Practicals

Olivier Gimenez

Practical #1

1. Use simulation to generate a data set of 100 covariate values from $X \sim \text{Normal}(20, \sigma = 2)$. Use a histogram to explore the distribution of X.

2. Generate a Poisson rate from these values according to the relationship lambda=exp(-3+0.2*X). Have a look graphically to the relationship between λ and X. Hint: You might want to order the values of X using the R function `sort()`.

3. Use simulation to generate a data set of 100 response values from a Poisson distribution according to this rate $Y \sim \text{Poisson}(\lambda(X))$. Explore the distribution of Y using a bar plot (`?barplot`).

4. Construct a data frame containing the covariate (X) and response (Y) data. Have a look to the first and last rows.

5. Use a GLM with the appropriate structure to try and retrieve the parameters -3 and 0.2 from question 2.

6. Increase sample size (multiply by 1000) and comment on the parameter estimates.

7. Overdispersion. We would like to do the same exercise with an overdispersed Poisson distribution. To do so, we need to generate data from a quasi-Poisson distribution. Interestingly, the negative-binomial (NB) distribution allows relaxing the Poisson assumption $E(Y) = V(Y) = \lambda$. There are several parameterizations for the NB distribution. Here, we will use $W \sim NB(\lambda, \phi)$ where λ is the expected value as in the Poisson distribution and ϕ is the overdispersion parameter. From the NB properties, we have that the mean of W is λ while its variance is $\lambda + \lambda^2/\phi$. Therefore, a small value of ϕ means a large deviation from a Poisson distribution (the variance of W is much larger than λ, which would also be the mean for a Poisson distribution), while as ϕ gets larger the NB looks more and more like a Poisson distribution (the term λ^2/ϕ tends to 0, and the mean and variance looks more and more alike). In R, we will specify:

```
lambda = 2
phi = 5
n = 1000

# simulate the response
w_nb <- rnbinom(n,size=phi,mu=lambda)

# recover the NB parameters from its mean and variance
mean(w_nb) # lambda
```

```
## [1] 1.994
```

```r
mean(w_nb)^2/(var(w_nb)-mean(w_nb)) # phi
```

```
## [1] 5.517072
```

Adopt the same approach as above to simulate data from a Poisson distribution with overdispersion parameter $\phi = 0.1$. Inspect the residuals. Do they look ok to you? If not, fit a GLM with an appropriate structure. Have a look to the relationship between the response and the predictor for both models, without and with the overdispersion parameter (use the `visreg` from the package `visreg`). Is there any difference?

Practical #2

1. Relationship between tree diameter and height and LMM. You have data on 1000 trees from 10 plots, with 4 up to 392 trees per plot and several measurements taken on each tree.

```r
# dbh is diameter at breast height (diameter of the trunk)
trees <- read.table("trees.txt", header=TRUE)
str(trees)
```

```
## 'data.frame':    1000 obs. of  6 variables:
##  $ plot  : int  2 4 5 2 4 4 1 5 1 2 ...
##  $ dbh   : num  38.9 26.1 42.7 20.7 21.8 ...
##  $ height: num  37.8 38.1 50.2 30.1 34 21.9 18.2 35.8 35.5 41 ...
##  $ sex   : chr  "female" "female" "female" "female" ...
##  $ dead  : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ dbh.c : num  13.85 1.05 17.66 -4.28 -3.17 ...
```

```r
head(trees)
```

```
##   plot   dbh height    sex dead  dbh.c
## 1    2 38.85   37.8 female    0  13.85
## 2    4 26.05   38.1 female    0   1.05
## 3    5 42.66   50.2 female    0  17.66
## 4    2 20.72   30.1 female    0  -4.28
## 5    4 21.83   34.0 female    0  -3.17
## 6    4  8.23   21.9   male    0 -16.77
```

Fit a linear model with height as the response variable and dbh as a predictor. Plot the data on a graph, and add the regression line (use `abline`).

Now we have only one intercept. What if allometry varies among plots? Fit a linear model with a different intercept for each plot, and inspect the results with `broom::tidy`. Try and represent the 20 regression lines on the same graph.

How many parameters are estimated in the model pooling all plots together vs. the model considering a different intercept for each plot?

Mixed models enable us to make a compromise between the two models by accounting for plot-to-plot variability.

Recall that:

$$
\begin{aligned}
y_{ij} &\sim N(\mu_{ij}, \sigma^2) \\
\mu ij &= a_j + b x_i \\
a_j &\sim N\left(\mu_a, \tau^2\right)
\end{aligned}
$$

How does this GLMM formulation translate into in our example? Fit model with plot as a random effect. Comment on the outputs. Visualise the effect using `allEffects` and/or `visreg`.

2. Longitudinal study on coral reef and GLMM. A survey of a coral reef uses 10 predefined linear transects covered by divers once every week. The response variable of interest is the abundance of a particular species of anemone as a function of water temperature. Counts of anemones are recorded at 20 regular line segments along the transect. The following piece of code will generate a data set with realistic properties according to the above design. Make sure you understand what it is doing. You might want to explain the script to the colleague next to you. Also, to try and make sense of the code of others, it is always good to plot and/or run small sections of the code.

```
transects <- 10
data <- NULL
for (tr in 1:transects){
  ref <- rnorm(1,0,.5) # random effect (intercept)
  t <- runif(1, 18,22) + runif(1,-.2,0.2)*1:20 # water temperature gradient
  ans <- exp(ref -14 + 1.8 * t - 0.045 * t^2) # Anemone gradient (expected response)
  an <- rpois(20, ans) # Actual observations: counts on 20 segments of the current transect
  data <- rbind(data,cbind(rep(tr, 20), t, an))
}
```

Generate a data set using the anemone code and fit to it the following mixed effects models:

- m1: GLM with temperature
- m2: GLM with quadratic formula in temperature
- m3: GLMM with temperature and random intercept
- m4: GLMM with quadratic temperature and random intercept

Carry out model selection from this set of models. Inspect the temperature effect with `visreg`.