Statistics in R

Mark Puttick (marknputtick@gmail.com)

24/04/2018

Contents

Statistics in R

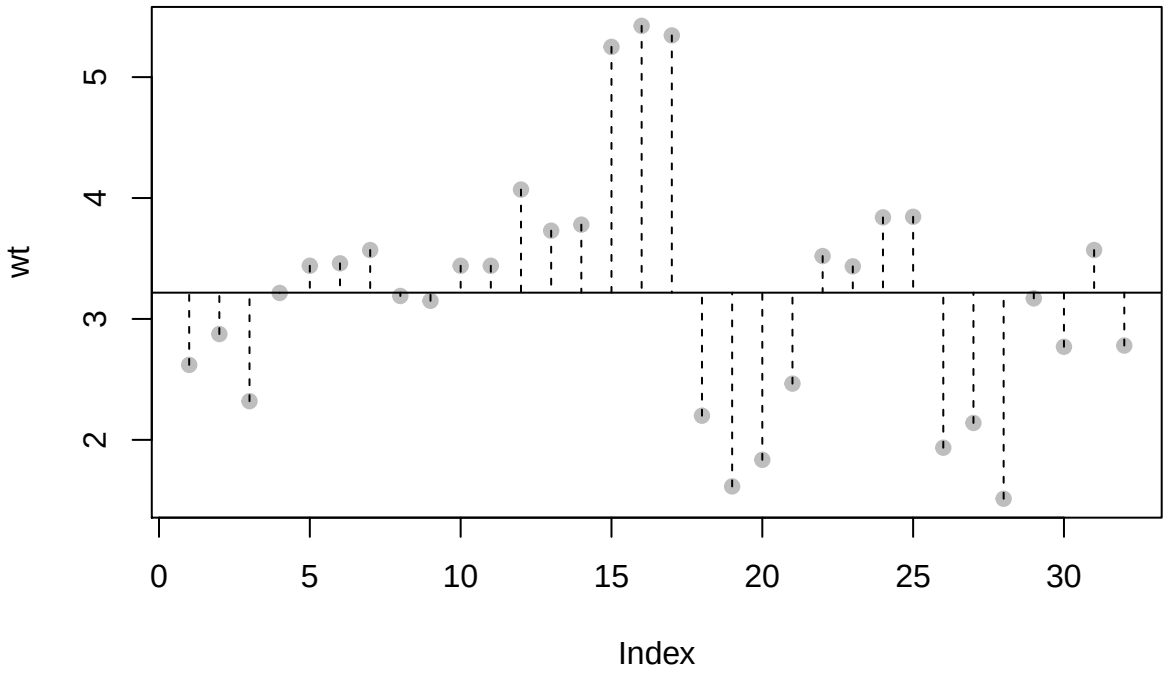
R is to perform flexible and powerful statistical analyses. A huge number of the commonly-used statistical tests, such as a Student's t-test, ANOVA, and regression are all available with the standard R installation. If we require more specialised tests we can generally find these in packages. For example, geometric morphometrics can be carried out using the geomorph package. Throughout this section I will be refer to notes and data from Michael Crawley's book Statistics: An Introduction Using R which is an excellent reference for the do's, and the do-not-do's, of statistics.

This course assumes you have some basic R knowledge, if not you may wish to consult the introduction available here

Variance and standard deviation

Variance is measure of the spread of our data. We can start by calculating it in a straightforward manner. First we will encounter the sum of squares. This is the amount each of our data points deviates from the mean of our data Here we will also introduce some new R notation. Whenever we encounter a # in an R script, R will ignore all that follows so it can be used to add comments to our scripts Here we will use the mtcars data and plot how far each data point falls from the mean in an index plot

```
# read in the data
data(mtcars)
#attach the data
attach(mtcars)
# do an index plot of weight (wt)
plot(wt, pch=19, col="grey")
# add a line to the plot using abline. The argument h stands for horizontal
abline(h=mean(wt))
# use a for loop to add lines from each data point to the mean
for(i in 1:length(wt)) lines(rep(i, 2), y=c(wt[i],
mean(wt)), lty=2)
```



We can now examine the numerical measure of variance, s^2, with

$$s^2 = \frac{\Sigma(\bar{y} - y)^2}{n - 1}$$

We can break this into elements. Firstly, we can calculate the numerator, which is the sum of squares:

$$SS = \Sigma(\bar{y} - y)^2$$

The symbol $\bar{y}$ (the line is called a macron also referred to as "ybar") refers to the mean of our data, y is each element in our data vector. Σ means sum which we can do with `sum` in R

```r
# calculate the sum of squares
sum((mean(wt)-wt)^2)
```

```
## [1] 29.67875
```

We can now concentrate on the second part of the formula by dividing by n-1. Where n refers to our sample size (i.e, the number of measurements we have in wt). Why sample size minus one? This refers to the degrees of freedom of our data. The sum of squares required us to calculate the mean which is technically a parameter of the data not something that is in the original data values. A definition of the degrees of freedom is the sample size of the dataset minus the number of parameters estimated from the data. This means we have one fewer degree of freedom so we divide by *n-1*.

```r
# calculate the variance by dividing by n-1
sum((mean(wt)-wt)^2) / (length(wt) - 1)
```

```
## [1] 0.957379
```

We have the variance of our data.

In future we probably don't want to go through all that hassle. Fortunately, R has an in-built function `var`

```r
# using the inbuilt var function
var(wt)
```

```
## [1] 0.957379
```

Next we can calculate the related measure of standard deviation

$$s = \sqrt{\frac{\Sigma(\bar{y} - y^2)}{n-1}}$$

We can see this is simply the square root of the variance we can do this 'by hand' in R following our calculations of variance above or alternatively use the inbuilt function `sd`.

```r
# standard deviation
sqrt(sum((mean(wt)-wt)^2) / (length(wt) - 1))
```

```
## [1] 0.9784574
```

```r
# or using the in-built R function
sd(wt)
```

```
## [1] 0.9784574
```

Task One

- Calculate the variance and standard deviation of `disp` from `mtcars`.
- Plot an index plot that shows the variance around the mean of `disp`

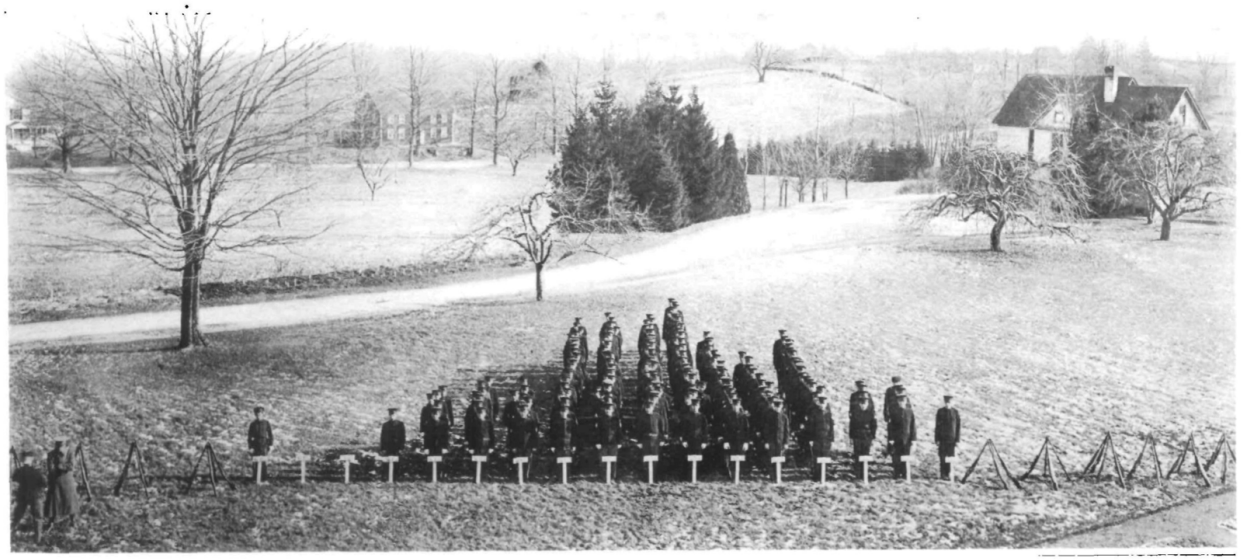
Hypothesis testing with normally distributed data

Choosing the right tests is one of the most confusing aspects of statistical analyses. There are a baffling number of tests to choose from and they have a range of confusing names, such as *Tukey's honest significance test*!

As Michael Crawley points out, we can ask some preliminary questions about data:

Are our data normally distributed? Are there outliers?

If we have normally distributed data and no outliers and no serial correlation then we can use parametric tests. If we invalidate one of these we need to use non-parametric methods.



Number of individuals in each rank	1	0	0	1	5	7	7	22	25	26	27	17	11	17	4	4	1
Heights in feet and inches to which ranks correspond	4:10	4:11	5.0	5:1	5:2	5:3	5:4	5:5	5:6	5:7	5:8	5:9	5:10	5:11	6:0	6:1	6:2

AN INTRODUCTION TO BIOMETRY

Company of students at Connecticut Agricultural College, grouped according to differences in height. The height of each rank, and the number of men in that rank, are shown in the figures below the photograph. The company constitutes what is technically known as a "population" grouped in "arrays of variates;" the middle array or row gives the median or average height of the population. If, now, a line be drawn connecting the upper ends of each row, and another line be drawn along the front rank to serve as a base, the resulting geometric figure forms a "scheme of distribution of variates" or more briefly, a "variability curve," since if the number of arrays were larger, the line joining their tops would form a perfectly smooth curve. The arrangement of homogeneous objects of any kind in such form as this is the first step in the study of variation by modern statistical methods, and on the study of variation much of the progress of genetics rests. (Fig. 12.)

Figure 1: Normal distribution of human height, from Blakeslee 1914

Normal distribution and the central limit theorem

If we repeat sampling of data and take sample means, these means will follow a normal distribution. Let's look an example of normal height using data collected from American men. We have the actual data, but we know some important parameters from it, the mean and standard deviation, as well as the fact it is a normal distribution. So we can simulate a distribution based on these.

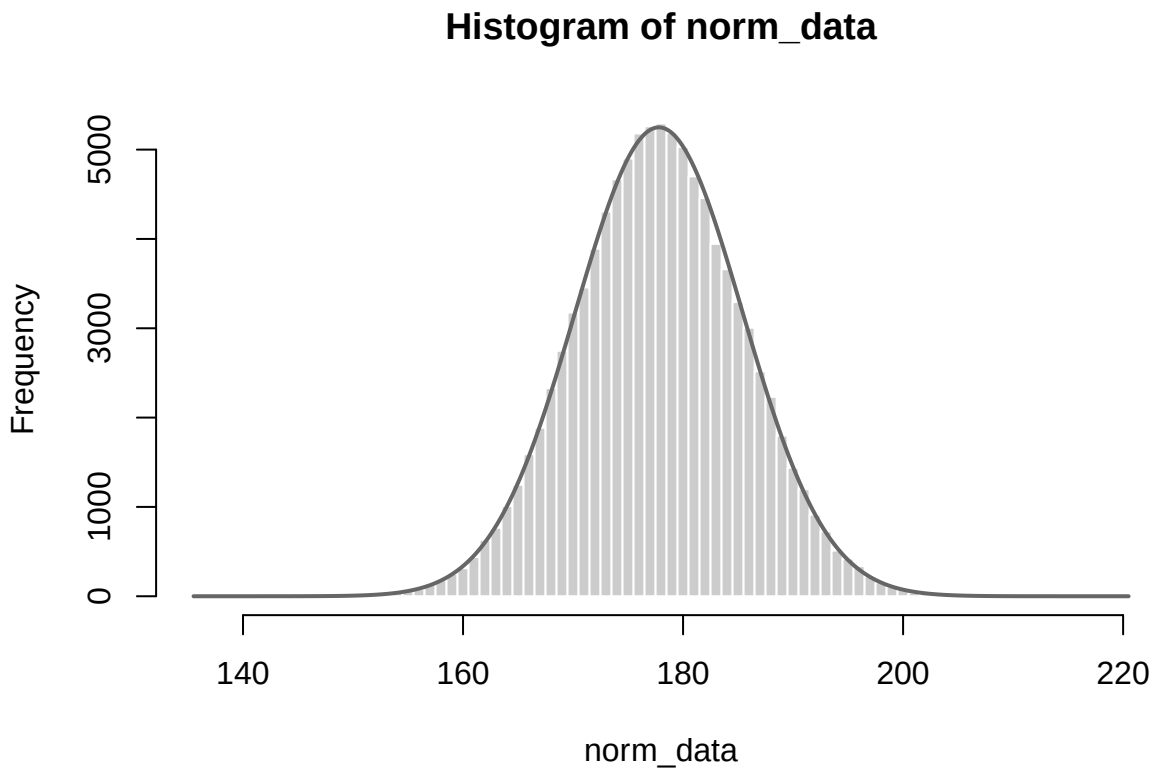
```
# simulate data for a normal distribution using some data for standard american men with a mean height
norm_data <- rnorm(100000, mean=177.8, sd=7.6)
# plot a histogram
hist(norm_data, breaks=135.5:220.5, col="grey80",
border="white")
# confirm our mean and sd our what we expect them to be pretty close!
mean(norm_data)
```

```
## [1] 177.8036
```

```
sd(norm_data)
```

```
## [1] 7.611264
```

```
# add a curve to our normal distribution
heightRange <- seq(135.5, 220.5, by=0.01)
lines(heightRange, 100000 * dnorm(heightRange,
mean=177.8, sd=7.6), lwd=2, col="grey40")
```



This looks good. We can explore some aspects of the normal distribution. For example it is well known that 95% of data values will fall within ± 1.96 standard deviations from the mean. So in our distribution we can see where data falls within one, two and three standard deviations. This is sometimes called the 68-95-99.7% rule

```
# Code inspired by Michael Crawley
# historgram of the data
hist(norm_data, breaks=135.5:220.5, col="white", border="white", main="", xlab="height")
# outline of the bell curve
lines(heightRange, 100000 * dnorm(heightRange, mean=177.8, sd=7.6), lwd=2, col="grey40")

# density of values in the height range
pd <- 100000 * dnorm(heightRange, mean=177.8, sd=7.6)

# heights within one standard deviation
xv_sd_one <- heightRange[heightRange >= 177.8 - 7.6 & heightRange <= 177.8 + 7.6]
yv_sd_one  <- pd[heightRange >= 177.8 - 7.6 & heightRange <= 177.8 + 7.6]
xv_one <- c(xv_sd_one, 177.8 + 7.6, 177.8 - 7.6)
yv_one <- c(yv_sd_one, pd[1], pd[1])

# heights within two standard deviations
xv_sd_two <- heightRange[heightRange >= 177.8 - 2 * 7.6 & heightRange <= 177.8 + 2 * 7.6]
yv_sd_two <- pd[heightRange >= 177.8 - 2 * 7.6 & heightRange <= 177.8 + 2 * 7.6]
xv_two <- c(xv_sd_two, 177.8 + 2 * 7.6, 177.8 - 2 * 7.6)
yv_two <- c(yv_sd_two, pd[1], pd[1])

# heights within three standard deviations
xv_sd_three <- heightRange[heightRange >= 177.8 - 3 * 7.6 & heightRange <= 177.8 + 3 * 7.6]
yv_sd_three <- pd[heightRange >= 177.8 - 3 * 7.6 & heightRange <= 177.8 + 3 * 7.6]
xv_three <- c(xv_sd_three, 177.8 + 3 * 7.6, 177.8 - 3 * 7.6)
```

```r
yv_three <- c(yv_sd_three, pd[1], pd[1])

# polygon of heights within three standard deviations
polygon(xv_three, yv_three, col="green4", border=NA)
# polygon of heights within two standard deviations
polygon(xv_two, yv_two, col="navy", border=NA)
# polygon of heights within one standard deviations
polygon(xv_one, yv_one, col="grey", border=NA)

# legend
legend("topleft", c("one SD from mean", "two SD mean", "three SD from mean"), bty="n", lwd=2, col=c("gr
```

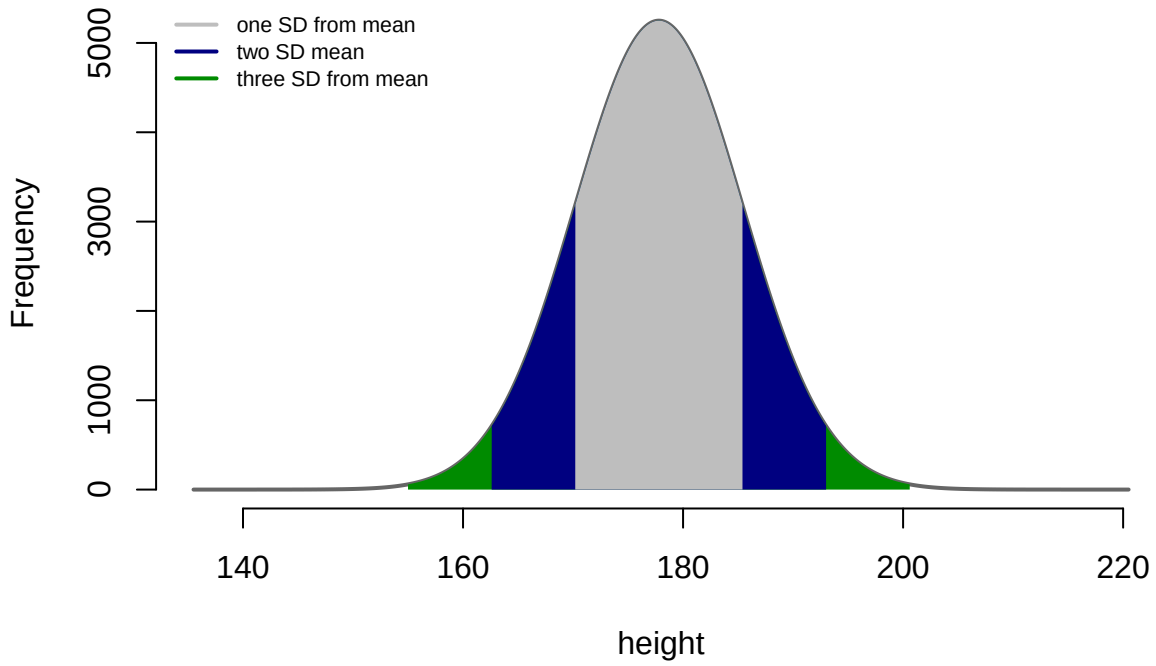


Figure 2: Shaquille O'Neal (left)

Hypothesis testing

We have these data but we probably want to know the probability of someone falling outside this size range.

Let's choose an American male: former basketball player Shaquille O'Neal. According to Wikipedia he measures 216 cm - Ok is he significantly different from our population?!

First let's find convert his height into a z score by determining how many standard deviations his height falls from our mean.

$$z = \frac{y - \bar{y}}{s}$$

This simply translates to subtracting the population mean (177.8) from O'Neal's height (216) and dividing by the standard deviations that falls from the mean. We can do this in R (as you may have guessed)

```
# find the z score
(216-177.8)/7.6
```

```
## [1] 5.026316
```

Ok – his height is whopping 5.02 standard deviations from the mean! We can use `pnorm` to get a significance value to determine if O'Neal is significantly taller than the distribution of heights

```
# what is the probability of being as tall as the "Shaq Attack"?
zscore <- (216-177.8)/7.6
pnorm(zscore)
```

```
## [1] 0.9999998
```

Huh? That seems odd. We could read this as the probability of finding someone as tall as O'Neal is nearly 1. However, it is actually the cumulative probability of being shorter than O'Neal, so this number being large makes more sense. To p value for Shaq being significantly different from the population

```
1-pnorm(zscore)
```

```
## [1] 2.499962e-07
```

If we take a standard significance of < 0.05 this is hugely significantly different from our sample! We can confidently say Shaquille O'Neal is tall without fear of being called out.

Task Two

- Calculate the p value for a height being found in-between your height and Shaq O'Neal's (I am assuming your height is different! If not choose a friend's height)
- Calculate whether your height differs significantly from the mean

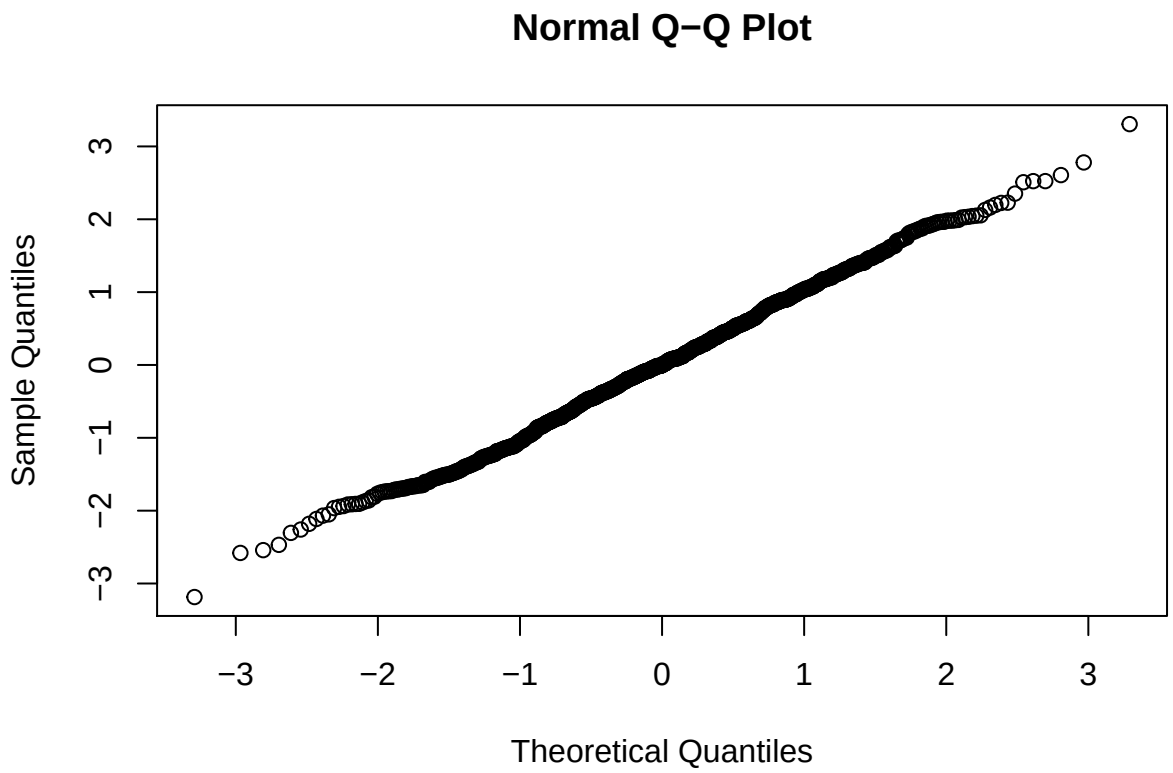
Non-normality

Parametric tests (the ones we have looked at above) assume distributions of the data. I.e, they assume that the data are normally distributed. However, non-parametric tests make fewer assumptions of the data. So if are data are non-normal it is better to use non-paramteric tests. Here we our going to use one of our example datasets, Michelson's 1880 experiments to determine the speed of light

```
light <- read.csv("https://puttickbiology.files.wordpress.com/2018/04/light.odt")
attach(light)
```

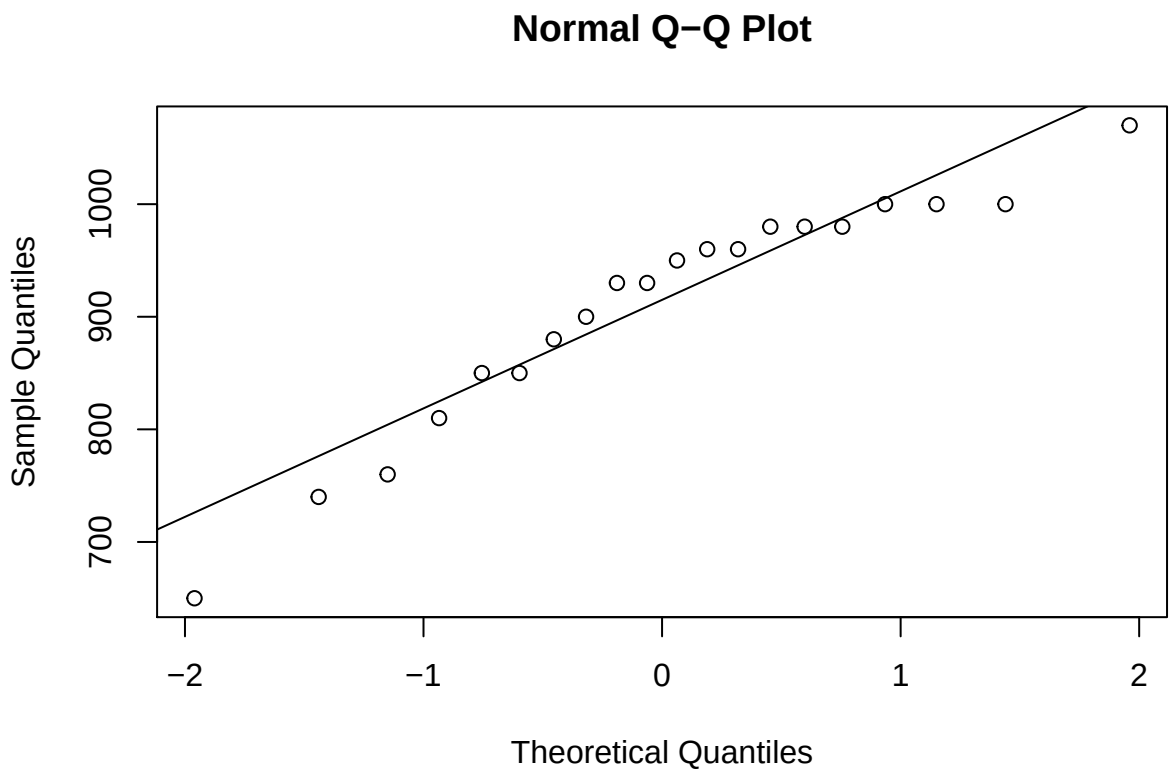
One of the best ways to get an idea of normality is to plot a quantile-quantile plot. On the y axis is the ranked scores of our data and on the x axis is the theoretical values we would expect these values to take were they from a normal distribution. If data are normal we should see a straight line. For example, using our height data from above:

```
# quantile-quantile plot for normal data
qqnorm(rnorm(1000))
```



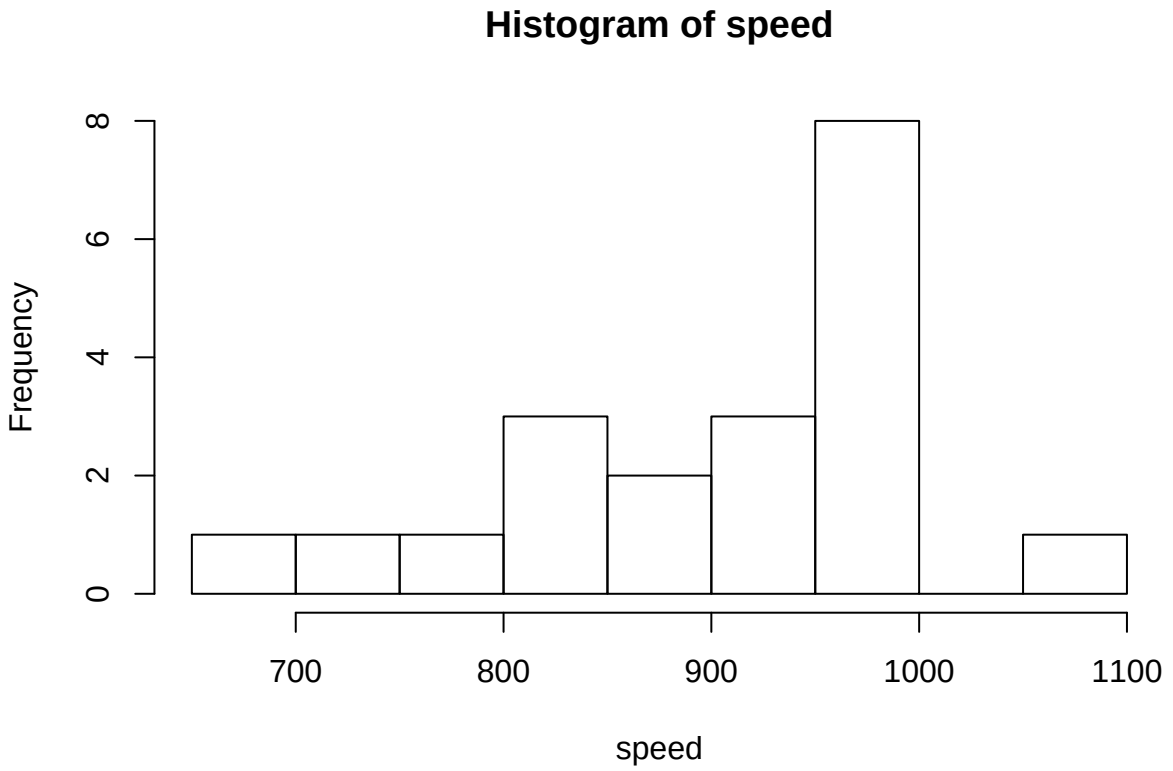
Let's try for our light data

```
qqnorm(speed)
qqline(speed)
```



It's ok but seems to be diverging from what we want it to be. It isn't straight enough. Let's investigate why with a histogram

```
hist(speed)
```



We can see that there is a skew to the left of the plot. This is known as a negative skew. Therefore, our data

are not normal so we need to employ a non-parametric test statistic The light data here is from Michelson's experiments so we wish to test if this is significantly different from the 299 990 (all data have had 290 000 subtracted from them)

We will use Wilcoxon's signed rank test

```
wilcox.test(speed, mu=990)
```

```
## Warning in wilcox.test.default(speed, mu = 990): cannot compute exact p-
## value with ties
```

```
##
##  Wilcoxon signed rank test with continuity correction
##
## data:  speed
## V = 22.5, p-value = 0.00213
## alternative hypothesis: true location is not equal to 990
```

The null hypothesis of light being 990 is rejected ($p < 0.05$), we can accept the hypothesis that light is significantly slower than 990.

Paired samples

If we measure samples from the same distribution at different times, we consider them to be paired. For example, to we can repeat the Wilcoxon's signed rank test above with a paired example. If we measure pollution in a city on different days we would consider this paired: it is the same city, same environment, same local conditions, etc. If we have pollution measurements on a day with no traffic and with traffic

```
traffic <- c(214, 159, 169, 202, 103, 119, 200, 109, 132,
142, 194, 104, 219, 119, 234)
noTraffic <- c(159, 135, 141, 101, 102, 168, 62, 167,
174, 159, 66, 118, 181, 171, 112)
```

These data are considered paired, but is there a significant difference in the level of pollution?

```
wilcox.test(traffic, noTraffic, paired=T)
```

```
##
##  Wilcoxon signed rank test
##
## data:  traffic and noTraffic
## V = 80, p-value = 0.2769
## alternative hypothesis: true location shift is not equal to 0
```

There is no significant difference between days with traffic and without on local pollution

Two samples

We will again use some data. This time, we utilise data from ozone levels from some gardens.

```
fTestData <- read.csv("https://puttickbiology.files.wordpress.com/2018/04/f-test-data.odt")
attach(fTestData)
names(fTestData)
```

```
## [1] "gardenB" "gardenC"
```

Let's get the variances of the different gardens (B and C):

```
var(gardenB)
```

```
## [1] 1.333333
```

```
var(gardenC)
```

```
## [1] 14.22222
```

We will use the F test that is simply the ratio of two samples given by dividing the larger variance with the smaller one

```
F.ratio <- var(gardenC) / var(gardenB)
F.ratio
```

```
## [1] 10.66667
```

What does this mean? On its own, not a lot. We need to look up the appropriate value of F to judge our test statistic. How many degrees of freedom to we have? First, let's get the length of our data, using `length` that measures the number of elements in an object to give its length (I may have over-complicated things)

```
length(gardenB)
```

```
## [1] 10
```

```
length(gardenC)
```

```
## [1] 10
```

We are estimated one parameter from each, the variance, so we have *10-1*, or *9*, degrees of freedom. We can now use the the function `qf` to detail get the appropriate F ratio:

```
qf(p=0.975, df1=9, df2=9)
```

```
## [1] 4.025994
```

Where did the p value come from? We are probably familiar with a significance value of $p = 0.05$ (also known as alpha = 0.05). As we are performing a two-tailed test, we allow either garden to potentially have higher or lower variance. So this means

```
1-(0.05/2)
```

```
## [1] 0.975
```

Ok so we now know the appropriate F value for our test.

```
1-(0.05/2)
```

```
## [1] 0.975
```

Recall

```
F.ratio
```

```
## [1] 10.66667
```

The null hypothesis is rejected as we can see that our F value is larger than the critical value. We can then get a p value for this

```
2*(1-pf(F.ratio, 9, 9))
```

```
## [1] 0.001624199
```

Excellent - the F test of variance is done. Unsurprisingly R has a function built-in to do this. We could have used and seen that the variances differ significantly between the two gardens

```
var.test(gardenB, gardenC)
```

```
##
##  F test to compare two variances
##
## data:  gardenB and gardenC
## F = 0.09375, num df = 9, denom df = 9, p-value = 0.001624
## alternative hypothesis: true ratio of variances is not equal to 1
## 95 percent confidence interval:
##  0.02328617 0.37743695
## sample estimates:
## ratio of variances
##            0.09375
```

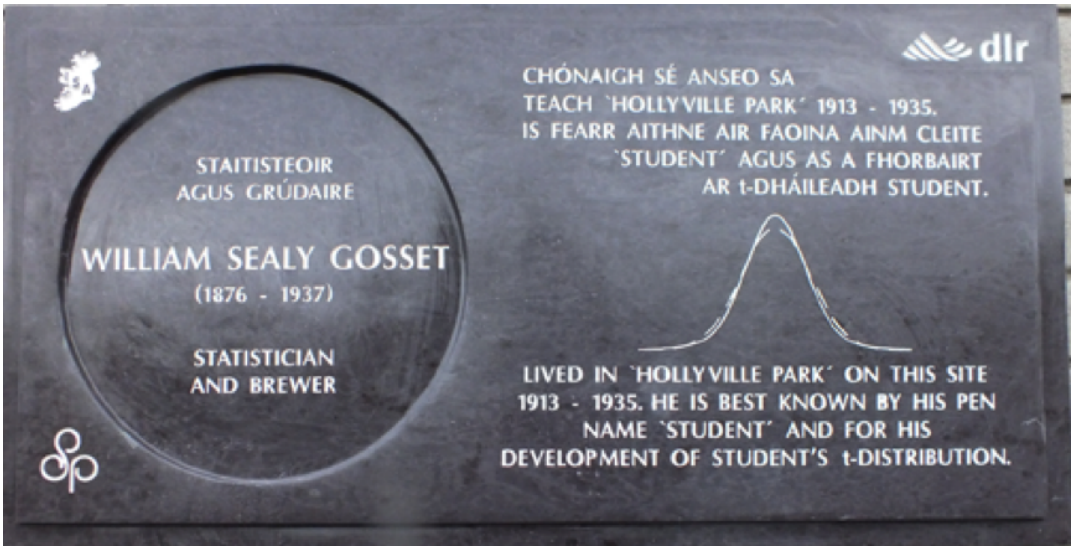


Figure 3: William Sealy Gosset, part brewer, part statistician, perpetual student

Student's t test

The two-sample Student's t-test is a classical statistical tool to compare whether a small sample from two independent distributions differ in location of the mean.

William Sealy Gosset developed the Student's t test when he was an employee of Guinness in Dublin. Due to some employment laws he was not allowed to publish the findings under his own name, so used the name 'Student' instead. So all thanks should go to stout quality-control procedures for making this section possible.

The t statistic of the Student's t test is calculated as follows:

$$t = \frac{difference\,between\,two\,means}{standard\,error\,of\,the\,difference}$$

It is straightforward to calculate the numerator. It is simply the mean of dataset one minus mean of dataset two. The standard error is the square root of the variance divided by the sample size. In independent samples we can calculate this the denominator, standard error of the difference as

$$SEdiff = \sqrt{\frac{sA^2}{nA} + \frac{sB^2}{nB}}$$

To some data!

```
t.test <-
read.csv("https://puttickbiology.files.wordpress.com/2018/04/t-test-data.odt")
attach(t.test)
```

```
## The following object is masked from fTestData:
##
##     gardenB
```

```
names(t.test)
```

```
## [1] "gardenA" "gardenB"
```

An assumption for the *Student's t test* is that variances do not differ significantly between two samples: we are testing differing locations of the mean, not differing variance

```
var(gardenA) / var(gardenB)
```

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

The variance is equal to one and so is definitely not significantly different

The top bit of the equation – difference in the means – is relatively simple

```
mean(gardenA) - mean(gardenB)
```

```
## [1] -2
```

and the second bit isn't too much harder

```
sqrt(var(gardenA)/length(gardenA)+var(gardenB)/length(gardenB))
```

```
## [1] 0.5163978
```

and put them together

```
(mean(gardenA) - mean(gardenB)) /
sqrt(var(gardenA)/length(gardenA)+var(gardenB)/
length(gardenB))
```

```
## [1] -3.872983
```

Now we have the t statistic, we also need to know the critical value given our data. First we need the degrees of freedom: we have 10 samples in each garden, and as we estimate the mean from both. So we have a sample size of data from both gardens and *10-n* degrees of freedom from each as we estimate the mean. So we have $df = 18$. We perform a two-tailed testas we don't know which garden has the larger value.

```
qt(p=0.975, df=18)
```

```
## [1] 2.100922
```

Our test statistic is larger than the critical value (we ignore the minus sign). So we can say that that the means are significantly different. The p value is

```
2*pt( -3.872983, 18)
```

```
## [1] 0.00111454
```

Again we could have done all this using

```
t.test(gardenA, gardenB)
```

```
##
##  Welch Two Sample t-test
##
## data:  gardenA and gardenB
## t = -3.873, df = 18, p-value = 0.001115
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -3.0849115 -0.9150885
## sample estimates:
## mean of x mean of y
##         3         5
```

We can now understand where the t and p value have come from, and again show this is a significant difference

Task Three

- The mean height for men is 177.8 cm with a standard deviation of 10.2 cm. The mean height for women is 165.1 with a standard deviation of 8.9
- Assume a normal distribution and simulate 10 000 variables for each
- Test if the variance is equal between samples
- Use a Student's t test to see if the two samples are significantly different

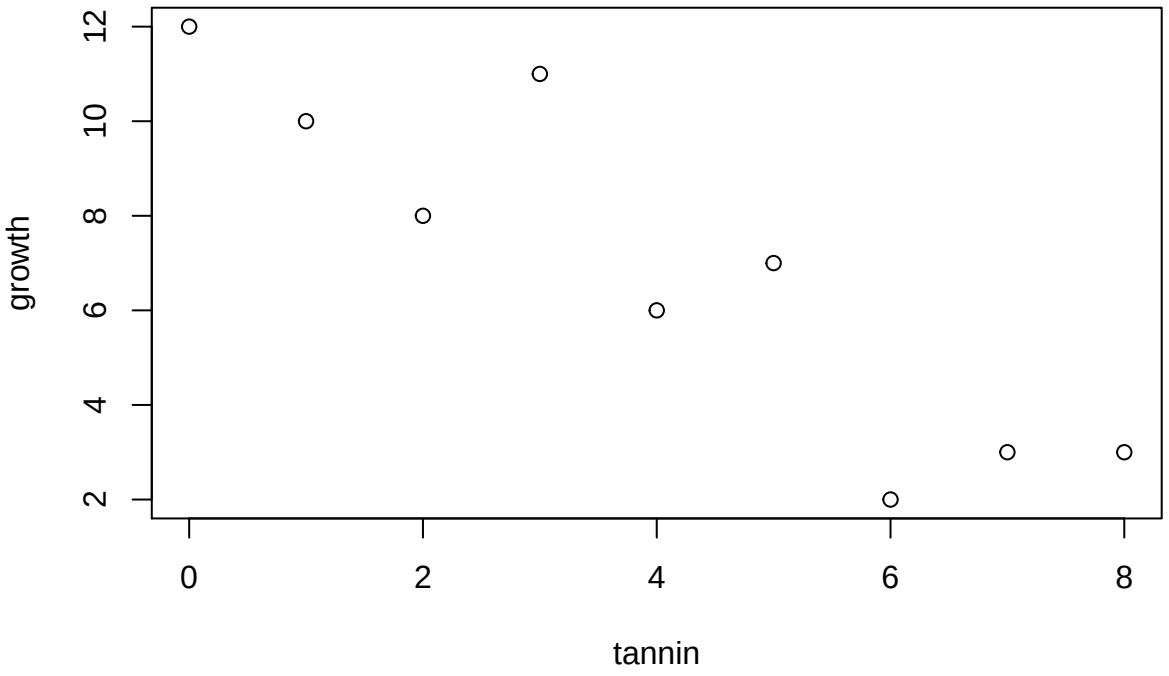
Regression

When we have two continuous variables we may wish to learn how one source of data affects another. We call one the explanatory variables which normally go on the x axis and a response variable which is plotted on the y axis. Ok, let's read some data

```
reg.data <- read.csv("https://puttickbiology.files.wordpress.com/2018/04/tannin.odt")
attach(reg.data)
names(reg.data)
```

```
## [1] "growth" "tannin"
```

```
plot(tannin, growth)
```



We can perform a simple linear regression. We will get two main parameters as output from this: the intercept and the slope. The parameters of the slope, b, and the intercept, a

$$y = a + bx$$

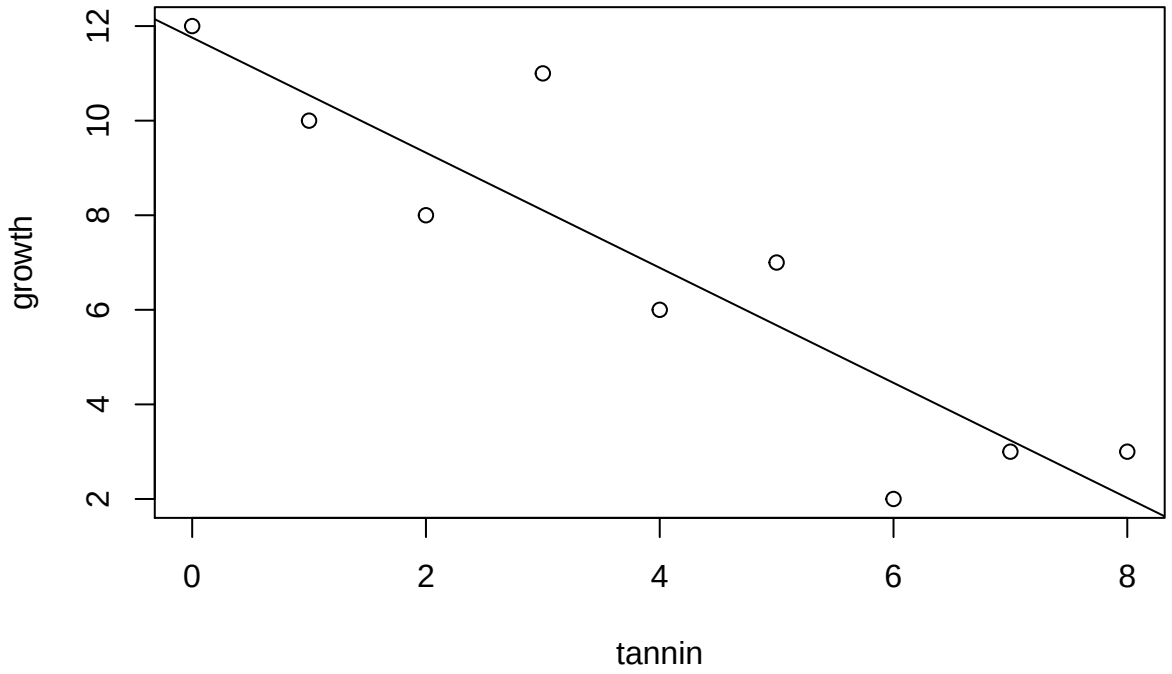
Here we see some new R notation. We use the symbol called tilde `~` to signify y as a function of x, i.e `y~x`. In this example growth is the response and tannin is the explanatory variable.

```
lm(growth~tannin)
```

```
##
## Call:
## lm(formula = growth ~ tannin)
##
## Coefficients:
## (Intercept)       tannin
##      11.756       -1.217
```

Great so we have parameters a and b. We can also overlay this on our plot using abline

```
plot(growth~tannin)
abline(lm(growth~tannin))
```



Note The slope parameter is calculated by minimizing the residual sum of error. I won't cover that here but a proof using some calculus is shown in Basic Statistics: An Introduction Using R (page 118)

Technically here we have identified the maximum likelihood estimate of the slope. That is, the model parameters (a, b) that will make the data (x, y) most likely.

Using the `lm` function we can learn a lot more from the regression model fit using summary

```
reg_model <- lm(growth~tannin)
summary(reg_model)
```

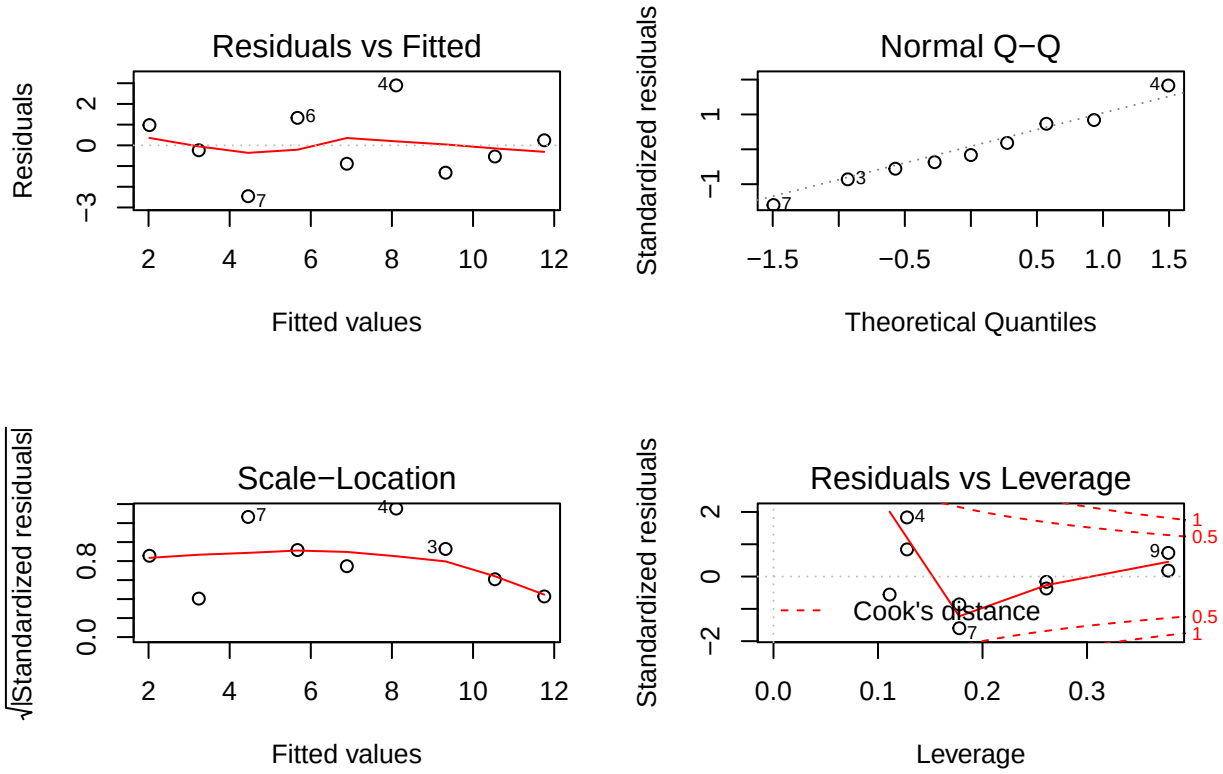
```
##
## Call:
## lm(formula = growth ~ tannin)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.4556 -0.8889 -0.2389  0.9778  2.8944
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  11.7556     1.0408  11.295 9.54e-06 ***
## tannin       -1.2167     0.2186  -5.565 0.000846 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.693 on 7 degrees of freedom
## Multiple R-squared:  0.8157, Adjusted R-squared:  0.7893
## F-statistic: 30.97 on 1 and 7 DF,  p-value: 0.0008461
```

Here we see the intercept and slope coefficients as well as their standard error and t values. The low values in the "Pr" column indicate the model parameters are very significant. The R squared value is of particular

interest as this tells us the propoprtion of our data in the response (y) is explained by the model. It can vary between 0 and 1. 0 means none of the variation is explained, and 1 means it is all explained perfectly. The bottom right p value refers to the *F test* that was performed that the overall model was significant.

An important aspect of regression models is to test that the data our suitable for regression. We can do model-checking by assessing the consistency of variance and the normality of errors.

```
par(mfrow=c(2,2))
plot(reg_model)
```



Below we see residuals against fitted values. We want these to be 'random' or in a more poetical phrase, like a clear starry night. We certainly don't want residuals to get bigger as fitted values get bigger. We met the `qqplot` before and it looks good here. The bottom left plot shows the same as the first but with positive square roots of the residuals. Here we do not want a triangle shape. The final plot shows the effects of individual data points on our estimates. It may show data points with large residuals may be having a large effect on the model. Cook's distance sounds like a holiday package but is actually the difference between the sum of squares with all predictors and with one predictor removed – a Cook's distance greater than one may mean this predictor is too influential (although this is a case-by-case judgement call). Leverage is similar in that it measures the distance of a predictor variable x from the mean all the predictor variables. If a variable has too high an influence it may need to be removed from the data, or indicate that logscaling the data would be better.

Note There are other forms of regression available in R:

- Generalised Least Squares. Here we have looked at ordinary least squares regression (OLS) in which the error variance is assumed to be normally distributed. We can fit generalized least squares (GLS) that takes the assumption that the error variance in our model is not normally distributed. However, if we have autocorrelation in our data or heteroscedasticity in which the error variance is not equal across our model. GLS models can be fit with the gls function in the nlme package
- Generalised Linear Models. Generalised Linear Models (GLMs) are used in situations in which we do not have a continuous response variable. In these cases we fit a number of link functions that specify

the distribution of the response, such as binomial function for binary data and a poisson function for counts. In R we can use the `glm` function.

- Generalised Additive Models. Generalised Additive Models (GAMs) are appropriate when we have a non-linear relationship between variables. In this case for our response we fit a non-parametric distribution to the shape of our response when we have no *a-priori* expectations of this relationship. They can be fit with the `gam` function from the mgcv package.

Task Four

- Using these data below check to for the assumptions of, and perform, an OLS regression

```
data.frame(lnLength = c(3.87, 3.61, 4.33, 3.43, 3.81, 3.83,3.46, 3.76,3.50, 3.58, 4.19, 3.78, 3.71, 3.73
lnWeight = c(4.87, 3.93, 6.46, 3.33, 4.38, 4.70,3.50, 4.50,3.58, 3.64, 5.90, 4.43, 4.38, 4.42, 4.25))
```

```
##    lnLength lnWeight
## 1      3.87     4.87
## 2      3.61     3.93
## 3      4.33     6.46
## 4      3.43     3.33
## 5      3.81     4.38
## 6      3.83     4.70
## 7      3.46     3.50
## 8      3.76     4.50
## 9      3.50     3.58
## 10     3.58     3.64
## 11     4.19     5.90
## 12     3.78     4.43
## 13     3.71     4.38
## 14     3.73     4.42
## 15     3.78     4.25
```

ANOVA

ANOVA, or Analysis Of Variance, is a situation in which we have a continuous response variable and multiple categorical explanatory variables. Each explanatory variable is now as a factor. In a one-way anova there is a single factor with multiple levels. Suppose a farmer compares wheat growth in three areas: "sunny field", "hill field", "nothing ever grows field". This means we have one factor, area, with three levels of different types of fields.

A two-way anova is used when there are multiple factors. Suppose the famer tries using no fertiliser and fertiliser on the fields – this is a new factor fertilisation with two levels – fertiliser and no fertiliser.

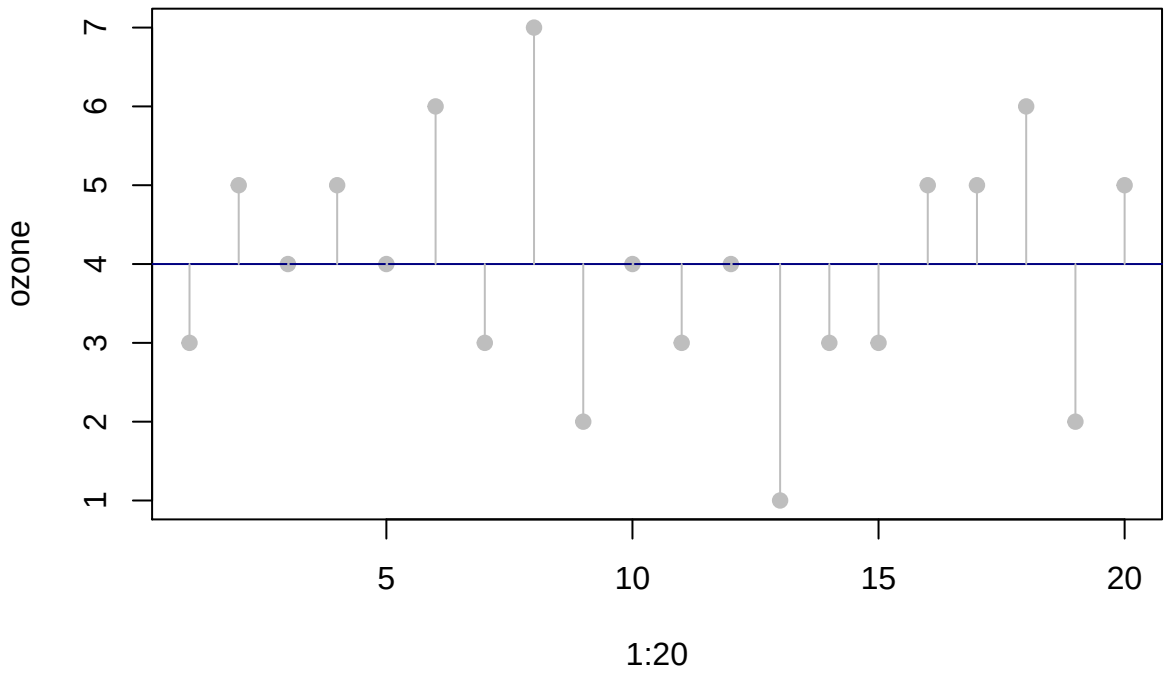
One-way ANOVA

In this case we have one factor and three or more levels. A model with two models is simply a *t test*. Essentially with an ANOVA we are searching whether a model with a single mean between levels can be rejected in favour of a model with more than one mean.

```
oneWay <- read.csv("https://puttickbiology.files.wordpress.com/2018/04/oneway.odt")
attach(oneWay)
names(oneWay)
```

```
## [1] "ozone"  "garden"
```

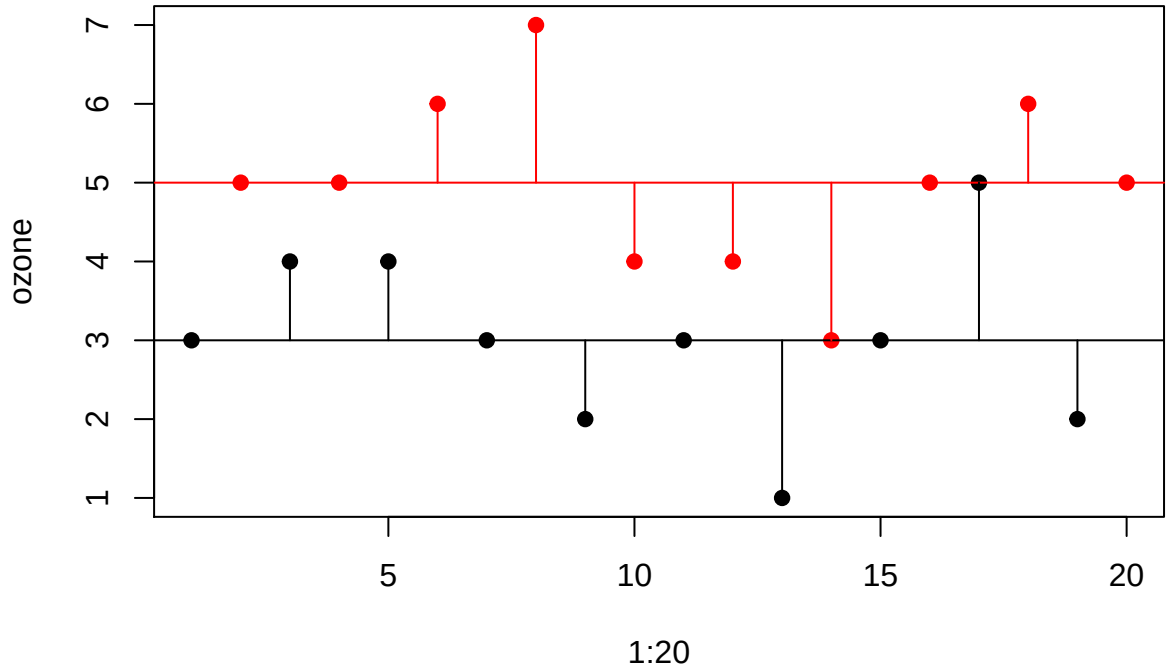
```
plot(1:20, ozone, pch=19, col="grey")
abline(h=mean(ozone), col="navy")
for(i in 1:20) lines(c(i,i), c(mean(ozone), ozone[i]), col="grey")
```



We can also plot a figure with both means separated

```
plot(1:20, ozone, pch=19, col=as.numeric(garden))
abline(h=mean(ozone[garden == "A"]), col="black")
abline(h=mean(ozone[garden == "B"]), col="red")
for(i in 1:length(ozone)) {
if(garden[i] == "A") lines(c(i,i), c(mean(ozone[garden ==
```

```
"A"]), ozone[i]), col="black")
else
lines(c(i,i), c(mean(ozone[garden == "B"]), ozone[i]),
col="red")
}
```



Basically with an ANOVA we want to know if the residuals on the second plot are smaller the on the first plot with two means. So in our example dataset, does ozone level differ between gardens. Let's do this backwards as to how we did this before. By using the R way first!

```
summary(aov(ozone ~ garden))
```

```
##             Df Sum Sq Mean Sq F value  Pr(>F)
## garden       1     20  20.000      15 0.00111 **
## Residuals   18     24   1.333
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Your eye may be drawn to the *p value* which shows the model is significant and we can reject the null hypothesis. We can get some more information from the summary of the linear model

```
summary.lm(aov(ozone ~ garden))
```

```
##
## Call:
## aov(formula = ozone ~ garden)
##
## Residuals:
##    Min     1Q Median     3Q    Max
##     -2     -1      0      1      2
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   3.0000     0.3651   8.216 1.67e-07 ***
```

```
## gardenB       2.0000     0.5164   3.873  0.00111 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.155 on 18 degrees of freedom
## Multiple R-squared:  0.4545, Adjusted R-squared:  0.4242
## F-statistic:    15 on 1 and 18 DF,  p-value: 0.001115
```

R takes some interpreting here. The first estimate is GardenA as it is ordered in alphabetical order and the second mean is the distance of this true mean from the first mean. Here it is $5 = 3 + 2$

And we can check this

```
tapply(ozone, garden, mean)
```

```
## A B
## 3 5
```

Excellent – all makes sense! In R a two way analysis can be conducted with the following:

```
# anovaMod <- aov(lm(response ~ factorOne * factorTwo))
# anova(anovaMod)
```

Feel free to explore this `?aov` in R. Also try a *post-hoc* test to determine which levels differ with a *Tukey's Honest Significant Difference test*!

Ancova

The Analysis of Covariance (ANCOVA) is a great way to explore complex relationships between data. It is very similar to a regression: we have a continuous response and a continuous explanatory variable (possibly more than one). We estimate a slope and intercept as with the regression analysis, but we can fit multiples of these relationships as dictated by a categorical response variable. We will take the iris dataset in R and for simplicity compare the sepal length as a function of sepal width for two groups of species. Group "a" is the species *versicolor* and *virginica* and group "b" is *setosa*.

```
attach(iris)
names(iris)
```

```
## [1] "Sepal.Length" "Sepal.Width"  "Petal.Length" "Petal.Width"
## [5] "Species"
```

```
group <- rep("a", length(Species))
group[which(Species == "setosa")] <- "b"
group <- factor(group)
```

We can fit our first model including the interaction term group. The symbol '*' indicates we wish to estimate a separate regression slope and intercept for groups a and b.

```
modelOne <- lm (Sepal.Length ~ Sepal.Width * group)
summary(modelOne)
```

```
##
## Call:
## lm(formula = Sepal.Length ~ Sepal.Width * group)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.00322 -0.27224 -0.01355  0.22518  1.73809
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)           3.0934     0.4144   7.465 6.95e-12 ***
## Sepal.Width           1.1033     0.1433   7.697 1.93e-12 ***
## groupb               -0.4544     0.7431  -0.612   0.5418
## Sepal.Width:groupb   -0.4128     0.2292  -1.801   0.0738 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4746 on 146 degrees of freedom
## Multiple R-squared:  0.6781, Adjusted R-squared:  0.6715
## F-statistic: 102.5 on 3 and 146 DF,  p-value: < 2.2e-16
```

It seems the intercept differs between our groups but the slope (Sepal.Width:group) does not ($p = 0.07$). Therefore, we can simplify our model to include only a different slope. We do this by changing the * to a +

```
modelTwo <- lm (Sepal.Length ~ Sepal.Width + group)
summary.aov(modelTwo)
```

```
##              Df Sum Sq Mean Sq F value Pr(>F)
## Sepal.Width   1   1.41    1.41   6.176 0.0141 *
## group         1  67.14   67.14 293.613 <2e-16 ***
## Residuals   147  33.61    0.23
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Is this significantly different from the more complex model. We can use a ANOVA to test this

```
anova(modelOne, modelTwo)
```

```
## Analysis of Variance Table
##
## Model 1: Sepal.Length ~ Sepal.Width * group
## Model 2: Sepal.Length ~ Sepal.Width + group
##   Res.Df    RSS Df Sum of Sq      F  Pr(>F)
## 1    146 32.884
## 2    147 33.615 -1  -0.73047 3.2431 0.07379 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

There is no significant difference between the models so we accept the simpler model: modelTwo

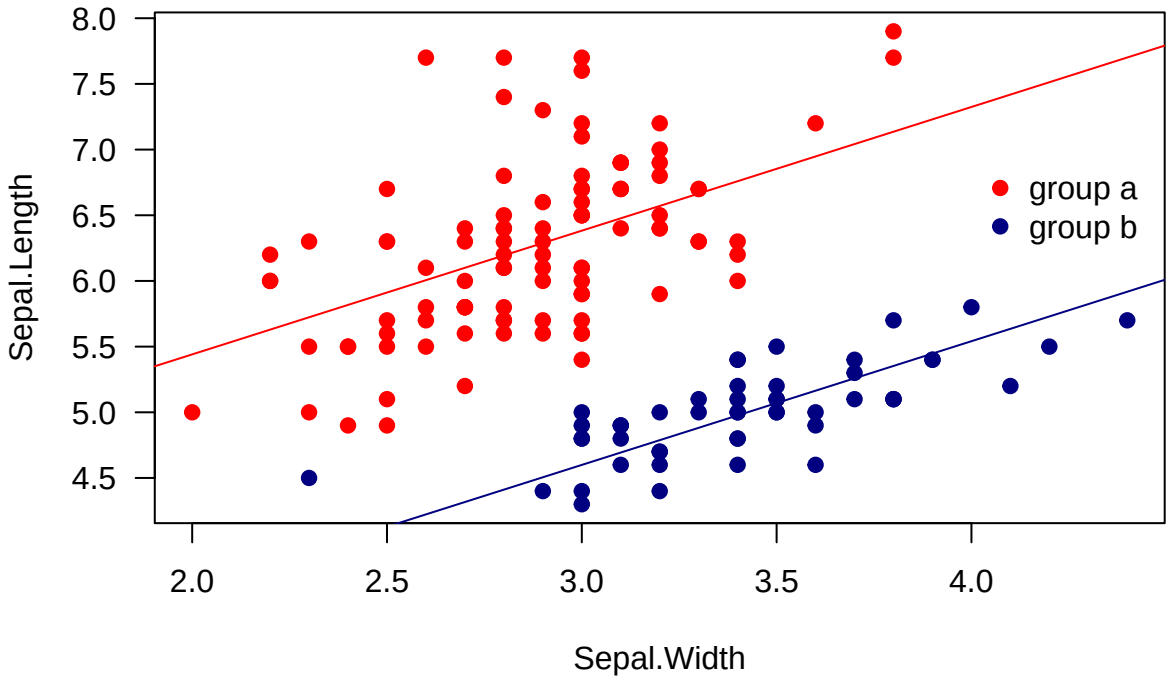
```
summary(modelTwo)
```

```
##
## Call:
## lm(formula = Sepal.Length ~ Sepal.Width + group)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.01164 -0.30001 -0.05054  0.25454  1.69418
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   3.5571     0.3272  10.871  < 2e-16 ***
## Sepal.Width   0.9418     0.1127   8.357 4.47e-14 ***
## groupb       -1.7797     0.1039 -17.135  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4782 on 147 degrees of freedom
## Multiple R-squared:  0.671,  Adjusted R-squared:  0.6665
## F-statistic: 149.9 on 2 and 147 DF,  p-value: < 2.2e-16
```

This shows us the slope for both groups (Sepal.Width) and the intercept for group a (Intercept). The intercept for group b is the sum of intercept for group a and the row marked *groupb*: 3.5571 + -1.7797 = 1.774

We can use these parameters to plot our data with the same slope with different intercepts:

```
plot(Sepal.Length ~ Sepal.Width , col=c("red",
"navy")[group], pch=19, las=1)
abline(3.5571, 0.9418, col="red")
abline(1.774, 0.9418, col="navy")
legend(4, 7, c("group a", "group b"), col=c("red",
"navy"), pch=19, bty="n")
```


8.0
7.5
7.0
6.5
6.0
5.5
5.0
4.5
Sepal.Length
2.0
2.5
3.0
3.5
4.0
Sepal.Width
group a
group b

Chi squared analysis

Task Five

- Complete the table in the following exercise (using R)

Typically we use this when we have count data. We use contingency tables to show how many times a particular event occurred. For example we can look at the relationship between hair colour and eye colour. In this example we have four possible outcomes from this (we have simplified this example to two types of hair colour and eye colour respectively)

	blue eyes	brown eyes	row totals
light hair			
dark hair			
column totals			

Here we have observed counts of when a situation has occurred in our sample. We need to know the chance of observing these data – that is we need to develop a model for these data. So we can develop a system of row and column totals

	blue eyes	brown eyes	row totals
light hair	38	11	49
dark hair	14	51	65
column totals	52	62	114

Ok our model is going to be simple (probably too simple) in that we are going to say that hair colour and eye colour are entirely independent. But this is useful as we can make our hypothesis falsifiable – ie., they are not independent.

If the two are independent we can say that the chance of getting either is entirely independent. So we can say there is an independent probability of having blue eyes and fair hair and work out the expected probabilities as follows

	blue eyes	brown eyes	row totals
light hair	52/114 * 49/114		
dark hair			
column totals	52	62	114

You can fill in the rest of table

As the row totals and column totals are the same (i.e., they equal the grand total) we can construct a formula. The expected frequency in each cell is the row total (R) multiplied by the column total (C) dividing by the grand total (G)

$$E = \frac{R * C}{G}$$

You can now calculate and complete the expected values in the table below:

blue eyes brow	n eyes row	totals	
light hair	22.35		
dark hair			
column totals	52	62	114

Now to our test statistic. We calculate our test statistic as the sum of the squared differemce between the observed and expected values divided by the expected values

$$x^2 = \Sigma \frac{(O-E)^2}{E}$$

The value x^2 is Pearson's chi-squared value. We now reach the the stage at which we can calculate x^2. We can do this by looking up (or using R) to assess the appropriate criticial value from a contingency table. We need to decide – the degree of certainty we wish to have and the degrees of freedom with which we will work. We can say the *df* is the product of number of rows -1 and the number of columns – 1. We will set our certainty level as alpha = 1 as we met this earlier. This states that the chance of rejecting the hypothesis when the null is true (everything in statistics seems backwards!). Now we can return to R to find this appropriate value.

```
qchisq(0.95, 1)
```

```
## [1] 3.841459
```

If we wanted to be super-certain we could have done

```
qchisq(0.99, 1)
```

```
## [1] 6.634897
```

We now need to calculate our test statistic from the table. We can do this in R

```
# establish our table
count <- matrix(c(38, 14, 11, 51), nrow=2)
# get the row and column totals
row_total <- rowSums(count)
col_total <- colSums(count)
grandTotal <- sum(col_total)
# ok we can now work out our expected values build expectation table
exp <- matrix(NA, nrow=2, ncol=2)
for(i in 1:2) exp[1,i] <- row_total[1] *
col_total[i]/grandTotal
for(i in 1:2) exp[2,i] <- row_total[2] *
col_total[i]/grandTotal
# we can take each matrix away from each other
# we will use a loop within a loop to iterate through both matrices
oMinuse <- c()
counter <- 1
for(i in 1:2){
  for(u in 1:2){
    oMinuse[counter] <- ((count[i,u] -exp[i,u])^2)/exp[i, u]
    counter <- counter + 1
    }
  }
# x-squared test statistics
x_sq <- sum(oMinuse)
x_sq
```

```
## [1] 35.33378
```

```
# this is bigger than the value given by the contigency table
qchisq(0.95, 1)
```

```
## [1] 3.841459
```

```
# you can get a a p value for this
1 - pchisq(x_sq, 1)
```

```
## [1] 2.777725e-09
```

```
# wow. very-significant we can reject the null hypothesis
# you will be unsurprised to learn R has an in-built function to do this
chisq.test(count)
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  count
## X-squared = 33.112, df = 1, p-value = 8.7e-09
```

```
# However, this is not the same as ours! Because Yate's correction has been applied
chisq.test(count, correct=F)
```

```
##
##  Pearson's Chi-squared test
##
## data:  count
## X-squared = 35.334, df = 1, p-value = 2.778e-09
```

```
# Excellent the same result!
```

Fisher's exact test

This is very similar to the chi-squared test above, except we have a situation in which an expected frequency is less than 5.

	column 1	column 2	row totals
row 1	a	b	a + b
row 2	c	d	c + d
column totals	a + c	b + d	n

The probability for an outcome has the scary-looking formula:

$$p = \frac{((a+b)!(c+d)!(a+c)!(b+d)!)!}{a!b!c!d!n!}$$

! means factorial

We have some data on ants and if they are found on a tree

	Tree A	Tree B	row totals
No Ants	6	2	8
Ants	4	8	12
column totals	10	10	20

In R we can calculate this probability

```
factorial(8) * factorial(12) * factorial(10) *
factorial(10) /
(factorial(6)*factorial(2)*factorial(4)*factorial(8)*
factorial(20))
```

```
## [1] 0.07501786
```

We need to consider two more extreme scenarios: Tree B having only one tree – the matrix totals would be 7, 1, 3, 9:

```
factorial(8) * factorial(12) * factorial(10) * factorial(10) / (factorial(7)*factorial(3)*factorial(1)*
factorial(20))
```

```
## [1] 0.009526078
```

Or a scenario with no ants in tree B

```
factorial(8) * factorial(12) * factorial(10) * factorial(10) / (factorial(8)*factorial(2)*factorial(0)*
```

```
## [1] 0.0003572279
```

Add this together

```
0.07501786 + 0.009526078 + 0.0003572279
```

```
## [1] 0.08490117
```

To turn this into a p value we need to double it to make it a *two-tailed* test

```
2*(0.07501786 + 0.009526078 + 0.0003572279)
```

```
## [1] 0.1698023
```

we have no reason to reject the null hypothesis – all this could have happened by chance.

```
x <- as.matrix(c(6,4,2,8))
dim(x) <- c(2,2)
fisher.test(x)
```

```
##
##  Fisher's Exact Test for Count Data
##
## data:  x
## p-value = 0.1698
## alternative hypothesis: true odds ratio is not equal to 1
## 95 percent confidence interval:
##   0.6026805 79.8309210
## sample estimates:
## odds ratio
##   5.430473
```

Fisher's Exact Test for Count Data – the p value is greater than 0.05