# Lab 7 - Instructions & partial code

Franz Mueter

October 3, 2021

For this lab, I provided partial code. Please complete all exercises and provide EITHER a knitted pdf or Word file if you use R markdown, OR paste your code for the exercises only, with answers to questions, into the text editor in Canvas (don't attach R script, please)

## Lab 7: Generalized Linear Models

### Poisson and negative binomial regression: Density of salamanders

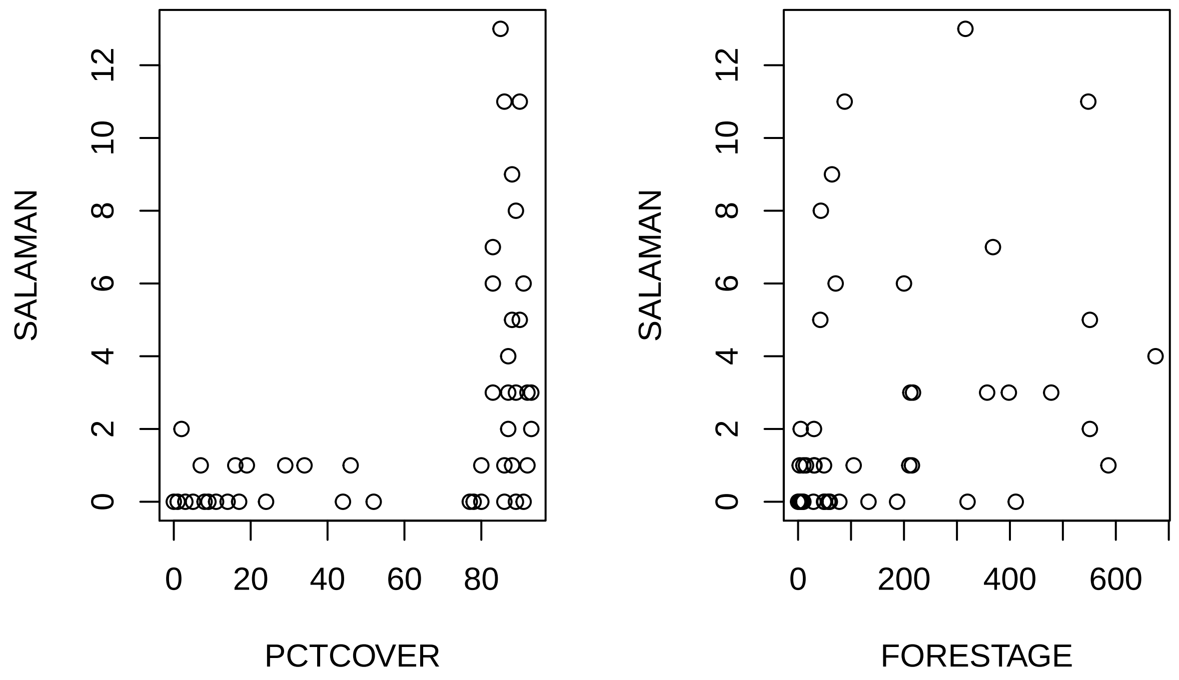
*Example from F.L. Ramsey and D.W. Schaefer 2002. The Statistical Sleuth: A course in Methods of Data Analysis. Duxbury, Pacific Grove, CA*

We will examine results from an observational study in which the number of salamanders in different locations was counted. The goal was to predict salamander density at various locations based on forest characteristics. The response variable (salamander counts per unit area) is modeled as a Poisson random variable as a function of several explanatory variables. Salamander count takes on small integer values. The explanatory variables percent canopy cover (PCTCOVER) and forest age (FORESTAGE) are continuous variables. Here is a brief summary of the data set:

SITE : 1 2 3 4 5 6 7 8 9 10 … SALAMAN : 13 11 11 9 8 7 6 6 5 5 … PCTCOVER : 85 86 90 88 89 83 83 91 88 90 … FORESTAGE: 316 88 548 64 43 368 200 71 42 551 …

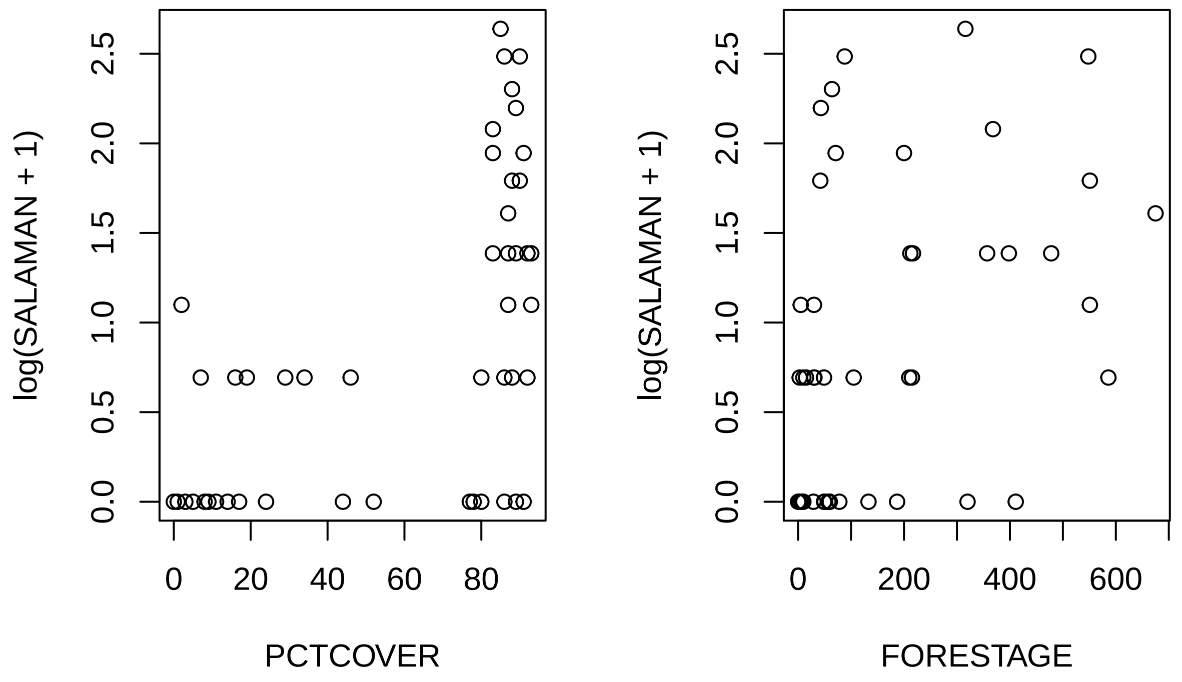
The salamander data is in file *salamander.csv*. Import the data into R and attach the resulting data frame. We will begin by looking at several descriptive plots of the data. First, look at scatterplots of salamander count versus 1) percent canopy cover and 2) forest age.

```r
salamander<-read.csv("salamander.csv")
attach(salamander)
par(mfrow = c(1, 2))
plot(PCTCOVER, SALAMAN)
plot(FORESTAGE, SALAMAN)
```



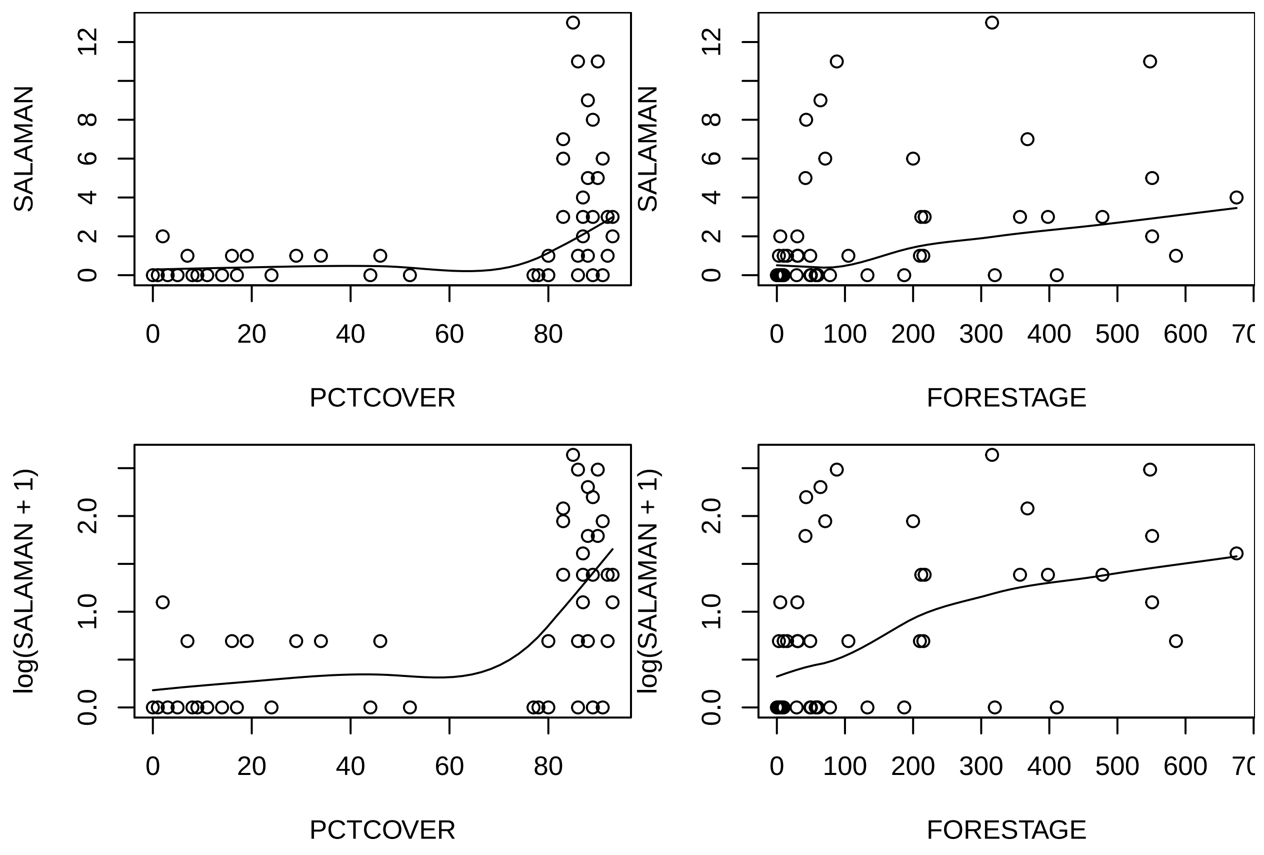
The `log` function is the conventional link function for Poisson regression. Hence it would be appropriate to plot the log of the response variables versus the explanatory variables to examine the relationship between the response and explanatory variables on the log-scale. However, some actual counts are 0. Therefore, we plot the log of one plus the salamander counts instead (The presence of zeros is not a problem when fitting the model using `glm` because predicted values cannot be zero, only very close to zero):

```
par(mfrow = c(1, 2))
plot(PCTCOVER, log(SALAMAN + 1))
plot(FORESTAGE, log(SALAMAN + 1))
```



Fit a LOWESS smooth function on the un-tansformed scale and on log-transformed scale:

```r
par(mfrow=c(2,2), mar=c(5,4,0,0))
scatter.smooth(PCTCOVER, SALAMAN)
scatter.smooth(FORESTAGE, SALAMAN)
scatter.smooth(PCTCOVER, log(SALAMAN+1))
scatter.smooth(FORESTAGE, log(SALAMAN+1))
```



```r
par(mfrow=c(1,1))
detach(salamander)
```

These plots show an unusual relationship between percent canopy cover and salamander count. When the canopy coverage is less than 60 percent, the counts are always small, ranging from zero to two. When the canopy coverage is greater than 70 percent, the counts are much larger, ranging from zero to twelve. There are no sites with intermediate canopy coverage. There also appears to be a large shift in variability of the response when canopy coverage jumps from below 60 percent to above 70 percent. There appears to be a weak relationship with forest age at either scale.

We will start our analysis by fitting a model with many parameters, including a dummy variable for whether or not the canopy cover is greater than 70 percent, quadratic effects and interactions. The following model includes these terms:

- a linear relationship between salamander counts and forest age
- a quadratic relationship with percent canopy cover
- an interaction between percent cover and forest age

These same relationships are fit separately for those forest plots with low canopy cover ($< 70\%$) and for forest plots with high canopy cover ($> 70\%$). The separate fits are coded in the model as an interaction between the above model terms and the categorical 'cover' variable as defined below.

To determine if the error structure (i.e. the Poisson assumption) is appropriate, we fit the full model and examine the residuals. If the parameter-rich model fits well, even if it has unnecessary variables, then the residuals should indicate a good fit! First, create a new dummy variable of low ($< 70\%$) vs. high ($>70\%$)

canopy cover and add it to the 'salamander' data frame. The new variable will be a logical vector, which serves as a dummy factor to contrast low and high cover:

```r
salamander$cover <- salamander$PCTCOVER > 70
salamander$cover
```

```
##  [1]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [13]  TRUE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE
## [25] FALSE  TRUE  TRUE  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [37] FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
```

```r
# Fit model
fit <- glm(SALAMAN ~ (PCTCOVER * FORESTAGE + I(PCTCOVER^2) +
             I(FORESTAGE^2)) * cover, data = salamander, family = poisson)
summary(fit)
```

```
##
## Call:
## glm(formula = SALAMAN ~ (PCTCOVER * FORESTAGE + I(PCTCOVER^2) +
##     I(FORESTAGE^2)) * cover, family = poisson, data = salamander)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.5499  -1.0005  -0.3325   0.3949   2.4898
##
## Coefficients:
##                               Estimate Std. Error z value Pr(>|z|)
## (Intercept)                 -1.452e+00  8.945e-01  -1.623  0.10464
## PCTCOVER                     5.540e-02  1.341e-01   0.413  0.67960
## FORESTAGE                    4.677e-02  1.001e-01   0.467  0.64028
## I(PCTCOVER^2)               -1.845e-03  4.112e-03  -0.449  0.65363
## I(FORESTAGE^2)              -2.579e-03  2.905e-03  -0.888  0.37470
## coverTRUE                   -2.743e+02  6.454e+01  -4.250 2.14e-05 ***
## PCTCOVER:FORESTAGE           2.810e-03  5.746e-03   0.489  0.62485
## PCTCOVER:coverTRUE           6.447e+00  1.511e+00   4.265 2.00e-05 ***
## FORESTAGE:coverTRUE         -7.933e-02  1.024e-01  -0.775  0.43852
## I(PCTCOVER^2):coverTRUE     -3.626e-02  9.716e-03  -3.731  0.00019 ***
## I(FORESTAGE^2):coverTRUE     2.576e-03  2.905e-03   0.887  0.37534
## PCTCOVER:FORESTAGE:coverTRUE -2.410e-03  5.751e-03  -0.419  0.67523
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 190.219  on 46  degrees of freedom
## Residual deviance:  89.178  on 35  degrees of freedom
## AIC: 198.24
##
## Number of Fisher Scoring iterations: 6
```

We can compute a goodness-of-fit test based on the Pearson residuals, the sum of which is identical to Pearson's goodness of fit statistic and provides a measure of how well the expected counts agree with the observed counts (Pearson's $\chi^2$ test).

We can compute Pearson's $\chi^2$ by first extracting the Pearson residuals from the object 'fit' (use function `resid()` with the `type="pearson"` argument), squaring the residuals, and then taking the sum.

This statistic should follow a $\chi^2$ distribution if the null hypothesis is true (i.e. if the data follow a Poisson distribution). Thus, under the null hypothesis, we can compute the probability of obtaining a $\chi^2$ value as large as or larger than the observed one using the fact that the $\chi^2$-statistic has a $\chi^2$ distribution with $n - p$ degrees of freedom. You can compute the desired probability by using the function `pchisq` In addition to the value of the $\chi^2$ statistic, you need to provide the residual degrees of freedom $(n - p)$ to this function (see `?pchisq`), which can be extracted from the fitted model object using: `df.residual(fit)` OR: `fit$df.resid`

```
chisq <- sum(residuals(fit, type="pearson")^2)
pchisq(chisq, df.residual(fit), lower.tail = FALSE)
```

```
## [1] 1.209879e-05
```

The $\chi^2$ value is rather large and the resulting p-value is very small, hence we can reject the null hypothesis that the data follow a Poisson distribution. Therefore our model, even with all of its parameters, does not fit very well and is not suitable for drawing inferences because the Poisson distribution is not appropriate! (A different and equally valid test is directly based on the deviance, which has the same $\chi^2$ distribution with $n - p$ degrees of freedom for large $n$).

For small sample sizes, the distribution of either of these statistics may differ from a $\chi^2$. Typically, both give very similar results. For the deviance test you can simply compute `deviance(fit)` and test whether its value is unusually large for a $\chi^2$ distribution with $n - p$ d.f. Note that the $\chi^2$ statistic (which we saved as `chisq`) and the deviance are very similar:

```
chisq
```

```
## [1] 82.01949
```

```
deviance(fit)
```

```
## [1] 89.17784
```

A possible explanation for the poor fit is that there is "extra" Poisson variation (= over-dispersion). As we saw before, the Poisson distribution assumes that the mean and variance are identical, a rather restrictive assumption). If the actual distribution of counts had a variance that is larger than the mean (which could, for example, result from unmeasured variables or a clustered distribution of salamanders in the forest), the model would not fit well.

## Overdispersion

There are several ways to check for over-dispersion. Graphically, over-dispersion should be apparent in the distribution of residuals. The standardized deviance residuals should have a variance of approximately 1, thus absolute residuals larger than 2 should be rare.

We can check for over-dispersion by extracting and plotting the residuals against the fitted values and add horizontal lines at 0 and at -2 and 2 (using `abline()` - see script).

The residual plot has a number of residuals larger than 2 in absolute value, but no extreme outliers.

This is strong evidence that the data may be over-dispersed (If the deviance were large due to a small number of extreme outliers, extra-Poisson variation may not have been the best explanation). Over-dispersion does not affect the parameter estimates themselves, but it does affects standard errors and p-values (both of which will be smaller than they should be).

Another, slightly more formal way to assess over-dispersion is by comparing the residual deviance with the residual degrees of freedom. If the former is much larger than the latter, this is indication of over-dispersion. In fact, an estimate of the over-dispersion parameter can be obtained by dividing the deviance by the residual degrees of freedom.

## Exercise 1: Estimate overdispersion

For this exercise, estimate the over-dispersion parameter for the salamander data (based on the full model that we fit above) using the ratio of the deviance to the residual degrees of freedom.

```
# Insert your code here
```

Is the estimate of overdispersion much larger than 1? What can you conclude?

INSERT YOUR ANSWER BETWEEN CURLY BRACKETS

### Eliminating Forest Age

In the previous fit, every term with forest age was insignificant. We can check to see if a model that drops all terms related to forest age provides a reasonable fit using a deviance test (likelihood ratio test).

## Exercise 2: Re-fit model without forest age

Fit the same model as above (`fit`) without any of the FORESTAGE terms and save output as `fit2`. Compare the two models using `anova()`. The F-test is not appropriate here (because of the poisson distribution) and you should use the Chi-square test by specifying the test = "Chisq" argument to `anova` (See `?anova.glm` for details).

```
# Eliminating Forest Age from analysis
# Insert your code here
```

Based on the results, does it seem justified to drop forest age from the model (or conversely, is the reduction in deviance enough to justify the extra parameters of the more complex model including forest age)? Note that over-dispersion in this case implies that standard errors and p-values are smaller than they should be, hence we are MORE likely to reject the null hypothesis and accept the more complex model than we would be without over-dispersion. Does that give you more or less confidence in your conclusion?

INSERT YOUR ANSWER BETWEEN CURLY BRACKETS

### Quasi-likelihood approach

A quasi-likelihood model permits extra variation by multiplying the variance term by an appropriate "adjustment factor". The fitted values will be identical to those from the Poisson log-linear regression, but the standard errors will be larger, and hence the p-values will be larger as well. As we saw earlier, a quasi-likelihood model for the Poisson can be fit by using family=quasipoisson instead of family=poisson in the call to `glm()`. (Similarly, an over-dispersed binomial model could be fit using `family = quasibinomial`).

Refit model `fit2` using family = quasipoisson and save as `fit3`. You can simply use the update function:

```r
# Un-comment (remove '#') the following line to run code:
#fit3 <- update(fit2, family = quasipoisson)
```

## Exercise 3: Over-dispersion parameter

Look at the model results produced by `summary`. Look for the estimate of the over-dispersion parameter and compare to the estimate obtained above! Compare the estimated coefficients and standard errors produced by `fit2` and `fit3`!

How and why do they differ?

INSERT YOUR ANSWER BETWEEN CURLY BRACKETS

## Exercise 4: Diagnostics

Generate some plots for residual diagnostics for our current "best" model:

```r
# Insert your code here
```

Are there any unusual patterns? Does the plot suggest any problems?

INSERT YOUR ANSWER BETWEEN CURLY BRACKETS

### Visualize fitted model with confidence bands

Compute predicted values for a range of PCTCOVER and plot both observed and predicted numbers of salamanders against PCTCOVER. Add 95% confidence bands!

I outline the approach here to help you understand the code or if you want to figure it out yourself - the actual code is included in the script file!

1. Create a data frame with two variables `PCTCOVER` and `cover` (for the predict function these need to have the same names as the explanatory variables in the fitted object - `fit3` in this case).

```r
new <- data.frame(PCTCOVER = 0:100, cover = c(0:100) > 70)
```

2. Use `predict()` with argument se=T and type = "response", which gives predictions on the back-transformed scale of the response variable, i.e. predicted salamander counts for the values of the explanatory variable in the data frame "new". Save the output, which is a list with the fitted values (component 'fit') and the standard error for each predicted value ('se.fit'). Note that these are standard errors of the **mean response** (for confidence intervals) at each value of PCTCOVER and **not** prediction intervals for a new observation! (which would need to take into account the residual variance in addition to the variance of the parameter estimates and hence would be much wider).
3. Plot salamander counts against `PCTCOVER`
4. Add a line to the plot for the predicted values (use lines with 0:100 as x-values and the predicted values as y-values - these are given in component `fit` of the output in 2.

5. Add lines for lower and upper confidence bands. Use predicted values +/- 2 times the standard error, which you can compute from the output produced in 2. (Components `fit` and `se.fit`). Use `lines()` with argument `lty=2` and / or a different color, e.g. `col = 2`.

Note that (on the back-transformed scale) the confidence band for `PCTCOVER` over ~50% goes through the roof because there are no observations. Because of the two piece fit, the fitted values and the confidence intervals are also discontinuous!

### An alternative approach to overdispersion

As we saw above, there is strong evidence of over-dispersion, as well as some evidence of heteroscedasticity. We used the quasi-likelihood approach (not a true likelihood approach, hence we can't compute likelihood-based statistics), to deal with over-dispersion. A different and more flexible approach to dealing with over-dispersion is the use of the negative binomial distribution instead of an over-dispersed binomial distribution. This does not, however, deal with heteroscedasticity. For simplicity, and because the variance is much larger at canopy cover values of over 70%, we will fit a model to the plots with over 70% canopy cover only to alleviate concerns about heteroscedasticity and to illustrate the negative binomial model.

The negative binomial model has an extra parameter that makes it more flexible and allows for a larger variance in the observed counts than the binomial distribution. The `glm` function currently does not implement maximum likelihood estimation using the negative binomial distribution, but the MASS library of Venables & Ripley includes a function to fit a negative binomial model (`glm.nb()`):

```r
library(MASS, quietly=T)
fit4 <- glm.nb(SALAMAN ~ PCTCOVER + I(PCTCOVER^2), data = salamander,
               link="log", subset = cover)

# Here, ‘cover‘ (which is a logical variable that is TRUE where ‘PCTCOVER‘ is > 70)
# is used to extract the desired subset of observations with PCTCOVER > 70.

summary(fit4)
```

```
##
## Call:
## glm.nb(formula = SALAMAN ~ PCTCOVER + I(PCTCOVER^2), data = salamander,
##     subset = cover, link = "log", init.theta = 1.770900007)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.2638  -0.8400  -0.1340   0.5534   1.3056
##
## Coefficients:
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -271.29624   88.63812  -3.061  0.00221 **
## PCTCOVER         6.26824    2.04847   3.060  0.00221 **
## I(PCTCOVER^2)   -0.03597    0.01182  -3.042  0.00235 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Negative Binomial(1.7709) family taken to be 1)
##
##     Null deviance: 41.430  on 27  degrees of freedom
## Residual deviance: 30.015  on 25  degrees of freedom
## AIC: 136.06
##
## Number of Fisher Scoring iterations: 1
##
##
##               Theta:  1.771
##           Std. Err.:  0.766
##
##  2 x log-likelihood:  -128.063
```

The output is similar to the Poisson output, but we also get an estimate for a parameter called 'theta', which is a parameter of the negative binomial distribution (a measure of "dispersion"").

## Exercise 5: Conclusions

Examine the z statistics from the `summary()` output above to test whether coefficients are different from zero, and examine the analysis of variance table that performs a sequential test of each term:

```r
anova(fit4, test = "Chisq")
```

```
## Warning in anova.negbin(fit4, test = "Chisq"): tests made without re-estimating
## 'theta'
```

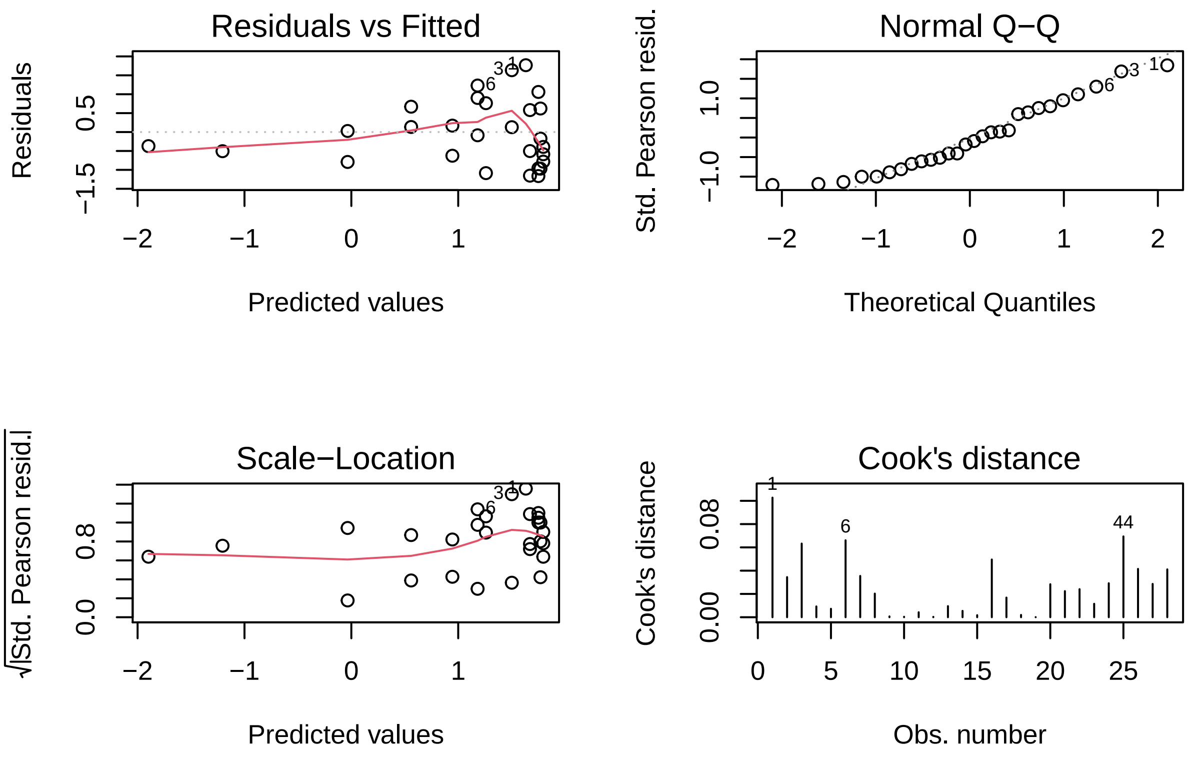
```
## Analysis of Deviance Table
##
## Model: Negative Binomial(1.7709), link: log
##
## Response: SALAMAN
##
## Terms added sequentially (first to last)
##
##
##               Df Deviance Resid. Df Resid. Dev Pr(>Chi)
## NULL                             27     41.430
## PCTCOVER       1    1.175        26     40.255 0.278374
## I(PCTCOVER^2)  1   10.240        25     30.015 0.001374 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

What do the z-tests and likelihood ratio tests returned by `anova` suggest? Should we choose a simpler model based on this test?

INSERT YOUR ANSWER BETWEEN CURLY BRACKETS

Finally, plot residual diagnostics:

```r
par(mfrow=c(2,2))
plot(fit4, which=1:4)
```



Do the plots suggest any problems?

INSERT YOUR ANSWER BETWEEN CURLY BRACKETS

## Poisson as special case of the Negative Binomial

Because the Poisson is a special case of the negative binomial model, we can directly test if the negative binomial model fits better using a likelihood ratio test (compute "by hand" or you can use the 'lrtest' function in the package 'lmtest')

First, we fit the same model assuming a Poisson distribution, then we extract the negative log-likelihood values from both models and construct the LRT statistic:

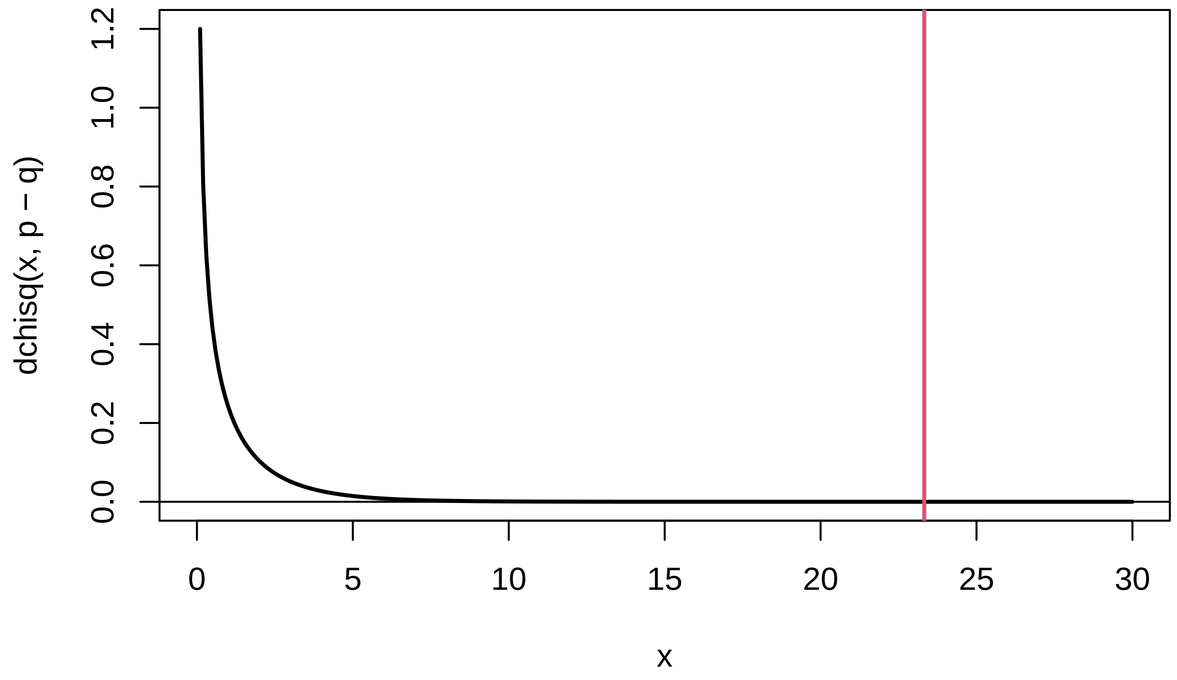
```
fit4b <- glm(SALAMAN ~ PCTCOVER + I(PCTCOVER^2), data = salamander,
   family = poisson, subset = cover)
# Compute likelihood ratio test statistic:
L1 <- -logLik(fit4b)    # minus negative log-Likelihood, Model 1
L2 <- -logLik(fit4)     # minus negative log-Likelihood, Model 2
q <- attr(L1, "df")   # model degrees of freedom are stored as
                      # attribute of the log-Likelihood
p <- attr(L2, "df")
LRT <- 2*L1 - 2*L2   # Likelihood-ratio test statistic!
```

Under the null hypothesis (no difference between models), the distribution of LRT follows a Chi-square distribution with $p - q$ degrees of freedom. The probability that a $\chi^2$ with $p - q$ d.f. is larger than LRT is

given by:
`1-pchisq(LRT, p-q)`
where `pchisq()` computes the cumulative distribution function.

Graphical illustration:

```
x <- seq(0, 30,by=0.1)
plot(x, dchisq(x, p-q), type="l", lwd=2); abline(h=0)
abline(v=LRT, col=2, lwd=2)
```



The probability that a Chi-square with 1 d.f. is as large as LRT = {23.32} corresponds to the area under the curve to the right of the red line, which is vanishingly small. Hence we can confidently reject the null hypotheses and conclude that the negative binomial model is clearly the better model!

As a simple alternative we can use 'lrtest' to get the same test:

```
library(lmtest, quietly=T, warn.conflicts=F)
```

```
##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```r
lrtest(fit4, fit4b)
```

```
## Warning in modelUpdate(objects[[i - 1]], objects[[i]]): original model was of
## class "negbin", updated model is of class "glm"
```

```
## Likelihood ratio test
##
## Model 1: SALAMAN ~ PCTCOVER + I(PCTCOVER^2)
## Model 2: SALAMAN ~ PCTCOVER + I(PCTCOVER^2)
##   #Df  LogLik Df  Chisq Pr(>Chisq)
## 1   4 -64.031
## 2   3 -75.693 -1 23.323  1.369e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

## Exercise 6: Confidence bands

Following the example above, plot the data with the fitted model and confidence bands for the negative binomial model that only considers cover > 70% ('fit4'). (As an alternatively, use ggplot if you wish) For comparison, superimpose the fitted line and confidence bands resulting from the Poisson model ('fit4b'), as well as confidence bands resulting from a quasipoisson model and discuss the differences with your group. (Use `update()` to re-fit model 'fit4b')

**Solution:**

```r
# Insert your code here
```

INSERT YOUR ANSWER BETWEEN CURLY BRACKETS

### `gamlss()` implementation

The same models can be fit using function 'gamlss' in the package of the same name. This function can also fit generalized linear models (as well as generalized additive models) and you can choose from a very large number of distributions (see help for 'gamlss.family')

```r
library(gamlss, quietly=T, warn.conflicts=F)
```

```
## Warning: package 'gamlss' was built under R version 4.0.5
```

```
## Warning: package 'gamlss.data' was built under R version 4.0.4
```

```
##
## Attaching package: 'gamlss.data'
```

```
## The following object is masked from 'package:datasets':
##
##     sleep
## Warning: package 'gamlss.dist' was built under R version 4.0.5

##  **********   GAMLSS Version 5.3-4  **********

## For more on GAMLSS look at https://www.gamlss.com/

## Type gamlssNews() to see new features/changes/bug fixes.
```

```
fit1 <- gamlss(SALAMAN ~ PCTCOVER + I(PCTCOVER^2),
    data = salamander[salamander$cover,], family = NBI)
```

```
## GAMLSS-RS iteration 1: Global Deviance = 128.0678
## GAMLSS-RS iteration 2: Global Deviance = 128.0629
## GAMLSS-RS iteration 3: Global Deviance = 128.0629
```

```
summary(fit1)
```

```
## Warning in summary.gamlss(fit1): summary: vcov has failed, option qr is used instead

## ******************************************************************
## Family:  c("NBI", "Negative Binomial type I")
##
## Call:  gamlss(formula = SALAMAN ~ PCTCOVER + I(PCTCOVER^2), family = NBI,
##     data = salamander[salamander$cover, ])
##
## Fitting method: RS()
##
## ------------------------------------------------------------------
## Mu link function:  log
## Mu Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -271.32276   88.67392  -3.060  0.00523 **
## PCTCOVER         6.26885    2.04927   3.059  0.00524 **
## I(PCTCOVER^2)   -0.03597    0.01183  -3.041  0.00547 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## ------------------------------------------------------------------
## Sigma link function:  log
## Sigma Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -0.5715     0.4399  -1.299    0.205
##
## ------------------------------------------------------------------
## No. of observations in the fit:  28
## Degrees of Freedom for the fit:  4
##       Residual Deg. of Freedom:  24
##                       at cycle:  3
##
## Global Deviance:     128.0629
##             AIC:     136.0629
##             SBC:     141.3917
## ******************************************************************
```

```r
fit2 <- gamlss(SALAMAN ~ PCTCOVER + I(PCTCOVER^2),
    data = salamander[salamander$cover,], family = PO)
```

```
## GAMLSS-RS iteration 1: Global Deviance = 151.3864
## GAMLSS-RS iteration 2: Global Deviance = 151.3864
```

```r
summary(fit2)
```

```
## Warning in summary.gamlss(fit2): summary: vcov has failed, option qr is used instead

## ******************************************************************
## Family:  c("PO", "Poisson")
##
## Call:  gamlss(formula = SALAMAN ~ PCTCOVER + I(PCTCOVER^2), family = PO,
##     data = salamander[salamander$cover, ])
##
## Fitting method: RS()
##
## ------------------------------------------------------------------
## Mu link function:  log
## Mu Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -2.199e+02  5.537e+01  -3.972 7.14e-05 ***
## PCTCOVER       5.101e+00  1.278e+00   3.992 6.55e-05 ***
## I(PCTCOVER^2) -2.935e-02  7.368e-03  -3.983 6.79e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## ------------------------------------------------------------------
## No. of observations in the fit:  28
## Degrees of Freedom for the fit:  3
##       Residual Deg. of Freedom:  25
##                       at cycle:  2
##
## Global Deviance:     151.3864
##             AIC:     157.3864
##             SBC:     161.383
## ******************************************************************
```

```r
lrtest(fit1, fit2)
```

```
## Likelihood ratio test
##
## Model 1: SALAMAN ~ PCTCOVER + I(PCTCOVER^2)
## Model 2: SALAMAN ~ PCTCOVER + I(PCTCOVER^2)
##   #Df  LogLik Df  Chisq Pr(>Chisq)
## 1   4 -64.031
## 2   3 -75.693 -1 23.323  1.369e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

## The End