R workshop #4: GLMs for analysis of categorical and bounded data

Nicola Romanò

Introduction

In previous workshops we have largely dealt with situations where a continuous variable was measured and we wanted to explain its variability as a function of one or more continuous or discrete variables. We have been using variations of the linear model to do so (remember, ANOVA can be also be considered just as a linear model).

However, there are situations where a linear model is not the best solution to use.

Example of these include

- Data where we measure a binary variable (*e.g.* does the subject have diabetes? Yes/No) or a proportion/probability (*e.g.* what are the odds of getting pathology A, depending on variable B?). Both these cases are bounded between 0 and 1 (in the first case the variable can only be 0 or 1) or, if you prefer, 0% and 100%.
- Count data. These are integer numbers, thus have a lower bound at 0 (you cannot count -20 cells!).

Linear models are very powerful, but they are problematic to use with bounded or discrete data, as they assume a continuous range of values that can assume any value from $-\infty$ to $+\infty$.

In this workshop we will see how to overcome some of these issues using generalised linear models (GLMs)[1].

Learning objectives

After completing this workshop you will be able to:

- Describe the concept of GLMs, and of link functions
- Create and interpret the output of GLMs for dealing with discrete and bounded data.

[1] Some people use the acronym GLiMs instead.

A note on χ^2 and Fisher's tests.

As you have seen extensively in lecture 12.3, the easiest way of dealing with count data is that of utilising the χ^2 or Fisher's tests. Please refer to the lecture slides for examples on how to perform these tests in R, using the *chisq.test* or the *fisher.test* functions.

Introduction to generalised linear models (GLMs)

At this point, you should be very familiar with the generic equation for a linear model:

$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_n X_n$

As said above, this equation is not good to represent bounded data, say a proportion or probability that goes from 0 to 1.

Indeed, if you were to model the proportion of patients with an illness depending on a certain parameter X with a linear model you may end up with something like:

$\% patients = 0.02 + 2.5X$

This means that if X is 50 your model will say that 125.02% of patients has the illness, which is not possible. Similarly, if X can take negative value you may find yourself with a negative % of patients which is, again, not possible.

Therefore, we need to introduce some "non-linearity" in the equation above, that allows us to, for instance, constrain our response to between 0 and 1.

Generalised linear model solve this by introducing a "link function" f such that $f(Y)$ is a linear combination of the predictors. Also, these models relax the assumption that residuals are normally distributed (see below).

$$f(Y) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_n X_n$$

For instance, if f is some logarithm, it will constrain the output of the model to positive number, thus imposing a lower bound of 0 to our response[2].

Note that this is still a linear model! Although the relationship between Y and $f(Y)$ as well as between Y and the predictors X_i is not linear, the relationship between $f(Y)$ and X_i is!

There are several link functions that are used in different context. We will only consider two of them in this workshop (the *logit* and the *log* link functions), but the reasoning is very similar for any function you may end up using[3].

[2] The linear models you have use so far use an "identity link function", that is simply defined as $f(x) = x$. You can see how they are a special case of the generalised version we are introducing in this workshop.

[3] Note that we cannot use any arbitrary function, but this is beyond the scope of this course!

Logistic regression

The first type of application of GLM that we are going to use is *logistic regression*[4]. You have already been thought about it in the lectures, it is a type of regression used to model binary (0/1, yes/no) outcomes, as well as percentages/proportions.

For example, we may want to model the odds of an event happening[5] as a function of some variable(s). Since we want to limit the response to between 0 and 1, we model log(odds) instead.

We can write:

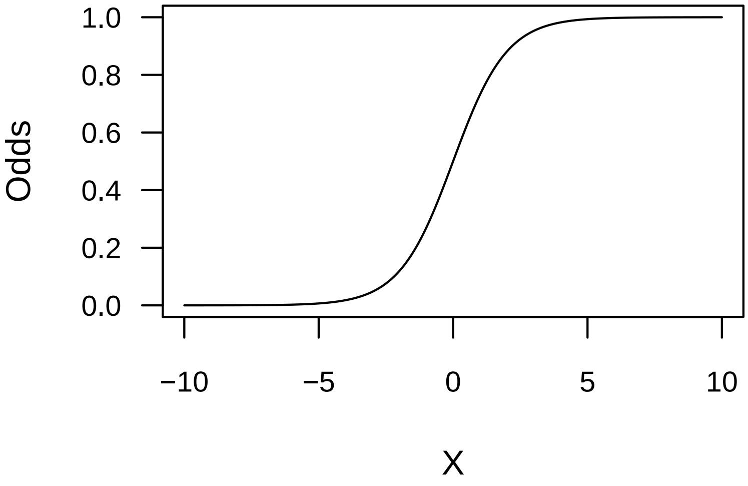
$$log(odds(Y)) = log\left(\frac{p(Y)}{1-p(Y)}\right) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_n X_n$$

with *p(Y)* being the probability of Y happening.

As explained above, this is a GLM; the link function used here (the log odds) is generally called a *logit* link function. We can also rewrite the model in terms of odds of Y, by using the inverse link function[6].

$$\frac{p(Y)}{1-p(Y)} = \frac{e^{\beta_0+\beta_1 X_1+...+\beta_n X_n}}{1+e^{\beta_0+\beta_1 X_1+...+\beta_n X_n}} + \epsilon$$

The logit link function is only defined in the interval (0, 1) and its inverse looks like this:



It is therefore a very good choice to model something that can only be between 0 and 1!

[4] You may also see this referred to as *logit regression*

[5] Remember from the lectures, $odds = \frac{p(X)}{1-p(X)}$

[6] In this case, since the link function is a logarithm, its inverse is the exponential.

Binary data

Let's see a practical example. We start with a binary outcome[7], namely whether babies develop respiratory disease in their first year of life, depending on their Gender and Feeding. In particular, three feeding types are being considered: "Bottle", "Breast", "Suppl".

Start by loading the *babyfood_workshop4.csv* file

```
babyfood <- read.csv("babyfood_workshop4.csv")
babyfood

##   disease nondisease  sex   food
## 1      77        381  Boy Bottle
## 2      19        128  Boy    Mix
## 3      47        447  Boy Breast
## 4      48        336 Girl Bottle
## 5      16        111 Girl    Mix
## 6      31        433 Girl Breast

# We reorder the food factor to have Breast as
# the reference group
babyfood$food <- factor(babyfood$food, levels = c("Breast",
    "Bottle", "Mix"))
```

We can now fit the model using the *glm* function. We specify that the data comes from a binomial distribution[8] and a logit link function[9].

```
model <- glm(cbind(disease, nondisease) ~ sex +
    food, family = binomial(link = logit), data = babyfood)
```

You should be pretty familiar with this notation. We pass both the occurrences of disease and non disease, using *cbind* (column bind) to "stick" the values together into a table with 2 columns.

Let's see the output of our model!

```
summary(model)

##
## Call:
## glm(formula = cbind(disease, nondisease) ~ sex + food, family = binomial(link = logit),
##     data = babyfood)
##
## Deviance Residuals:
##       1        2        3        4        5        6
##  0.1096  -0.5052   0.1922  -0.1342   0.5896  -0.2284
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
```

[7] These data are taken from Payne, 1987, and also analysed in Faraway, 2006

[8] A binomial distribution is good to represent the probability of success in some trial.

[9] Note that logit is the default value, so you can even omit specifying it

```
## (Intercept)  -2.2820     0.1322 -17.259  < 2e-16 ***
## sexGirl      -0.3126     0.1410  -2.216   0.0267 *
## foodBottle    0.6693     0.1530   4.374 1.22e-05 ***
## foodMix       0.4968     0.2164   2.296   0.0217 *
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 26.37529  on 5  degrees of freedom
## Residual deviance:  0.72192  on 2  degrees of freedom
## AIC: 40.24
##
## Number of Fisher Scoring iterations: 4
```

We see that the intercept is different from 0, as this represents the basal odds of disease for the control group (breast-fed boys). We see that there are also significant effects of both gender and food type.

Now, you should be very careful interpreting these coefficients, because remember that we are modelling the *ln(odds)*, so we should exponentiate them to get the odds!

So, for instance, for girls, $\hat{\beta} = -0.3126$

```
exp(-0.3126)
```

```
## [1] 0.7315425
```

This means that being a girl brings the odds of having respiratory disease to 73.2%, compared to the reference level (boys). You can calculate confidence intervals for the estimates using the *confint* function[10]. Remember to exponentiate them so that you can talk about odds, rather than log(odds)!

```
exp(confint(model))
```

```
## Waiting for profiling to be done...
```

```
##                   2.5 %    97.5 %
## (Intercept) 0.07818602 0.1313591
## sexGirl     0.55362089 0.9629225
## foodBottle  1.45028833 2.6441703
## foodMix     1.06463534 2.4926583
```

We can thus say that being a girl reduces the odds of having respiratory disease to 73.2% (95% CI: [55.3, 96.3]) compared to boys. You can interpret the other coefficients in a similar manner.

Finally, the model's summary also reports a measure of *deviance*. This is a goodness-of-fit measure useful for GLMs; in general the lower the deviance, the better.

[10] Alternatively, as seen in the lecture, you can approximate the 95% CIs using $exp(\hat{\beta} \pm 1.96 * SE_{\hat{\beta}})$. For example for $\hat{\beta}_1$ we have $exp(-0.3126 \pm 1.96 * 0.1410)$ giving $[0.5549041, 0.9644088]$, very similar to what calculated by *confint*. Note how these interval are not symmetric, since we are working on a non-linear scale.

The summary reports a Null deviance of 26.38 on 5 degrees of freedom and a Residual deviance of 0.72 on 2 degrees of freedom. The null deviance refers to the intercept-only model (essentially a null model where we say that neither Sex or Feeding have an effect on the odds of disease). Since we have 6 observations, that null modle has 5 degrees of freedom. Our current model adds 3 variables (1 dummy for Gender, 2 dummies for food), thus has only 2 degrees of freedom, but has a much reduced variance, indicating that our model fits the data much better than an intercept-only model!

As we have seen in a previous workshop, we can use the *drop1* function to see the contribution of each model parameter.

```
drop1(model)
```

```
## Single term deletions
##
## Model:
## cbind(disease, nondisease) ~ sex + food
##        Df Deviance    AIC
## <none>      0.7219 40.240
## sex     1   5.6990 43.217
## food    2  20.8992 56.417
```

Not surprisingly, we see that removing either Sex or Food from the model results in an increased deviance (and an increased AIC, another goodness-of-fit measure for which, again, the lower the better.)

Obviously, when looking at this type of data we always need to be very aware that many other confounding factors (e.g. socioeconomic status) may be important to consider.

Percentages

The same reasoning applies for datasets where we have measured a probability, or a percentage.

For instance, let's load the file *smoking_workshop4.csv*. This contains survival data[11] from 24321 male UK doctors born between 1900 and 1930, in relation to whether they are smokers or not (this only includes life-long smokers).

```
smoking <- read.csv("smoking_workshop4.csv")
head(smoking)
```

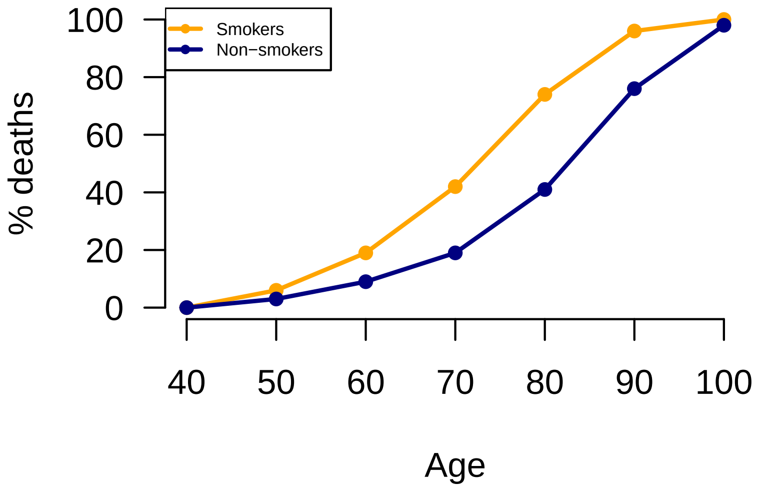
```
##   Age Smoker Alive Dead
## 1  40      N   100    0
## 2  50      N    97    3
## 3  60      N    91    9
```

[11] From Doll, 2004

```
## 4  70      N    81   19
## 5  80      N    59   41
## 6  90      N    24   76
```

Just as before, we can fit a GLM.

Try plotting the data, for example I have got this graph[12].



What would you conclude from looking at the data?

Let's now fit a GLM to these data

```
smoking$AgeAdj <- smoking$Age - 40

model.2 <- glm(cbind(Dead, Alive) ~ AgeAdj + Smoker,
    family = binomial(link = "logit"), data = smoking)

summary(model.2)
```

```
##
## Call:
## glm(formula = cbind(Dead, Alive) ~ AgeAdj + Smoker, family = binomial(link = "logit"),
##     data = smoking)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.5589  -0.7459   0.3528   1.2303   1.9902
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -5.709451   0.311152 -18.349  < 2e-16 ***
## AgeAdj       0.140632   0.007226  19.461  < 2e-16 ***
## SmokerY      1.305192   0.180313   7.238 4.54e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
```

[12] Remember to share your code on the forum!

```
##
##     Null deviance: 1024.724  on 13  degrees of freedom
## Residual deviance:   19.996  on 11  degrees of freedom
## AIC: 71.108
##
## Number of Fisher Scoring iterations: 4
```

You will note that rather than Age I have modelled Age - 40; this will only influence the intercept, making it easier to interpret. It will not influence the other coefficients[13].

The intercept is ~ -5.7. This represent the basal log(odds) of dying for someone at the reference level (non smoker) and at AgeAdj = 0. Since AgeAdj = Age - 40, the intercept shows the basal log(odds) for a 40 year old non smoker[14].

So, the odds of dying for a 40 year old non smoker are

```
exp(-5.709451)
```

```
## [1] 0.003314492
```

Remember, these are odds, so they are $\frac{P(dying)}{1-P(dying)}$; this value is very low, representative of the fact that all of the subjects were alive at age 40.

We can also see a strong effect of Age on probability of dying[15], and also a significant effect of smoking. In particular, smoking increases the odds of dying by

```
exp(1.305192)
```

```
## [1] 3.688397
```

The coefficient for Age is interpreted as the log odds-ratio for 1 year age difference.

That is: $\frac{odds(dying, age\ x+1)}{odds(dying, age\ x)}$, where odds are defined as above.

Finally, we can graphically check that our model fits the data correctly.

We can ask the model to predict the values at different ages for smokers and non smokers. We use the *predict* function for this. This function wants a list with elements named as the parameters of the model.

For example, if we wanted to predict the log odds of dying for smokers and non smokers from 40 to 100 years old in steps of 1 year we could do the following:

```
pred.age <- 40:100
```

[13] Try it by yourself! See what happens when you use Age instead. If this is confusing, try to do it on a simple linear model, it will be more intuitive there.

[14] If we modelled Age and not AgeAdj, the intercept would refer to 0 year old, which is probably less interesting.

[15] I guess we didn't really need a model to say that!

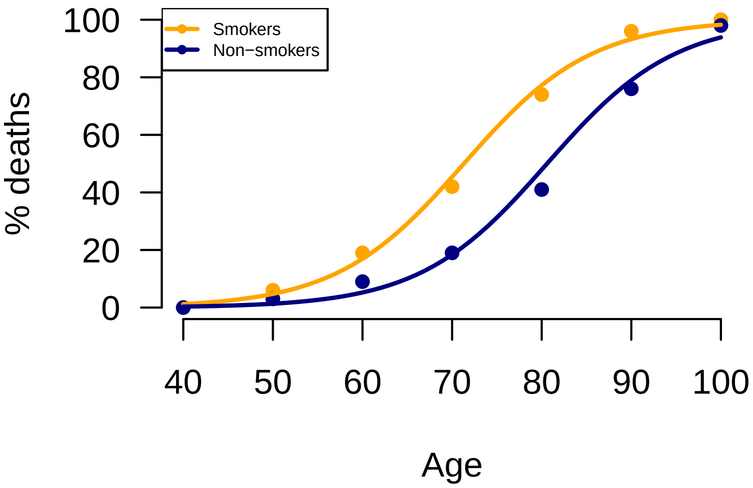
```
smokers <- list(AgeAdj = pred.age - 40, Smoker = rep("Y",
    length(pred.age)))

nonsmokers <- list(AgeAdj = pred.age - 40, Smoker = rep("N",
    length(pred.age)))
```

We can now use *predict* to ask the model what the log(odds) would be for these new data points[16].

```
pr.smokers <- predict(model.2, type = "response",
    newdata = smokers) * 100
pr.nonsmoker <- predict(model.2, type = "response",
    newdata = nonsmokers) * 100
```

We can now plot the prediction on top of our data, showing that the model works extremely well!



So... can I use linear regression instead?

As explained above, that is probably a bad solution. Let's see what happens if we use *lm*.

```
# Proportion of dead subjects
smoking$PercDead <- smoking$Dead/(smoking$Dead +
    smoking$Alive)
model.lm <- lm(PercDead ~ AgeAdj + Smoker, data = smoking)
```

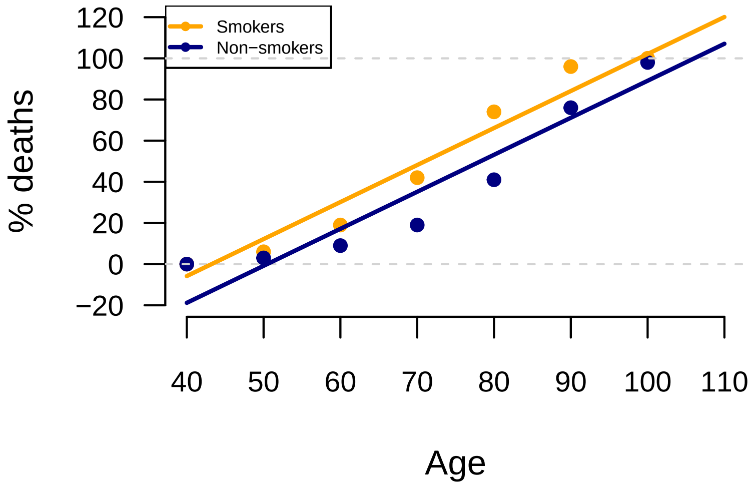
We model the percentage of dead subject against Age - 40 and Smoker.

[16] The `type = "response"` parameter gives us prediction in terms of odds, rather than log(odds). If you omit it you will have to exponentiate the results. In general, it will allow you to see the prediction of the model in terms of Y rather than *f(Y)*, where *f* is the link function.

```
summary(model.lm)
```

```
##
## Call:
## lm(formula = PercDead ~ AgeAdj + Smoker, data = smoking)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.161429 -0.076652  0.008661  0.073571  0.188036
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.188036   0.061884  -3.038   0.0113 *
## AgeAdj       0.017982   0.001501  11.981 1.18e-07 ***
## SmokerY      0.130000   0.060037   2.165   0.0532 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1123 on 11 degrees of freedom
## Multiple R-squared:  0.9309, Adjusted R-squared:  0.9184
## F-statistic: 74.11 on 2 and 11 DF,  p-value: 4.136e-07
```

We can already see that we have a negative intercept meaning that, at age 40… -18% of patients are dead! Plotting model predictions shows that the model does not a good job, especially for non-smokers.



Similarly, consider a 120 year old smoker. What is the probability of he being dead?

```
# From the logistic regression
predict(model.2, list(AgeAdj = 80, Smoker = "Y"),
    type = "response")
```

```
##         1
## 0.9989377
```

```
# From the linear regression
predict(model.lm, list(AgeAdj = 80, Smoker = "Y"))
```

```
##        1
## 1.380536
```

So, the logistic regression tells us that the odds of the patient being dead are 99.89%, while the linear model predicts a value of 138%!

In summary, linear regression is not a good choice to model binary data or percentages.

Count data

Finally, we are going to see an example of model of counts. These are often modelled using what is called *count regression*, or *Poisson regression*.

This is done with a GLM modelling Poisson data and a log link function[17], simply obtained by specifying `family=poisson(link=log)` in the call to *glm*.

This means modelling

$$log(Y) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_n X_n$$

Let's consider the data in *lizards-workshop4.csv*. This shows the counts of three species of lizards (A, B, and C) in three different locations (Loc1 to Loc3). For each location lizards were counted in three different plots of land.

```
lizards <- read.csv("lizards-workshop4.csv")
summary(lizards)
```

```
##  Location   Plot   Species     Count
##  Loc1:12   P1:9   A:12    Min.   : 1.00
##  Loc2:12   P2:9   B:12    1st Qu.:13.50
##  Loc3:12   P3:9   C:12    Median :22.00
##            P4:9           Mean   :20.06
##                           3rd Qu.:27.25
##                           Max.   :37.00
```

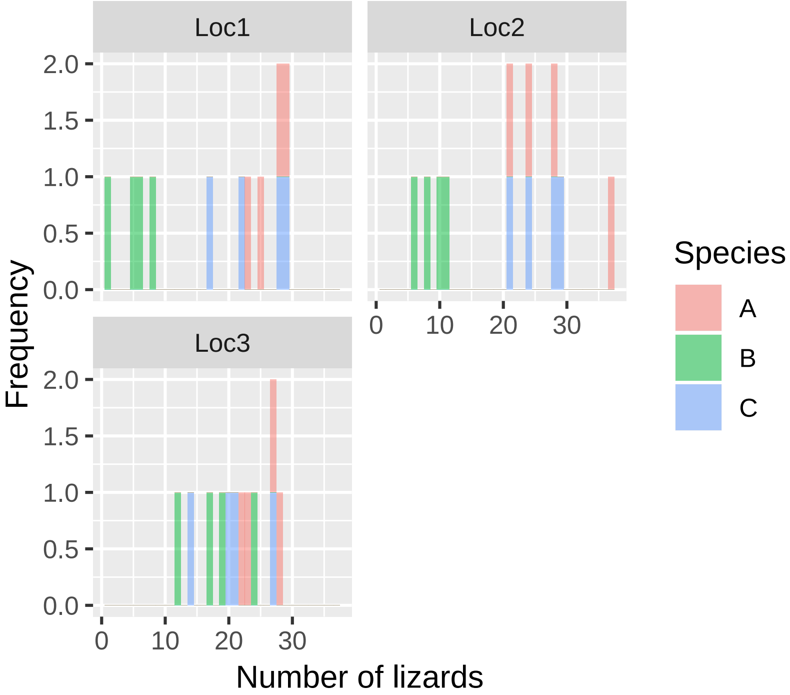
```
head(lizards)
```

```
##   Location Plot Species Count
## 1     Loc1   P1       A    28
## 2     Loc1   P1       B     1
## 3     Loc1   P1       C    28
## 4     Loc1   P2       A    29
```

[17] Just as above, we choose a Poisson distribution because it is good to model counts, as it is a discrete distribution, and the log link to limit the output to $Y > 0$. Note that the Poisson distribution is not always the best choice for counts, other options are available. Specifically, you may want to avoid Poisson regression in cases of large numbers of zeroes in your data (zero-inflated distributions are better suited to that) or in case of overdispersion of the data (negative binomial is better suited to this case).

```
## 5     Loc1   P2       B     5
## 6     Loc1   P2       C    22
```

We can start by plotting the data:



It looks like in locations 1 and 2, species A and C are in similar numbers, higher than species B. However, in Location 3, all three species seem to have a similar frequency.

This is not an obvious situation to analyse, let's see how to use a GLM to model it! For simplicity, we will consider plots as independent, although you should have spotted that this is a nested design, therefore the random effect from the plot should, in theory, be accounted for! You can indeed create a mixed-effect GLM[18], but we will not cover that here, so I leave that to your curiosity!

We start by creating the GLM. Since we noted a clear Species/Location interaction, we add that to our model

```
model.3 <- glm(Count ~ Species * Location, data = lizards,
    family = poisson(link = log))
```

```
summary(model.3)
```

```
##
## Call:
## glm(formula = Count ~ Species * Location, family = poisson(link = log),
##     data = lizards)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.18658  -0.62152   0.04753   0.54296   1.71993
##
```

[18] For instance, using the *glmm* function in the *glmm* package or the *glmer* function in the *lme4* package

```
## Coefficients:
##                        Estimate Std. Error z value Pr(>|z|)
## (Intercept)             3.26767    0.09759  33.484  < 2e-16 ***
## SpeciesB               -1.65823    0.24397  -6.797 1.07e-11 ***
## SpeciesC               -0.08961    0.14121  -0.635   0.5257
## LocationLoc2            0.04652    0.13644   0.341   0.7331
## LocationLoc3           -0.04879    0.13973  -0.349   0.7270
## SpeciesB:LocationLoc2  0.51310    0.31174   1.646   0.0998 .
## SpeciesC:LocationLoc2  0.01410    0.19707   0.072   0.9429
## SpeciesB:LocationLoc3  1.32972    0.28881   4.604 4.14e-06 ***
## SpeciesC:LocationLoc3 -0.10884    0.20527  -0.530   0.5960
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 157.26  on 35  degrees of freedom
## Residual deviance:  29.35  on 27  degrees of freedom
## AIC: 216.13
##
## Number of Fisher Scoring iterations: 4
```

This is quite a complex output. Let’s decipher it! First of all, remember what we are modelling:

$log(Counts) = \beta_0 + \beta_1 * SpeciesB + \beta_2 * SpeciesC + \beta_3 * Location2 + \beta_4 * Location3 + (\text{interactions, with coefficients } \beta_5\text{to } \beta_8)$

Where SpeciesB and SpeciesC are the two dummy variables used to represent the three-level factor Species and and Location2 and Location3 are the two dummy variables used to represent Locations.

Thus, $\hat{\beta}_0$ is the log(mean counts) for the basal level (Species A in location 1).

Indeed, if we check the mean manually with:

```
mean(lizards$Count[lizards$Location == "Loc1" &
    lizards$Species == "A"])
```

```
## [1] 26.25
```

We can see that the model approximates it pretty well!

```
exp(3.26767)  # exp(beta1)
```

```
## [1] 26.25011
```

You can interpret the other coefficients in a similar way. For example $\beta_{SpeciesC} = -0.08961$ tells us that the effect of species C is to decreases the counts to $e^{-0.08961} \approx 0.91 \approx 91\%$ of the reference level. Again, we can calculate 95% CIs using *confint*.

```
exp(confint(model.3))
```

```
## Waiting for profiling to be done...
```

```
##                            2.5 %     97.5 %
## (Intercept)           21.5438454 31.5958139
## SpeciesB               0.1147362  0.3001501
## SpeciesC               0.6925339  1.2056592
## LocationLoc2           0.8017074  1.3697220
## LocationLoc3           0.7236934  1.2525673
## SpeciesB:LocationLoc2  0.9144204  3.1205880
## SpeciesC:LocationLoc2  0.6890984  1.4929051
## SpeciesB:LocationLoc3  2.1800599  6.7938404
## SpeciesC:LocationLoc3  0.5993073  1.3409254
```

So for SpeciesC the counts are 91% (CI = (69.2%, 120%)) of the reference level.

We can also see that there is a significant interaction of Species B and Location 3. This is not unexpected. Interpreting interaction coefficients is always tricky, but luckily we can use our trusted friend *emmeans*!

```
library(emmeans)
marginals <- emmeans(model.3, ~Species * Location)
pairs(marginals, by = "Species", type = "response")
```

```
## Species = A:
##  contrast        ratio         SE  df z.ratio p.value
##  Loc1 / Loc2 0.9545455 0.13023414 Inf  -0.341  0.9379
##  Loc1 / Loc3 1.0500000 0.14671401 Inf   0.349  0.9350
##  Loc2 / Loc3 1.1000000 0.15198684 Inf   0.690  0.7695
##
## Species = B:
##  contrast        ratio         SE  df z.ratio p.value
##  Loc1 / Loc2 0.5714286 0.16017155 Inf  -1.996  0.1130
##  Loc1 / Loc3 0.2777778 0.07021004 Inf  -5.068  <.0001
##  Loc2 / Loc3 0.4861111 0.10016757 Inf  -3.501  0.0013
##
## Species = C:
##  contrast        ratio         SE  df z.ratio p.value
##  Loc1 / Loc2 0.9411765 0.13383446 Inf  -0.426  0.9047
##  Loc1 / Loc3 1.1707317 0.17604546 Inf   1.048  0.5464
##  Loc2 / Loc3 1.2439024 0.18449653 Inf   1.471  0.3047
##
## P value adjustment: tukey method for comparing a family of 3 estimates
## Tests are performed on the log scale
```

As expected, the only statistically significant pairwise ratio is between location 1 and 3 for species B. The estimate is 0.28, meaning that counts for species B in Loc3 are about $1/0.28 \approx 3.6$ times the counts for species B in location 1 (or that the counts in location 1 are approximately 28% of those in location 3)[19].

This workshop should have given you the basic tools to analyse binary, proportion, and count data. As always, we are only scratching the surface here, but this should be quite a good start, and if you are interested in these topics there is a lot to be found! This is the most advanced type of linear model that we are going to look at this year. Next semester we will look at classification and prediction models, as well as some more advanced statistical techniques.

[19] A similar result can be obtained directly by summing the exponentiated $\hat{\beta}$