Testing for relationships Part 2

Aims of this practical

1. Write clear hypotheses

2. Explore descriptive plots /create appropriate plots to visualise hypotheses

3. Carry out regression models

    - Linear model
    - Multiple linear model
    - Generalised linear model

4. Model validation

5. Report your results

Coding you will learn:

Tests conducted

Name	Data type
Linear regression	Continuous data
Multiple linear regression	Continuous data could also include categorical predictors
GLM	Count and binary data

Other useful code for…
More plotting: pairs, corrplot, scatterplotMatrix, model validation plots
predict function

Linear model

To begin with lets recap the linear model from last week. Remember to set up your script like previous weeks with the code you can copy from weeks 1-4 (see code below). As a refresher, below is the equation for the linear regression.

Y=a+bX **OR** Y=mx+c **either way** Y=intercept+(slope*X)

Confusingly, textbooks will call the X and Y components of the model different things so here are the most commonly used names. The ones in bold are the ones we will use for this course.

Y = **response** variable = dependent variable
X = **explanatory** variable = predictor = independent variable

Remember, when you plot your model variables you need to know which ones go on which axis. The response should be on the Y-axis and the explanatory should be on the X-axis.

```
# DD/MM/YY Your Name
# Data: various
# Analysis: Tests for relationships

# clears r console
rm(list=ls())
# set working directory
setwd("/path/to/my/data/My_R_Space")
# check working directory
getwd()
# list of files in working directory
list.files()
```

We are going to start with a nice simple example using data you have seen in the previous weeks. Load the built in iris data set. The first thing we will do is reduce the data set using the code below to create an object containing only the first 50 rows of the iris dataset.

```
# Test: Linear regression
# Data: iris dataset
data(iris)
myiris <- iris[1:50, ] # use first 50 rows
```

Then go ahead and check the data using *names()* and *str()*. For this example we will use Sepal.Length to predict Sepal.Width – sepal width is the response (Y) variable and sepal length is the explanatory (X) variable. We want to first know if there is a significant

relationship and, if so, then we want to be able to use the explanatory variable to predict values for sepal width using the linear regression equation.

Ok, state your null hypothesis for this data and plot the data using a simple scatter plot. Now the fun part, lets model the data using the *lm* function.

```
# 1. Null Hypothesis - there is no relationship between SW and SL

# 2. Plot the data
plot(Sepal.Width~Sepal.Length, data=myiris) #  do you think sepal length is a strong explanatory variable of sepal width?

# 3. model the data
iris.mod <- lm(Sepal.Width~Sepal.Length, data=myiris)
```

Before we look at the results of the model we need to first validate the fit of the model. To do this we first conduct a histogram of the residuals and then use the plot function on the model object.

```
#4. model validation
hist( resid(iris.mod) )

class(iris.mod) # tells you that the iris.mod object is of the class 'lm' (linear model)
par(mfrow=c(2,2))
plot(iris.mod)
```

Do the validation plots look ok? Do the residuals look normally distributed? Are there any strong patterns in the residuals? Are any outliers having a strong effect in the model?

ONLY AFTER CHECKING THE VALIDATION PLOTS DO WE THEN CHECK THE MODEL RESULTS

Using the **summary()** function on your model object will give you the results below

```
Call:
lm(formula = Sepal.Width ~ Sepal.Length, data = myiris)

Residuals:
     Min       1Q   Median       3Q      Max
-0.72394 -0.18273 -0.00306  0.15738  0.51709

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)   -0.5694     0.5217  -1.091    0.281
Sepal.Length   0.7985     0.1040   7.681 6.71e-10 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.2565 on 48 degrees of freedom
Multiple R-squared:  0.5514,    Adjusted R-squared:  0.542
F-statistic: 58.99 on 1 and 48 DF,  p-value: 6.71e-10
```

The test output gives you the value of the F-statistic, the degrees of freedom and finally the significance value (p-value) for the model. The output also gives you the model **intercept** and **slope**, which you can use with the linear regression equation to predict values of Y (sepal width). The model output also tells you which of these values is significantly different from zero. Generally, we are not interested in whether the intercept is different from zero but we are more concerned with knowing if the slope is different from zero, which in this case the slope of 0.7985 is significantly different from a slope of zero ($P<0.001$). This tells us there is a positive relationship between our explanatory variable (sepal length) and our response variable (sepal width).

We can also check the confidence interval around the slope and intercept using the following code **confint(iris.mod, level = 0.95)**

This tells us that we can be 95% certain the true intercept falls between -1.62 and 0.48 (and as this range contains zero the above test can not report a significant difference from zero). Also, the true slope falls within the range 0.59-1.01 which doesn't contain zero (so must be significantly different from zero). Isn't science fun!

Reporting the results

Now report the results in a nice concise sentence, e.g. we found sepal length is a significant linear predictor of sepal width (linear regression: $R^2 = 0.55$, df = 48, $P < 0.001$).

We also need to plot the data. To do this we could simply use the **plot()** function followed by the **abline()** function as we have done in previous practicals. But as we will use the **predict()** function later in this practical lets use it now on this familiar dataset.

```
# 5. Report results
# Make new X data, which we will use to make predictions
newX<-data.frame(Sepal.Length=seq(from=4.3, to=5.8, length.out=50)) # 50 new X data values
newX

# Use predict to make new Y's (our predictions) with confidence intervals
newY<-predict(iris.mod, newdata=newX, interval="confidence")

# Housekeeping - bring newX and newY together
add.these2plot<-cbind(newX, newY)

#take a look at add.these2plot object. Now lets make the plot
plot(Sepal.Width~Sepal.Length, data=myiris, xlab="Sepal length (cm)", ylab="Sepal width (cm)")
lines(fit ~ Sepal.Length, data=add.these2plot, col="blue", lwd=3, lty=3)
# Using the add.these2plot object, see if you can add the upper and lower confidence intervals
```

Using the model to predict values

Finally, we said our goal was to not only test the relationship but also to use the linear regression equation to predict values of sepal width.

Lets look the equation again: Y= ab+c **OR** Y=intercept+(slope*X)

Calculate Y values (sepal width values) for the X values (sepal length values) of **2** and **5** using the linear regression equation (type the equation into R-Studio).

Extend you plot axes and see if the Y values you calculated look correct.

```
plot(Sepal.Width~Sepal.Length, data=myiris, ylim=c(-1,6), xlim=c(-1,6))
abline(iris.mod)
```

Multiple linear regression 1

What if you have more than one explanatory variable! Don't panic, its time to use multiple linear regression! The next data set to download from Moodle is the '*sleep.csv*' file. This data set consists of sleep times of various mammal species. We want to know if sleep time (hrs/day) is related to body size and any other covariates. Variables in the dataset are;

BodyWt = body weight (kg)
BrainWt = brain weight (g)
TotalSleep = sleep time per day (hrs)
LifeSpan = species mean life span (yrs)
Gestation = gestation period (days)
Danger = overall danger index (1-5) 1 = least danger (from other animals); 5 = most danger (from other animals)

First, what is your response and what are your explanatory variables?

1. Clearly state your null hypothesis

2. Plot the data. Remember, scatters for continuous data and boxplots for factors.

Do the variables look sensible, are there any outliers (hint: look at body weight and brain weight). We should also look to see if any of the explanatory variables are correlated. Strong correlation indicates collinearity in explanatory variables and we will need to deal with this before modelling. The code below will help assess collinearity and also check for normality in our covariates.

```
sleep<-na.omit(sleep)  # Remove all NA's from dataset

# Check for normality of variables (boxplots centred) and linearity of relationships
library(car)
scatterplotMatrix(~BodyWt+BrainWt+LifeSpan+Gestation, data=sleep,
diagonal=list(method="boxplot"))

# Ok, so the covariates are not normal, we could see outliers from the scatter plots above. One
solution is to transform (log10) the non-normal variables and try again.
scatterplotMatrix(~log10(BodyWt)+log10(BrainWt)+LifeSpan+Gestation,
data=sleep,diagonal=list(method="boxplot"))

# This looks much better!

# Lets look at collinearity now
pairs(~log10(BodyWt)+log10(BrainWt)+LifeSpan+Gestation, data=sleep, upper.panel = NULL)
# We have some strong correlations, lets see how strong using corrplot
library(corrplot)
M<-cor(sleep[2:6], method="pearson") # create matrix of pearson correlations
corrplot(M, method="number", type="lower") # plot matrix showing correlation scores
```

So we have some strong correlations in our covariates and hence collinearity in our data, we will need to deal with this before we can build our model. BrainWt is strongly correlated with our main explanatory variable BodyWt and also Gestation, so we should not include BrainWt in the model.

Let's build the full model and test the significance of each covariate.

```
# 3. Fit the model
Mult.lm1 <- lm(TotalSleep~log10(BodyWt)+LifeSpan+Gestation+Danger, data=sleep)
summary(Mult.lm1)
# if all the terms (covariates) are not significant try removing the least significant and rerun
# the model. Repeat this until all terms are significant – but don't forget to keep in body weight
# as this is the thing we are interested in.
```

What is your final model?

4. Now lets check the model validation plots as above in the linear model example. Do these you sensible? Do your residuals look normally distributed?

5. If you are happy with the model validation you can then extract the results using **summary()** and report these in a sentence.

To plot the data we need to first create some predicted data using the **predict()** function**.**

```
 # Make new X data, first finding out the min and max of the
min(sleep$BodyWt)
max(sleep$BodyWt)
newX<-expand.grid(BodyWt = seq(from=0.005, to=6654, length=50),
            Danger =seq(from=1, to=5, by=1))
newX

# Predictions of TotalSleep using model. Note: Mult.lm3 should be the name of your final model
newY<-predict(Mult.lm3, newdata=newX, interval="confidence")

# Housekeeping
sleep.plot.metrics<-cbind(newX, newY)
sleep.plot.metrics
head(sleep.plot.metrics)
```

We have modeled data for each of the separate danger index values, and have subset only the middle danger index of 3 to use in our plot (you can do all of them if you have time).

```
danger3<-subset(sleep.plot.metrics, sleep.plot.metrics$Danger == 3)

plot(TotalSleep~log10(BodyWt), data=sleep)  # plot data
lines(fit ~ log10(BodyWt), data=danger3) # add modelled line
lines(upr ~ log10(BodyWt), data=danger3, lty=2) # add upper CI
lines(lwr ~ log10(BodyWt), data=danger3, lty=2) # add lower CI
```

Generalised linear model 1: count data

A GLM allows you to fit a model when the error structure (residuals) is not normally distributed. It is a flexible tool that allows you to change the model error structure to accommodate tricky real-life data.

In ecology we often count things, abundance of animals, species richness of plants, this count data is often suited to a poisson error distribution. A good rule of thumb is if you have count data for your response variable you will probably need to use a poisson GLM when looking at relationships.

The data set we use now contains the number of herbivores (count data), the annual net primary productivity (ANPP) of grassland and the winter rainfall. Specifically, we want to know if there is a relationship between the number of herbivores and productivity. Go ahead and load the '*herbivore.csv*' file from moodle. Have a look at the variable names and data types.

Which is the response variable?
Which are the explanatory variables?

1. Write a null hypothesis to test.

2. Now plot your data using either base R (or ggplot2).
As we have two explanatory variables we also need to explore these and check for collinearity (see code below).

```
# Test: GLM
# Data: herbivore abundance
# Null hypothesis:

library(ggplot2)
ggplot(herb, aes(x=ANPP, y=herb.abund)) +
  geom_point()

# Check for collinearity
pairs(~herb.abund+ANPP+winter.precip, data=herb, upper.panel = NULL)
# No correlation between ANPP and winter.precip

# Or using corrplot
library(corrplot)
M<-cor(herb[1:3], method="pearson")
corrplot(M, method="number", type="lower")
# Again no correlation between explanatory variables
```

Ok great, no correlation between explanatory variables so we can use both in our model. If winter precipitation was strongly correlated with ANPP we would have to leave this variable out of our model.

From our scatter plot of the response and explanatory variable do you think we will accept or reject our null hypothesis?

Ok, now lets set up the model. Its very similar to the linear models we have used previously but this time we use the **glm()** function and specify the error structure by setting family equal to poisson.

```
#3. Preform GLM test
herb.mod<-glm(herb.abund ~ ANPP + winter.precip, family = poisson, data=herb)
summary(herb.mod)
```

You can see from the summary output that not all of our explanatory variables are significant. Winter precipitation is not significant so we can remove this term from the model and try again.

```
#3. Preform GLM test
herb.mod<-glm(herb.abund ~ ANPP, family = poisson, data=herb) # overwrite the first model
summary(herb.mod)
```

Now that all our explanatory variables are significant we need to check the model validation plots.

```
#4. Model validation
par(mfrow=c(2,2))
plot(herb.mod)
```

Are there any strong patterns in the residuals? What about Cooks distance, have we got any outliers strongly influencing the model?

5. If you are happy with the validation plots now use the **summary()** function to read the model output so we can start to report the results.

```
Call:
glm(formula = herb.abund ~ ANPP, family = poisson, data = herb)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-2.65553  -0.72859  -0.05732   0.57165   2.43592

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -0.2660242  0.0890232  -2.988  0.00281 **
ANPP         0.0064326  0.0002484  25.898  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 752.28  on 197  degrees of freedom
Residual deviance: 192.55  on 196  degrees of freedom
AIC: 806.81
```

The p-value is much smaller than 0.05 so you should reject your null hypothesis, i.e. there is a significant relationship between the two variables. To write this up in a clear statement we first need to get a test statistic, for GLM's we can use a pseudo-R^2. To calculate this we need to use the following equation;

Pseudo-R^2 = (Null deviance-Residual deviance)/Null deviance

This value will vary between 0-1 with higher values meaning your model has greater explanatory power, just like R^2 in a linear model. Write a clear results statement.

Now lets use the **predict()** function again to predict some values from our model.

```
# Make some new X data
newX<-data.frame(ANPP = seq(from=30, to=644, length=50))
newX
# Predictions using model
newY<-predict(herb.mod, newdata=newX, type="response", se=T)

# Confidence intervals is   X +/- (2xSE)
pred.plus<-newY$fit+2*newY$se
pred.minus<-newY$fit-2*newY$se

# Housekeeping
herb.plot.metrics<-cbind(newX, newY, pred.plus, pred.minus)
herb.plot.metrics
head(herb.plot.metrics) # view the first few lines of your new object
```

Now make a plot including the modeled line and confidence intervals using either base R **line()** or ggplot2 **geom_line()**. [hint: we did this for sleep and iris data, but in this GLM our confidence intervals are called pred.plus and pred.minus rather than lwr and upr.

Generalised linear model 2: count data

For the final model in this practical go ahead and import the data set called "*birds.csv*" from moodle. The data represent a nationwide survey of the number of small brown passerines nesting in 127 representative districts in England, Scotland and Wales.

Question: Is there any relationship between the number of birds and the area of the district and does the relationship differ among countries? What is your response variable? What are your explanatory variables?

1. Write out your null hypothesis

2. Inspect the data (summary, str, names.) and inspect the data graphically (scatter & boxplot)

3. Model the data, remember this is counts of bird abundance so which type of model should you us?

Check to make sure all the model terms (explanatory variables) are all significant

4. Model validation. How do the residuals look?

5. Report results and plot your modeled lines (see below if you need help with last step).

```
# Make new X data
min(birds$Area)
max(birds$Area)
newX<-expand.grid(Area = seq(from=0, to=160, by=1),
                  Country =rep(levels(birds$Country)))
newX
# Predictions using model [bird1 is the name of my model – what did you call yours?]
newY<-predict(bir1, newdata=newX, type="response", se=T)
# Confidence intervals
pred.plus<-newY$fit+2*newY$se
pred.minus<-newY$fit-2*newY$se

# Housekeeping
bird.plot.metrics<-cbind(newX, newY, pred.plus, pred.minus)
bird.plot.metrics
head(bird.plot.metrics)

par(mfrow=c(1,1))
plot(Number~Area, data=birds)
lines(fit ~ Area, data=bird.plot.metrics[bird.plot.metrics$Country=="ENGLAND",])
lines(fit ~ Area, data=bird.plot.metrics[bird.plot.metrics$Country=="WALES",], lty=2)
lines(fit ~ Area, data=bird.plot.metrics[bird.plot.metrics$Country=="SCOTLAND",], lty=3)
```