Tutorial #8: Count models - Negative Binomial GLM

Guy F. Sutton

Centre for Biological Control, Rhodes University

8.1. Analysis of count (abundance) data

In the previous tutorial, we covered the use of the Poisson GLM - a natural starting point for the analysing of count/abundance data. However, many abundance datasets contain much more variation than expected under the Poisson distribution, and as such, we must account for this extra variation in our model. The way to do this is by using a negative binomial GLM - which expects more variation than expected under the Poisson distribution.

8.2. Negative binomial GLM

Let's assume we have the same experimental design as for the Poisson GLM in the previous tutorial, whereby we counted the number of fish per quadrat (`no_fish_quadrat`) (i.e. a measure of abundance) at each of three different sites (`site`). We want to know whether abundances differ between the 3 sites? I have slightly altered the raw data since the previous tutorial.

```
## Rows: 600
## Columns: 2
## $ no_fish_quadrat <dbl> 3, 5, 0, 5, 5, 5, 2, 7, 7, 5, 7, 2, 9, 0, 3, 1, 5, ...
## $ site            <chr> "A", "A", "A", "A", "A", "A", "A", "A", "A", "A", "...
```

Fitting a Poisson GLM

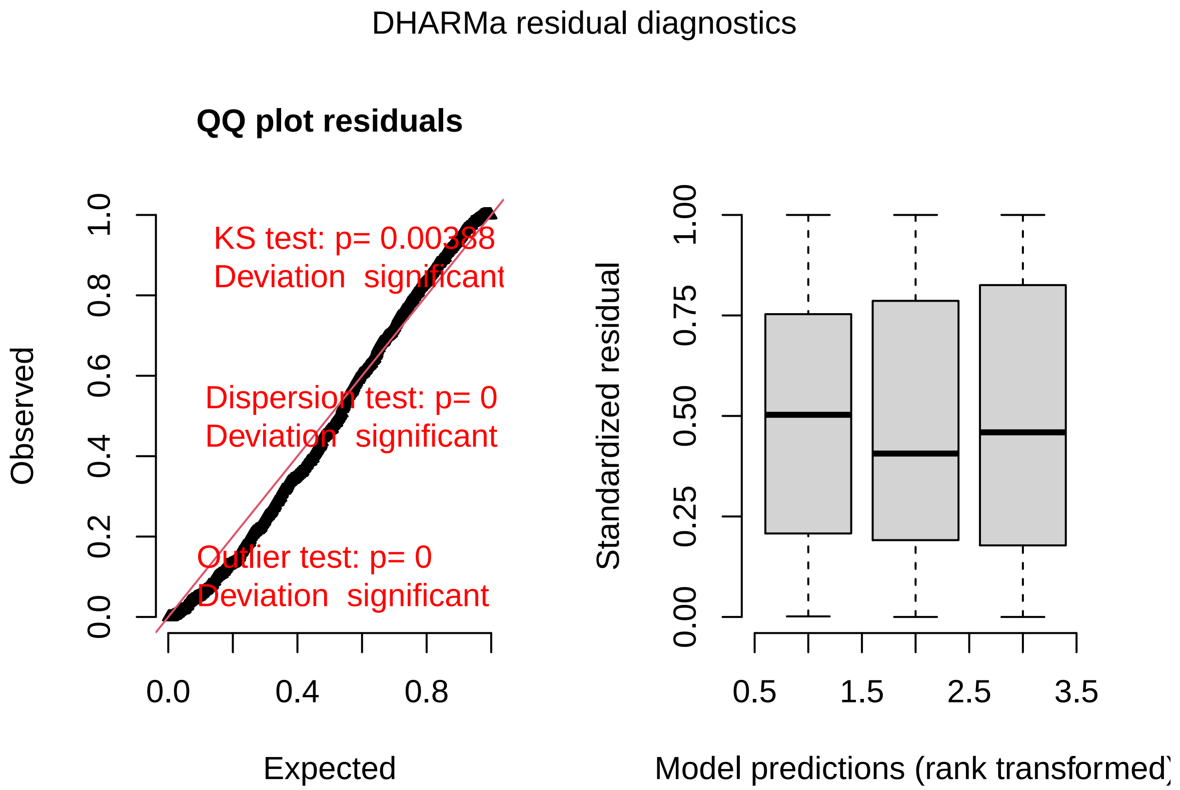
Naturally, we should first try a Poisson GLM to our data because we are dealing with count data, and evaluate whether it was a good fit or not.

```r
mod1 <- glm(no_fish_quadrat ~ site,
            data = data,
            family = poisson(link = "log"))
```

And now we should evaluate the fit:

```r
mod1_diagnostis <- DHARMa::simulateResiduals(mod1)
plot(mod1_diagnostis)
```

```
## DHARMa:plot used testOutliers with type = binomial for computational reasons (nObs > 500). Note that
```



Clearly, the Poisson GLM was not a good fit to our data. The Kolmogorov-Smirnoff Test (KS-test) was significant ($P = 0.004$), telling us that our data do not conform to expectations under the Poisson distribution.

Fitting a negative binomial GLM

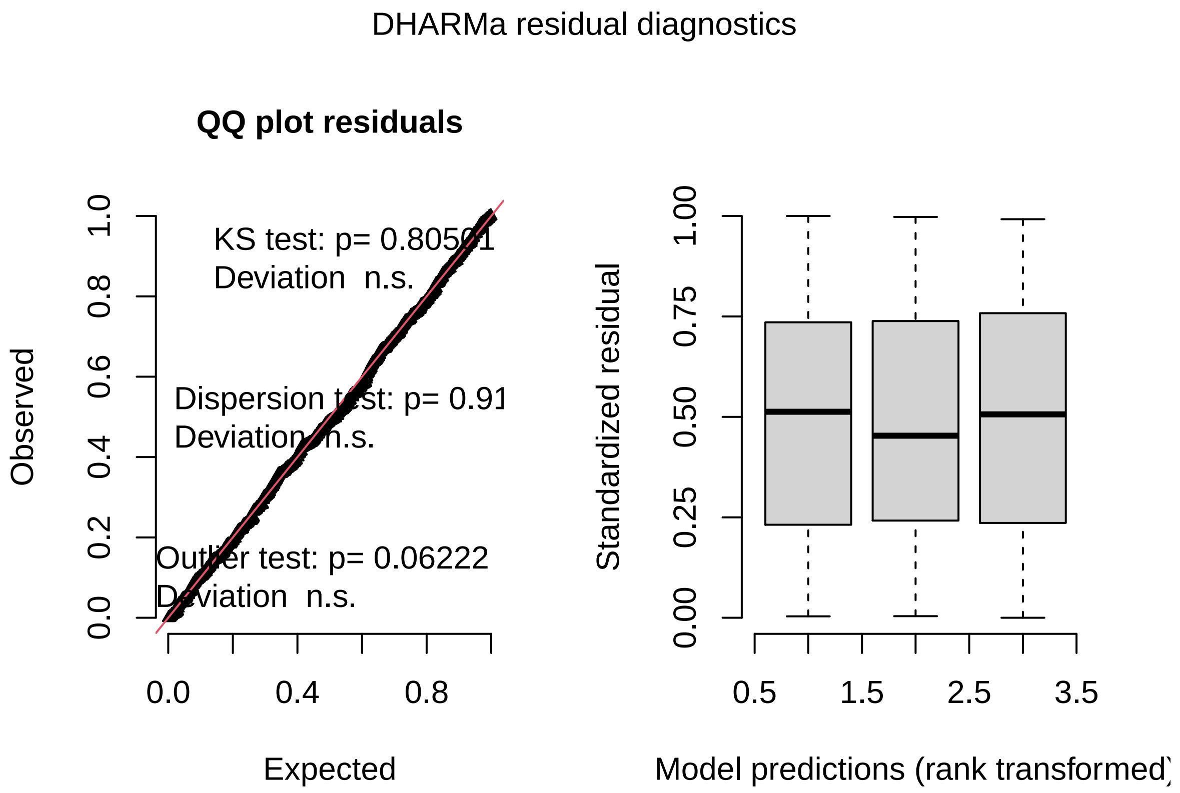
To fit a negative binomial GLM, we cannot use the `glm` function that we have used for all our models so far. We will need to use the `glmmTMB` function from the package with the same name. Specifying our response and predictor variables is exactly the same (`no_fish_quadrat ~ site`). We need to change the `family` argument to `nbinom2` to tell `R` to fit a negative binomial GLM.

```r
# Fit a negative binomial GLM
mod1 <- glmmTMB::glmmTMB(no_fish_quadrat ~ site,
                         data = data,
                         # Fit neg.bin (nbinom2)
                         family = nbinom2)
```

Evaluate model fit

Evaluating the fit of a negative binomial (actually any GLM's) is exactly the same as for the Gaussian and Poisson GLM's that we have already covered.

```
mod1_diagnostis <- DHARMa::simulateResiduals(mod1)
plot(mod1_diagnostis)
```



LHS plot (QQplot): This plot tells us whether our data conforms to a negative binomial data distribution. If our GLM is a good fit to the data, the white triangles will fall approximately on the red 1:1 line and the KS test P-value will be greater than 0.05. The Kolmorogorov-Smirnoff test (KS test) evaluates whether our data follow the distribution we specified in the `family` argument in the GLM call above (remember: we said `family = nbinom2`).

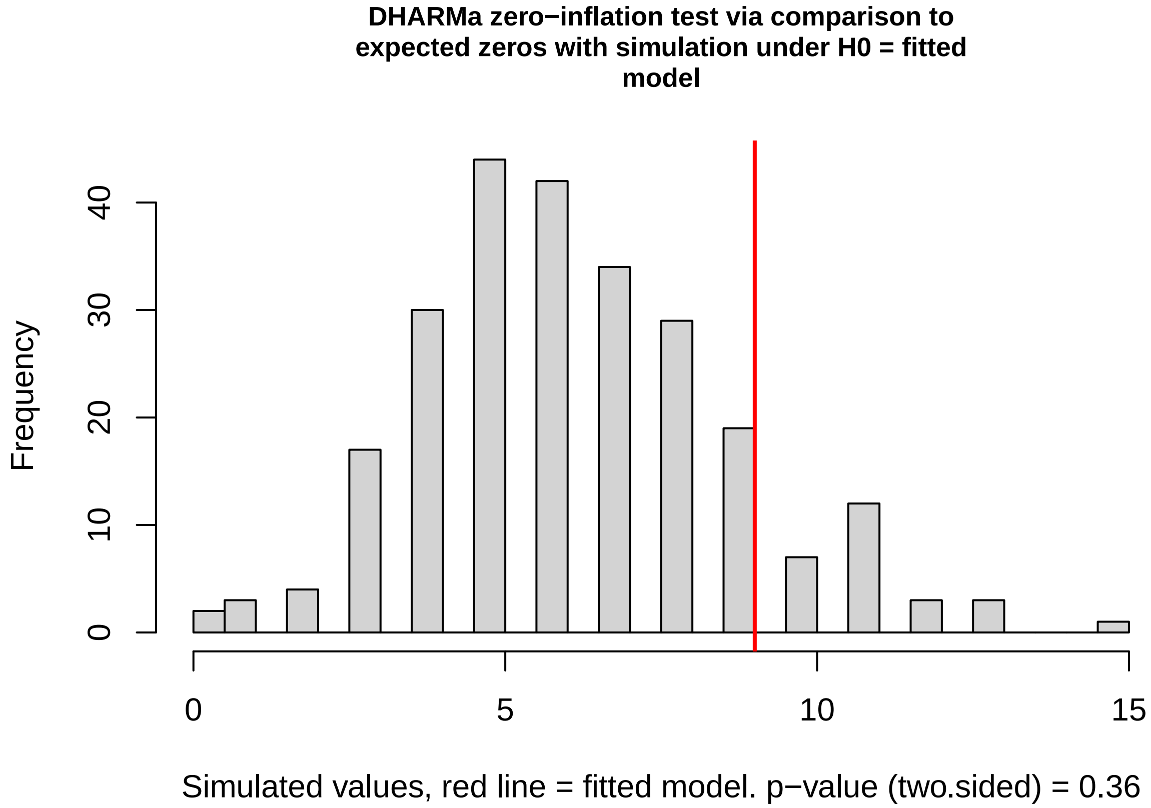
- Interpreting this plot tells us that the negative binomial data distribution was a good fit. Note that the points fall approximately along the 1:1 red line and the KS test result was not significant - which tells us that our model approximates a negative binomial distribution well.

RHS plot (Fitted vs Residuals): This plot tells us whether there are issues with homogeneity of variances. GLM's, like ANOVA and linear regression, assume that the variance in the different groups, or across the range of x values, are approximately equal. We want to see the bold black lines for the different groups fall approximately at y = [0.25, 0.50, 0.75]. If there are issues, the bold lines will fall away from the y = [0.25, 0.50, 0.75], and all the lines will be red (not black).

- Interpreting this plot we can see that the bold lines fall approximately on the y = [0.25, 0.50, 0.75] lines, as would be expected. There is a no evidence for a deviation from equal variances.

Testing for zero-inflation Count data often contains a large amount of zeroes. Anyone who has done fieldwork counting animals or plants will be acutely aware of this - I spent most of my PhD counting no insects during each of my surveys. We can specifically test for whether our data has too many zeroes than expected under the negative binomial distribution (this is called 'zero-inflation' in the literature).

```
# Test for zero-inflation
DHARMa::testZeroInflation(mod1)
```



```
##
##  DHARMa zero-inflation test via comparison to expected zeros with
##  simulation under H0 = fitted model
##
## data:  simulationOutput
## ratioObsSim = 1.4178, p-value = 0.36
## alternative hypothesis: two.sided
```

If there are too many zeros in our data, the p-value will be significant (< 0.05).

- This example tells us that there aren't too many zeros in our dataset (P = 0.36).

Take home: The GLM that we fit was a good fit to the data. We can proceed with looking and the results.

Results

Now we get to the bit we are interested in: assessing statistical significance and calculating p-values. To do this, we will use a Likelihood Ratio Test (LRT). When we only have 1 predictor variable (here: `site`), we need to calculate p-values using type I sum-of-squares (SOS). SOS's are just different ways that we ask `R` to calculate p-values.

PLEASE DO NOT USE SUMMARY() - THIS WILL PRODUCE THE WRONG P-VALUES WHEN YOU HAVE MORE THAN 1 PREDICTOR VARIABLE

Parameter significance Because we are using `glmmTMB` not the usual `glm` we used in previous tutorials, we need to change the `test` argument below to `test = "Chisq` to calculate parameter significance for the negative binomial GLM.

```r
# Perform LRT  with type I sum-of-squares
car::Anova(mod1,
           # Specify we want a Likelihood Ratio Test
           test = "Chisq")
```

```
## Analysis of Deviance Table (Type II Wald chisquare tests)
##
## Response: no_fish_quadrat
##       Chisq Df Pr(>Chisq)
## site 834.49  2  < 2.2e-16 ***
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
```

So, the LRT tells us that the number of fish we counted per quadrat (`no_fish_quadrat`) was significantly different between sites (`site`) (χ^2 = 834.49, d.f. = 1, P < 0.001).

***Post hoc* analysis** We often want to determine where the significant differences occur that we have identified above. To do this, we can perform *post hoc* tests. Below, we will perform Tukey's *post hoc* comparisons, which is arguably the most common and widely used test. We have to feed in the model name (`mod1`), and then tell `R` that we can't pairwise comparisons amongst our predictor variable (here: `site`).

```r
emm1 <- emmeans(mod1,
                specs = pairwise ~ site,
                adjust = "tukey")

# Get 95% confidence intervals
emm1$contrasts %>%
  summary(infer = TRUE)
```

```
##  contrast estimate     SE  df lower.CL upper.CL t.ratio p.value
##  A - B      -1.157 0.0479 596   -1.270  -1.0447 -24.170 <.0001
##  A - C      -1.333 0.0472 596   -1.444  -1.2220 -28.246 <.0001
##  B - C      -0.176 0.0366 596   -0.262  -0.0897  -4.799 <.0001
##
## Results are given on the log (not the response) scale.
## Confidence level used: 0.95
## Conf-level adjustment: tukey method for comparing a family of 3 estimates
## P value adjustment: tukey method for comparing a family of 3 estimates
```

So, we can see that the number of fish recorded per quadrat was higher at `siteB` than `siteA` ($P < 0.001$), higher at `siteC` than `siteA` ($P < 0.001$), and higher at `siteC` than `siteB` ($P < 0.001$).