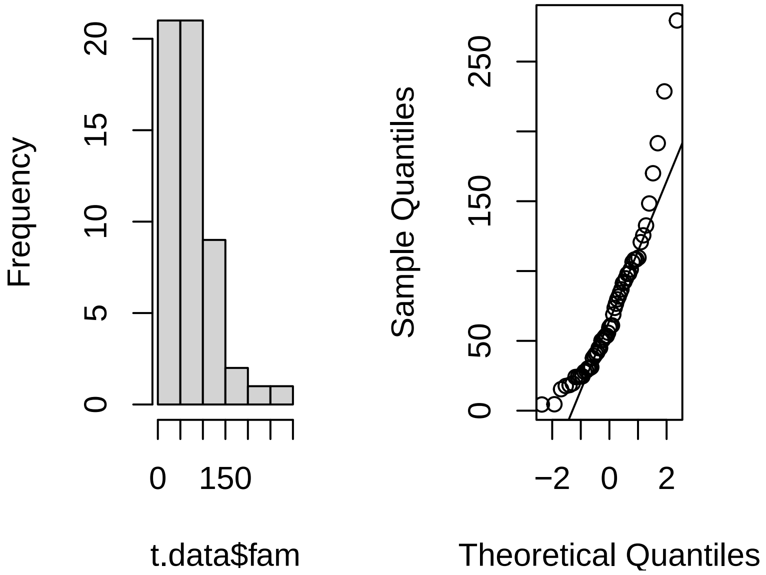
3 Variation in faecal androgen metabolites (FAM)

Rather than measuring the level of T in the blood, the authors measured the amount of androgen metabolites in faecal samples (FAM). Before we start looking at its relationship with parasite presence, diversity and abundance, we should have a closer look at FAM itself. Do we find that dominant females have higher levels of FAM?

3.1 Checking for normality

Let us first of all have a look at the distribution of `fam` by plotting a histogram and a QQ-plot:

```
par(mfrow=c(1,2))
hist(t.data$fam, main=NULL)
qqnorm(t.data$fam, main = NULL)
qqline(t.data$fam)
```



Unfortunately this doesn't look very normal at all. We can confirm this with a Shapiro-Wilk test, which reveals a significant deviation from normality:

```
shapiro.test(t.data$fam)
```

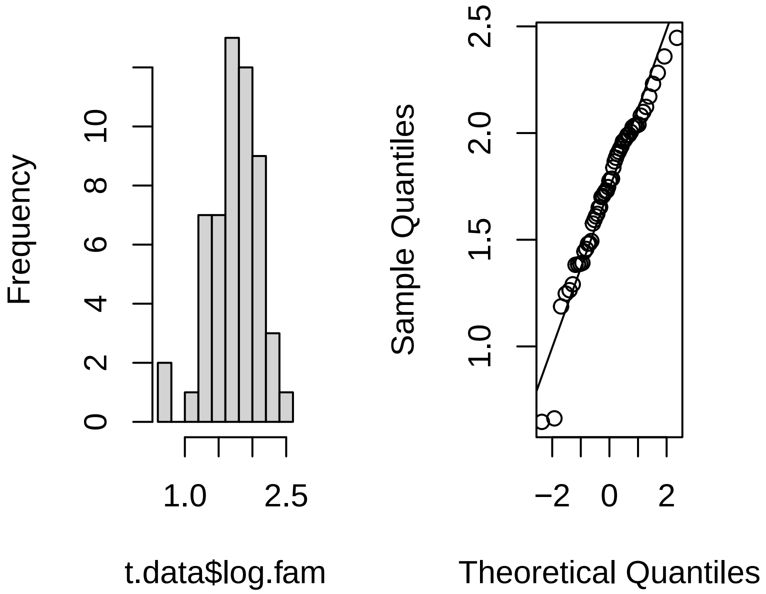
```

	Shapiro-Wilk normality test

data:  t.data$fam
W = 0.87091, p-value = 2.794e-05
```

The latter result should however be taken with a grain of salt: Provided with enough data, a Shapiro-Wilk test will almost always come up significant. Similarly, if you have very little data, you will lack the statistical power to detect even severe deviations from normality.

3.2 Transformations

Given that FAM is neither a count nor a proportion, a GLM is unlikely to provide a solution. Instead we might be better off transforming our data. A common transformation to make right-skewed data such as these more symmetrical is the log transformation:

```
par(mfrow=c(1,2))
t.data$log.fam <- log10(t.data$fam)
hist(t.data$log.fam, main = NULL)
qqnorm(t.data$log.fam, main = NULL)
qqline(t.data$log.fam)
```



NOTE
`log()` computes natural logarithms (ln), and `exp()` uses base e. So `log(10)` gives $ln(10) = 2.303$, and `exp(1)` is equal to $e^1 = 2.718$. To use base 10 logarithms, use `log10()`. It doesn't matter which one we use for our log transformation, but here we follow the authors and use log_{10}.

This looks a lot better! The Shapiro-Wilk test still rejects the null-hypothesis that our data comes from a normal distribution, but I wouldn't be too worried about this.

```
shapiro.test(t.data$log.fam)
```

```


	Shapiro-Wilk normality test

data:  t.data$log.fam
W = 0.95549, p-value = 0.04042
```

EXERCISE 4.3 (HARD)
Use the `boxcox()` function that is part of the `MASS` package to apply a Box-Cox transformation to `fam` and find the value of λ that brings it closest to normality. Use this value of λ to transform the data and compare the result to that provided by the log-transformation.

3.3 Testing for a difference in FAM between dominant and subordinate females

If we are happy with our log transformation, we can now test whether there is a difference in `log.fam` between dominant and subordinate females using `lm()`:

```r
m <- lm(log.fam ~ status, data=t.data)
summary(m)
```

```

Call:
lm(formula = log.fam ~ status, data = t.data)

Residuals:
    Min      1Q  Median      3Q     Max 
-1.0089 -0.1665  0.0490  0.2363  0.6269 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  1.92772    0.08359  23.061  < 2e-16 ***
statusS     -0.27237    0.10057  -2.708  0.00908 ** 
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.3447 on 53 degrees of freedom
Multiple R-squared:  0.1216,    Adjusted R-squared:  0.105 
F-statistic: 7.335 on 1 and 53 DF,  p-value: 0.009084
```

EXERCISE 4.4
Compare this result to that provided in the Electronic Supplementary Material (ESM). Did the authors find the same result?

Although this reveals significantly lower values of `log.fam` in subordinate females, these estimates are on a log_{10} scale, and we have to back-transform them to get the predicted means for both groups of females: The predicted FAM concentration for a dominant female (the intercept) is $10^{1.93}$=84.67 and for a subordinate female it would be $10^{1.93+-0.27}$ =45.22.

Rather than copy-pasting the coefficients from the output, we can extract them from our model and use these for our calculations:

```r
b.intercept <- m$coefficients[1]
b.status <- m$coefficients[2]
10^b.intercept             # dominant females
```

```
(Intercept) 
   84.66806 
```

```r
10^(b.intercept+b.status) # subordinate females
```

```
(Intercept) 
   45.22179 
```

EXERCISE 4.5 (VERY HARD)
How do these predictions compare to the means reported in the ESM (and those provided by `aggregate(t.data$fam, by=list(t.data$status), mean)`? Do you understand why they are not the same?

Rather than transforming FAM, we could also have used a non-parametric Wilcoxon test to compare the two groups:

```r
wilcox.test(fam ~ status, data=t.data)
```

```

    Wilcoxon rank sum exact test
```

```
data:  fam by status
W = 474, p-value = 0.005315
alternative hypothesis: true location shift is not equal to 0
```

This is in line with the results from our linear model applied to `log.fam`, as it again shows a statistically significant difference between dominant and subordinate females.

> **EXERCISE 4.6**
> Repeat the Wilcoxon test, but now use `log.fam` rather than `fam`. Does this give different results? Why? Why not?

4 Variation in parasite burden

Now that we have established that dominant females have higher levels of FAM, and presumable higher levels of circulating T, we are ready to explore the consequences this may have for their parasite burden. Although our explanatory variables do not need to be normally distributed, the authors have chosen to use log-transformed FAM for all their analyses. This is not an uncommon thing to do when it comes to concentrations. To make our results directly comparable to theirs, we will therefore do the same. However, try to forget about this log-transformation of FAM for what is to come.

Our main goal is to test if there is an effect of `log.fam` on parasite species richness (`psr`). However, there are many other variables potentially contributing to variation in `psr`, such as `status`, `rainfall`, `weight`, `pregnant` and `group.size`.

Because `psr` is a count of the number of parasites found in a female (or rather in her faeces), the authors have chosen to use a generalised linear model with a Poisson error distribution and a log link function.

4.1 Model simplification

We start with a full model, including all predictors:

```
m.full <- glm(psr ~ log.fam + rainfall + weight + pregnant + group.size + status,
              data=t.data, family=poisson(link=log))
```

Note that `link=log` is the default for `family=poisson`, so we don't have to specify this (but there is no harm in doing so).

We can now use `summary()` to look at the results:

```
summary(m.full)
```

```

Call:
glm(formula = psr ~ log.fam + rainfall + weight + pregnant + 
    group.size + status, family = poisson(link = log), data = t.data)

Coefficients:
              Estimate Std. Error z value Pr(>|z|)  
(Intercept)  0.2339596  0.7954368   0.294   0.7687  
log.fam      0.6233408  0.2664444   2.339   0.0193 *
rainfall    -0.0054760  0.0051886  -1.055   0.2912  
weight      -0.0001061  0.0008050  -0.132   0.8952  
pregnantY    0.0548131  0.2154183   0.254   0.7991  
group.size   0.0031423  0.0151158   0.208   0.8353  
statusS     -0.3313721  0.2252175  -1.471   0.1412  
---
```

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

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 42.266  on 54  degrees of freedom
Residual deviance: 24.639  on 48  degrees of freedom
AIC: 193.12

Number of Fisher Scoring iterations: 4
```

The first thing we do is check for overdispersion. In this case, the residual deviance (*i.e.* the deviance not explained by the model) is actually smaller than the residual degrees of freedom (the sample size minus the number of parameter estimates, so 55-7=48). So in this case we do not need to worry about overdispersion (we might have to worry about *underdispersion*, but more on this later…).

The p-values provided by applying `summary()` to a GLM are only approximate, so to formally test if any of the predictors are significant, we need to drop them from the model one-by-one and compare a model without this term to the full model. This is a very laborious process, and we didn't even include any interactions!

A useful function that allows us to remove or include specific terms, is `update()`. So to fit a model without `weight` (the term in `m.full` that appears to be the least significant), we can run:

```
summary(m.reduced.1 <- update(m.full, . ~ . - weight))
```

```

Call:
glm(formula = psr ~ log.fam + rainfall + pregnant + group.size + 
    status, family = poisson(link = log), data = t.data)

Coefficients:
             Estimate Std. Error z value Pr(>|z|)  
(Intercept)  0.164839   0.597347   0.276   0.7826  
log.fam      0.619735   0.264504   2.343   0.0191 *
rainfall    -0.005574   0.005138  -1.085   0.2780  
pregnantY    0.042028   0.192635   0.218   0.8273  
group.size   0.003244   0.015099   0.215   0.8299  
statusS     -0.316616   0.195173  -1.622   0.1048  
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 42.266  on 54  degrees of freedom
Residual deviance: 24.657  on 49  degrees of freedom
AIC: 191.14

Number of Fisher Scoring iterations: 4
```

Because the difference in the residual deviance between the two models is approximately Chi-square distributed, we can compare `m.reduced.1` to `m.full` using a Chi-square test (and not an F-test as we did before) with `anova()` and ask if the removal of `weight` has made the model significantly worse:

```
anova(m.reduced.1, m.full, test="Chisq")
```

```
Analysis of Deviance Table

Model 1: psr ~ log.fam + rainfall + pregnant + group.size + status
Model 2: psr ~ log.fam + rainfall + weight + pregnant + group.size + status
  Resid. Df Resid. Dev Df Deviance Pr(>Chi)
1        49     24.657
2        48     24.639  1 0.017322   0.8953
```

As expected from a simpler model, `m.reduced.1` explains a little bit less variation in `psr` (the deviance has increased slightly). However, this reduction is not statistically significant (p=0.895). In other words, we can remove `weight` from our model without making the model significantly worse.

Alternatively we could compare the AIC values for both models:

```
AIC(m.full, m.reduced.1)
```

```
            df      AIC
m.full       7 193.1237
m.reduced.1  6 191.1410
```

As lower AIC values are better, `m.reduced.1` again comes out on top. However, remember that although this tells us that of the two models, `m.reduced.1` is the better one, this may still be a bad model.

To test the significance of all predictors, we would have to repeat all of the steps above for each one of them. Conveniently we can speed this up a bit by using the `drop1()` function. Applying this function to `m.full` will one-by-one drop a term from our model and test whether this resulted in a significantly worse fit. It will also provide the AIC for each of these reduced models. Remember that because we are comparing GLMs, we need to specify `test='Chisq'`:

```
drop1(m.full, test="Chisq")
```

```
Single term deletions

Model:
psr ~ log.fam + rainfall + weight + pregnant + group.size + status
           Df Deviance    AIC    LRT Pr(>Chi)  
<none>          24.639 193.12                  
log.fam     1   30.403 196.89 5.7634  0.01636 *
rainfall    1   25.790 192.27 1.1506  0.28343  
weight      1   24.657 191.14 0.0173  0.89529  
pregnant    1   24.704 191.19 0.0645  0.79957  
group.size  1   24.682 191.17 0.0430  0.83580  
status      1   26.805 193.29 2.1653  0.14116  
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

If we have a good *a-priori* reason for including all of these predictors, the fact that some of them turn out to be non-significant is an interesting result and we could stop here. However, sometimes we would like to have the minimal adequate model, *i.e.* the model that is simpler and explains less variation than the full model, but not significantly so.

> **EXERCISE 4.7**
> Use backward elimination to arrive at the minimal adequate model.

Backward elimination has left us with a model that contains `log.fam` but not `status`. However, in 3.3 we had found an effect of `status` on `log.fam`. In other words, both effect are correlated, or *collinear*. Indeed, if we fit a model that includes only `status`, we find a highly significant difference in `psr` between subordinate and dominant females:

```
summary(glm(psr ~ status, data=t.data, family=poisson))
```

```

Call:
```

```
glm(formula = psr ~ status, family = poisson, data = t.data)

Coefficients:
            Estimate Std. Error z value Pr(>|z|)    
(Intercept)   1.3715     0.1222  11.226  < 2e-16 ***
statusS      -0.4873     0.1606  -3.034  0.00241 ** 
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 42.266  on 54  degrees of freedom
Residual deviance: 33.374  on 53  degrees of freedom
AIC: 191.86

Number of Fisher Scoring iterations: 4
```

However, during backward elimination, `status` is dropped because when included together with `log.fam` it is (just) not significant:

```
drop1(glm(psr ~ log.fam + status, data=t.data, family=poisson(link=log)), test="Chisq")
```

```
Single term deletions

Model:
psr ~ log.fam + status
        Df Deviance    AIC    LRT Pr(>Chi)   
<none>       26.163 186.65                   
log.fam  1   33.374 191.86 7.2114 0.007244 **
status   1   29.273 187.76 3.1103 0.077797 . 
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

This suggests that although there is a difference in `psr` between subordinate and dominant females, much of this difference is accounted for by the difference in `log.fam` between the two groups. In line with this, including `log.fam` reduces the parameter estimate for `status` from -0.49 to -0.31. Furthermore, when we compare the AIC value of a model that includes only `status` to a model that includes only `log.fam`, we find that the latter has a lower AIC:

```
m.status <- glm(psr ~ status, data=t.data, family=poisson)
m.log.fam <- glm(psr ~ log.fam, data=t.data, family=poisson)
AIC(m.status, m.log.fam)
```

```
          df      AIC
m.status   2 191.8585
m.log.fam  2 187.7575
```

```
summary(m.log.fam)
```

```

Call:
glm(formula = psr ~ log.fam, family = poisson, data = t.data)

Coefficients:
            Estimate Std. Error z value Pr(>|z|)    
(Intercept)  -0.4644     0.4568  -1.017 0.309330    
log.fam       0.8520     0.2446   3.483 0.000496 ***
```

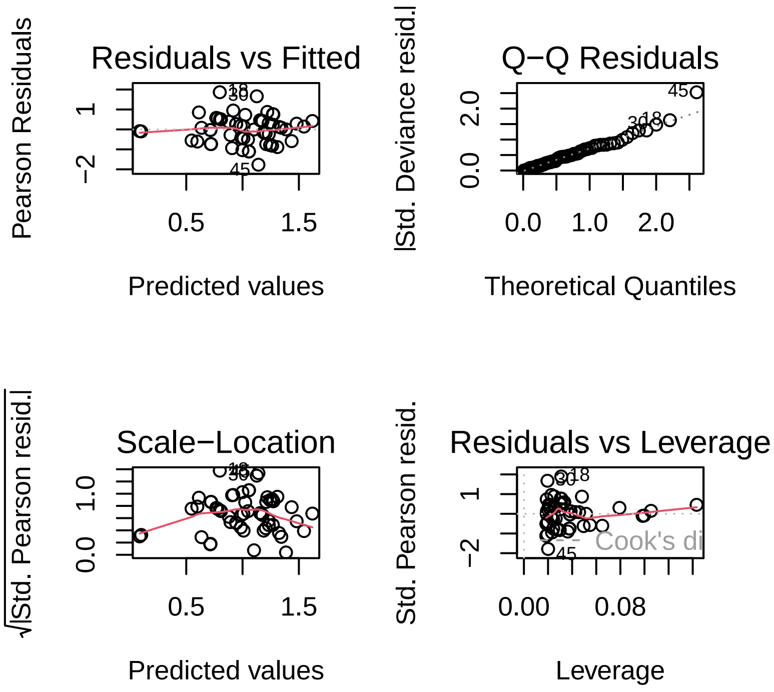
```
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 42.266  on 54  degrees of freedom
Residual deviance: 29.273  on 53  degrees of freedom
AIC: 187.76

Number of Fisher Scoring iterations: 4
```

```
anova(m.log.fam, test="Chisq")
```

```
Analysis of Deviance Table

Model: poisson, link: log

Response: psr

Terms added sequentially (first to last)


        Df Deviance Resid. Df Resid. Dev  Pr(>Chi)
NULL                       54     42.266
log.fam  1   12.993        53     29.273 0.0003127 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Finally, let's have a look at the diagnostic plots for `m.log.fam`:

```
par(mfrow=c(2,2))
plot(m.log.fam)
```



EXERCISE 4.8
Are these diagnostic plots a reason to worry?

On the whole, we can conclude that there is a positive effect of `log.fam` on `psr`. If we would have to report this result in a paper, we need to report the test statistic, the degrees of freedom, and the p-value. As we did a Chi-square test, we need to report the Chi-square (χ^2) value. However, although we asked `anova()` to perform a Chi-square test, at first sight it doesn't seem to have provided us with a Chi-square value. However, in these models, our Chi-square value is equal to the difference in the deviance between the two models, so in this case χ^2=12.99. Finally, it is good practice to report a measure of effect size, such as the parameter and standard error for `log.fam`. However, it is important to point out to the reader that this estimate is on the log-scale. Although this makes it difficult interpret the estimate quantitatively, the fact that the estimate is positive tells use that parasite diversity increases with FAM.

In our paper we could write something along these lines: "There was a significantly positive relationship between the concentration of androgen metabolites and parasite species richness ($b \pm SE = 0.852 \pm 0.245; \chi^2_1 = 13.0, P = 0.0003$). Note that the parameter estimate is on the log-scale."

> **EXERCISE 4.9**
> Compare your findings to those reported by the authors. Do they find the same results? Why (not)?

4.2 Plotting the effect of FAM on PSR

Having established that there is a relationship between `fam` and `prs`, we need to give the reader an idea of the direction and strength of the relationship. This is particularly important when it comes to generalised linear models, as the parameter estimates are difficult to interpret. To this end, we would like to make a plot that is similar to Figure 1 in Smyth *et al.*.

In a first step, we need to obtain the predicted `prs` for all values of `log.fam` on the transformed (*i.e.* log) scale. Note that these estimates are on a log-scale not because we log-transformed FAM, but because we specified `family=poisson(link=log)`. We can do this by extracting the parameter estimates from `m.log.fam`:

```
m.log.fam$coefficients
```

```
(Intercept)     log.fam
 -0.4643784   0.8519805
```

```
b0 <- m.log.fam$coefficients[1]
b1 <- m.log.fam$coefficients[2]
```

We now use these to predict `prs` for each value of `log.fam`:

```
log.psr.predicted <- b0 + t.data$log.fam*b1
```

However, these predictions are on the log-scale, so before we can add them to our plot of `prs` against `log.fam`, we need to transform them back to the scale of the original data:

```
psr.predicted <- exp(log.psr.predicted)
```

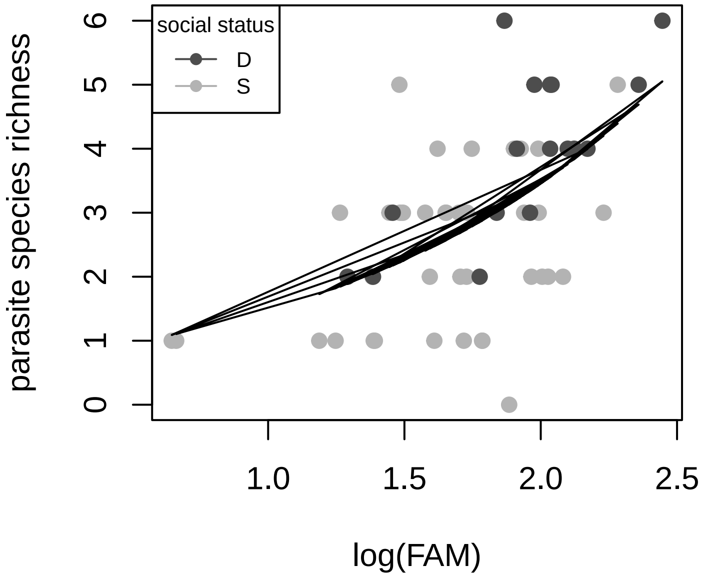
Alternatively, we can have `predict()` do the back-transformation of `log.psr.predicted` for us by specifying `type='response'`:

```
psr.predicted <- predict(m.log.fam, type='response')
```

Now we are ready to create our plot. Because we would like to plot the data for dominant and subordinate females separately, we start with a plot using all data but without any symbols. This is a trick to get the scaling of the axes right. We then go on to add the data points for the dominants and subordinates, the regression line, and a legend:

```
plot(psr ~ log.fam, data=t.data, pch=NA,
     xlab="log(FAM)", ylab="parasite species richness")
points(psr ~ log.fam, data=t.data[t.data$status=="S", ], pch=19, col="grey70")
points(psr ~ log.fam, data=t.data[t.data$status=="D", ], pch=19, col="grey30")
```

```
lines(psr.predicted ~ t.data$log.fam)

legend(x="topleft", legend=c("D", "S"), pch=19, lty = 1,
       col=c("grey30", "grey70"), title="social status", cex=0.7)
```



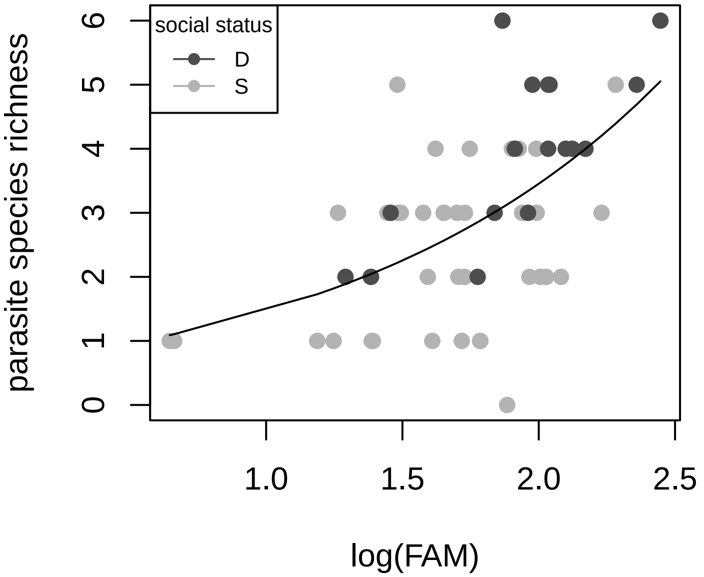
That doesn't look very good, but it becomes clear what the problem is when we look at what we are trying to plot:

```
head(cbind(psr.predicted, t.data$log.fam))
```

```
  psr.predicted
1      4.393303 2.282282
2      3.556989 2.034428
3      3.558569 2.034949
4      3.176456 1.901622
5      2.675722 1.700271
6      3.354342 1.965578
```

The line we are plotting connects each of these points one-by-one, so to create a single continuous line we need to sort both columns by the variable on the x-axis (`t.data$log.fam`):

```
plot(psr ~ log.fam, data=t.data, pch=NA,
     xlab="log(FAM)", ylab="parasite species richness")
points(psr ~ log.fam, data=t.data[t.data$status=="S", ], pch=19, col="grey70")
points(psr ~ log.fam, data=t.data[t.data$status=="D", ], pch=19, col="grey30")

lines(psr.predicted[order(t.data$log.fam)] ~ sort(t.data$log.fam))

legend(x="topleft", legend=c("D", "S"), pch=19, lty = 1,
       col=c("grey30", "grey70"), title="social status", cex=0.7)
```



EXERCISE 4.10
If anything, our data showed signs of underdispersion, *i.e.* the variance in `psr` is smaller than expected if `psr` follows a Poisson distribution. If we wanted to account for underdispersion, how would we do this? Would this change our results?

5 Explaining variation in parasite presence

In addition to analysing the number of parasite species present (`psr`), the authors also test for an effect of `log.fam` on the probability that a certain species is present. To this end, they fit a series of binomial models. Here we will focus on one of the parasite species, *Oxynema suricattae*.

You will first need to create a new presence/absence variable that is `0` if *O. suricattae* is absent, and `1` if it is present:

```
t.data$oxy.present <- ifelse(t.data$oxy==0, 0, 1)
```

5.1 Model simplification

We start by fitting a full model, similar to `m.full`, but this time using `family=binomial` and `link=logit`:

```
m.full.oxy <- glm(oxy.present ~ log.fam + rainfall + weight + pregnant + group.size + status,
                  data=t.data, family=binomial(link=logit))
summary(m.full.oxy)
```

```

Call:
glm(formula = oxy.present ~ log.fam + rainfall + weight + pregnant +
    group.size + status, family = binomial(link = logit), data = t.data)

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  0.269730   3.468897   0.078   0.9380
log.fam      2.624361   1.126246   2.330   0.0198 *
rainfall     0.005752   0.020428   0.282   0.7783
weight      -0.003495   0.003243  -1.078   0.2812
pregnantY   -0.410522   0.951423  -0.431   0.6661
```

```
group.size   0.002279   0.072676   0.031   0.9750  
statusS     -2.765048   1.225766  -2.256   0.0241 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 74.031  on 54  degrees of freedom
Residual deviance: 56.163  on 48  degrees of freedom
AIC: 70.163

Number of Fisher Scoring iterations: 5
```

Again, because `link=logit` is the default link function for `family=binomial`, we don't need to specify it. Also, remember that we can't detect overdispersion in binary traits, so there is no need to check for it this time.

Just like we did before, we can use backward elimination to arrive at the minimal adequate model:

```r
drop1(m.full.oxy, test="Chisq")
```

```
Single term deletions

Model:
oxy.present ~ log.fam + rainfall + weight + pregnant + group.size + 
    status
           Df Deviance    AIC    LRT Pr(>Chi)   
<none>          56.163 70.163                   
log.fam     1   62.940 74.940 6.7775 0.009231 **
rainfall    1   56.243 68.243 0.0800 0.777244   
weight      1   57.406 69.406 1.2429 0.264917   
pregnant    1   56.350 68.350 0.1872 0.665272   
group.size  1   56.164 68.164 0.0010 0.974972   
status      1   62.870 74.870 6.7069 0.009604 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
m.reduced.oxy.1 <- update(m.full.oxy, . ~ . - group.size)
drop1(m.reduced.oxy.1, test="Chisq")
```

```
Single term deletions

Model:
oxy.present ~ log.fam + rainfall + weight + pregnant + status
         Df Deviance    AIC    LRT Pr(>Chi)   
<none>        56.164 68.164                   
log.fam   1   63.087 73.087 6.9230 0.008509 **
rainfall  1   56.249 66.249 0.0855 0.770016   
weight    1   57.512 67.512 1.3480 0.245625   
pregnant  1   56.350 66.350 0.1864 0.665940   
status    1   62.873 72.873 6.7093 0.009591 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
m.reduced.oxy.2 <- update(m.reduced.oxy.1, . ~ . - rainfall)
drop1(m.reduced.oxy.2, test="Chisq")
```

```
Single term deletions

Model:
oxy.present ~ log.fam + weight + pregnant + status
         Df Deviance    AIC    LRT Pr(>Chi)   
<none>        56.249 66.249                   
log.fam   1   63.089 71.089 6.8402 0.008913 **
weight    1   57.512 65.512 1.2629 0.261094   
pregnant  1   56.414 64.414 0.1643 0.685273   
status    1   62.877 70.877 6.6277 0.010041 * 
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
m.reduced.oxy.3 <- update(m.reduced.oxy.2, . ~ . - pregnant)
drop1(m.reduced.oxy.3, test="Chisq")
```

```
Single term deletions

Model:
oxy.present ~ log.fam + weight + status
        Df Deviance    AIC    LRT Pr(>Chi)   
<none>       56.414 64.414                   
log.fam  1   63.149 69.149 6.7350 0.009454 **
weight   1   58.226 64.226 1.8122 0.178248   
status   1   62.904 68.904 6.4904 0.010846 * 
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
m.reduced.oxy.4 <- update(m.reduced.oxy.3, . ~ . - weight)
drop1(m.reduced.oxy.4, test="Chisq")
```

```
Single term deletions

Model:
oxy.present ~ log.fam + status
        Df Deviance    AIC    LRT Pr(>Chi)   
<none>       58.226 64.226                   
log.fam  1   64.889 68.889 6.6633 0.009842 **
status   1   63.005 67.005 4.7793 0.028803 * 
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
mam.oxy <- m.reduced.oxy.4
summary(mam.oxy)
```

```

Call:
glm(formula = oxy.present ~ log.fam + status, family = binomial(link = logit), 
    data = t.data)

Coefficients:
            Estimate Std. Error z value Pr(>|z|)  
(Intercept)  -2.6897     2.1026  -1.279   0.2008  
log.fam       2.5629     1.1165   2.295   0.0217 *
statusS      -1.7095     0.8607  -1.986   0.0470 *
---
```

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

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 74.031  on 54  degrees of freedom
Residual deviance: 58.226  on 52  degrees of freedom
AIC: 64.226

Number of Fisher Scoring iterations: 5
```

We can test if the effect of `log.fam` differs between dominant and subordinate females by including the interaction using `update()`, and comparing this model to `mam.oxy`. This reveals no evidence for such a difference:

```
m.oxy.int <- update(mam.oxy, . ~ . + log.fam:status)
anova(m.oxy.int, mam.oxy, test="Chisq")
```

```
Analysis of Deviance Table

Model 1: oxy.present ~ log.fam + status + log.fam:status
Model 2: oxy.present ~ log.fam + status
  Resid. Df Resid. Dev Df Deviance Pr(>Chi)
1        51     57.971
2        52     58.226 -1 -0.25426   0.6141
```

5.2 From parameter estimates to probabilities

Let's have a closer look at `summary(mam.oxy)`:

```
summary(mam.oxy)
```

```

Call:
glm(formula = oxy.present ~ log.fam + status, family = binomial(link = logit),
    data = t.data)

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  -2.6897     2.1026  -1.279   0.2008
log.fam       2.5629     1.1165   2.295   0.0217 *
statusS      -1.7095     0.8607  -1.986   0.0470 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 74.031  on 54  degrees of freedom
Residual deviance: 58.226  on 52  degrees of freedom
AIC: 64.226

Number of Fisher Scoring iterations: 5
```

What is the probability that a dominant female with average FAM levels (averaged across all females) is infected? And what about a subordinate female? On a logit scale, these probabilities are:

```
b0 <- mam.oxy$coefficients[1] # intercept
b1 <- mam.oxy$coefficients[2] # log.fam
```

```r
b2 <- mam.oxy$coefficients[3] # status

mean.log.fam <- mean(t.data$log.fam)

logit.p.dominant <- b0 + mean.log.fam*b1
logit.p.subordinate <- b0 + mean.log.fam*b1 + b2
logit.p.dominant
```

```
(Intercept)
   1.768606
```

```r
logit.p.subordinate
```

```
(Intercept)
 0.05909846
```

We can now back-transform `logit.p.dominant` and `logit.p.subordinate` to obtain the probabilities of being infected for both groups of females:

```r
1/(1+exp(-logit.p.dominant))    # dominant
```

```
(Intercept)
  0.8542842
```

```r
1/(1+exp(-logit.p.subordinate)) # subordinate
```

```
(Intercept)
  0.5147703
```

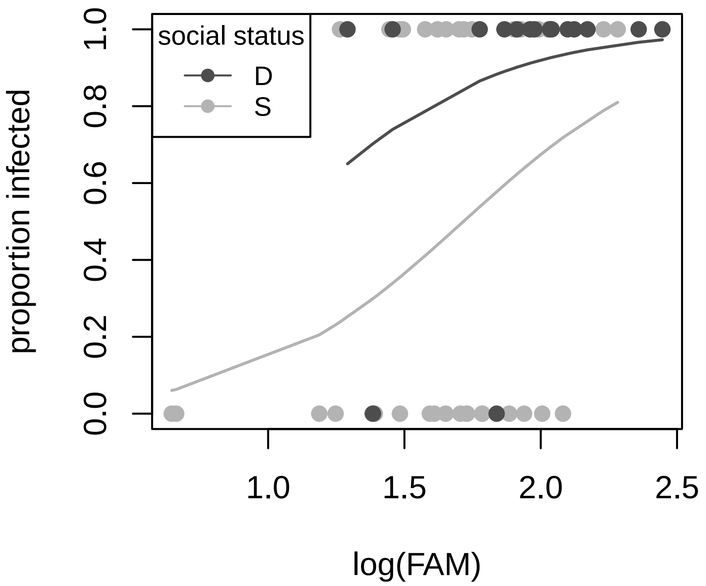
We can use a similar approach to predict infection probabilities for the full range of `log.fam` for both groups, but we could also use `predict()` to obtain predictions on the original scale for both groups of females:

```r
p.predicted <- predict(mam.oxy, type='response')
p.predicted.d <- p.predicted[t.data$status=="D"]
p.predicted.s <- p.predicted[t.data$status=="S"]
```

5.3 Plotting a logistic regression

Now we can use these predictions to create a plot that illustrates the results of the regression of `oxy.present` against `log.fam` for both types of females:

```r
plot(oxy.present ~ log.fam, data=t.data, pch=NA,
     xlab="log(FAM)", ylab="proportion infected")
points(oxy.present ~ log.fam, data=t.data[t.data$status=="S", ], pch=19, col="grey70")
points(oxy.present ~ log.fam, data=t.data[t.data$status=="D", ], pch=19, col="grey30")

lines(p.predicted.s[order(t.data$log.fam[t.data$status=="S"])] ~
        sort(t.data$log.fam[t.data$status=="S"]), lwd=1.5, col="grey70")
lines(p.predicted.d[order(t.data$log.fam[t.data$status=="D"])] ~
        sort(t.data$log.fam[t.data$status=="D"]), lwd=1.5, col="grey30")

legend(x="topleft", legend=c("D", "S"), pch=19,
       col=c("grey30", "grey70"), lwd=c(1,1), title="social status", cex=0.8)
```



EXERCISE 4.11 (HARD)
Especially for lower values of `log.fam`, the regression line for subordinates doesn't look very smooth. Why not? Would you be able to improve this?

6 Is parasite species richness (PSR) a count or a proportion?

In the above analyses, both we and the authors of the paper have treated `psr` as count data, and we therefore assumed a Poisson error distribution. However, is this really appropriate? When counting species in a certain time, space or volume, we usually know which species are there, but we don't know all the species that aren't there. In this case, however, we would seem to know exactly which of the the six possible species are present, and which ones are absent. So could we instead analyse `psr` as a proportion?

EXERCISE 4.12
What do you think, should we treat `psr` as a proportion? What speaks for it? And what speaks against it?

Irrespective of what we *think* makes more sense, let's ask the data which of the two is a better fit. To this end, we can fit a model similar to `m.log.fam`, but this time using `family=binomial(link=logit)`. To specify our dependent variable, the proportion of species observed, we need to provide both the number of species we have observed (`psr`) and the number we did not observe (`6-psr`), and combine both dependent variables with `cbind()`:

```
m.log.fam.prop <- glm(cbind(psr, 6-psr) ~ log.fam,
                      data=t.data, family=binomial(link=logit))
summary(m.log.fam.prop)
```

```
Call:
glm(formula = cbind(psr, 6 - psr) ~ log.fam, family = binomial(link = logit),
    data = t.data)

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  -2.9412     0.6355  -4.628 3.69e-06 ***
log.fam       1.6397     0.3548   4.622 3.80e-06 ***
---
```

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

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 85.365  on 54  degrees of freedom
Residual deviance: 60.754  on 53  degrees of freedom
AIC: 175.48

Number of Fisher Scoring iterations: 4
```

Again we find a highly significant and positive effect of `log.fam` on `psr`. Furthermore, it is worth noting that whereas the Poisson model was *underdispersed*, the residual deviance of our binomial model (60.754) is similar to the residual degrees of freedom (53).

We can also compare the AIC values of both models:

```r
AIC(m.log.fam, m.log.fam.prop)
```

```
               df      AIC
m.log.fam       2 187.7575
m.log.fam.prop  2 175.4831
```

```r
AIC(m.log.fam, m.log.fam.prop)
```

```
               df      AIC
m.log.fam       2 187.7575
m.log.fam.prop  2 175.4831
```

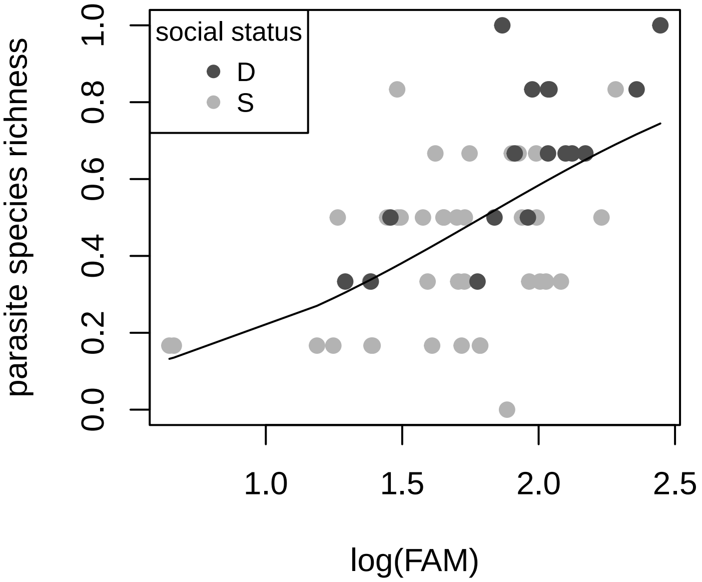
This reveals that of the two models, the model treating `psr` as a proportion is a better fit to the data.

However, because the parameter estimate is on the logit scale this time, we can't compare it directly to the estimate provided by `m.log.fam`. To compare the models further, we should therefore use `m.log.fam.prop` to predict `psr` for all values of `log.fam`. Just like we did above, we first calculate our predictions on the transformed (*i.e.* logit) scale and transform these to the data scale:

```r
b0 <- m.log.fam.prop$coefficients[1]
b1 <- m.log.fam.prop$coefficients[2]
logit.psr.predicted <- b0 + b1*t.data$log.fam
psr.predicted.2 <- 1/(1+exp(-logit.psr.predicted))
```

We can create a similar plot as we did in 4.2, but this time of the proportion of parasites observed against `log.fam`:

```r
plot(psr/6 ~ log.fam, data=t.data, pch=NA,
     xlab="log(FAM)", ylab="parasite species richness")

points(psr/6 ~ log.fam, data=t.data[t.data$status=="S", ], pch=19, col="grey70")
points(psr/6 ~ log.fam, data=t.data[t.data$status=="D", ], pch=19, col="grey30")

lines(psr.predicted.2[order(t.data$fam)] ~ sort(t.data$log.fam))

legend(x="topleft", legend=c("D", "S"), pch=19,
       col=c("grey30", "grey70"), title="social status", cex=0.8)
```



EXERCISE 4.13
Compare this figure to that obtained in 4.2. What are the main differences? Which one do you think provides a better representation of the effect of FAM on PSR?